MELSEC System Q/L Series

Programmable Logic Controllers

Programming Manual

21112011
Version A

Programming Manual for the
MELSEC System Q and L Series
Art. No.:

Version			Changes / Additions / Corrections
A	11/2011	akl	First edition

About this Manual

The texts, illustrations, diagrams, and examples contained in this manual are intended exclusively as support material for the explanation, handling, programming, and operation of the programmable logic controllers of the MELSEC System Q and L series.

If you have any questions concerning the programming and operation of the equipment described in this manual, please contact your relevant sales office or department (refer to back of cover).

Current information and answers to frequently asked questions are also available through the Internet (www.mitsubishi-automation.com)

Contents

1 Introduction

1.1 Further manuals . . . 1-1
1.2 CPU types . . . 1-2
1.3 Software . . . 1-2
1.4 Finding an instruction . . . 1-3
1.5 PLC parameters . . . 1-4
1.6 Comparison between the software packages . . . 1-6

2 Instruction Tables

2.1 Subdivision of instructions . . . 2-1
2.2 Overview of instructions . . . 2-4
2.2.1 Description of the overview tables . . . 2-4
2.3 Sequence instructions . . . 2-6
2.3.1 Input instructions . . . 2-6
2.3.2 Connection instructions . . . 2-7
2.3.3 Output instruction . . . 2-8
2.3.4 Shift instructions . . . 2-9
2.3.5 Master control instructions . . . 2-9
2.3.6 Program termination instructions . . . 2-9
2.3.7 Miscellaneous instructions . . . 2-9
2.4 Application instructions, Part 1 . . . 2-10
2.4.1 Comparison operation instructions . . . 2-10
2.4.2 Arithmetic operation instructions . . . 2-14
2.4.3 Data conversion instructions . . . 2-20
2.4.4 Data transfer instructions . . . 2-23
2.4.5 Program branch instructions . . . 2-25
2.4.6 Interrupt program execution control instructions . . . 2-25
2.4.7 Data refresh instructions . . . 2-25
2.4.8 Other convenient instructions . . . 2-26
2.5 Application instructions, Part 2 . . . 2-27
2.5.1 Logical operation instructions . . . 2-27
2.5.2 Rotation instructions . . . 2-30
2.5.3 Shift instructions . . . 2-31
2.5.4 Bit processing instructions . . . 2-32
2.5.5 Data processing instructions . . . 2-33
2.5.6 Structured program instructions . . . 2-36
2.5.7 Data table operation instructions . . . 2-38
2.5.8 Buffer memory access instructions . . . 2-39
2.5.9 Display instructions . . . 2-39
2.5.10 Debugging and failure diagnosis instructions . . . 2-40
2.5.11 Character string processing instructions . . . 2-41
2.5.12 Special function instructions . . . 2-44

2.5.13 Data control instructions . . . 2-48
2.5.14 File register switching instructions . . . 2-50
2.5.15 Clock instructions . . . 2-51
2.5.16 Expansion clock instructions . . . 2-53
2.5.17 Program instructions . . . 2-54
2.5.18 Other instructions . . . 2-55
2.6 Data link instructions . . . 2-58
2.6.1 Instructions for network refresh . . . 2-58
2.6.2 Read/write routing information . . . 2-58
2.7 Multiple CPU dedicated instruction . . . 2-59
2.7.1 Instructions for writing to the CPU shared memory of host CPU . . . 2-59
2.7.2 Instructions for reading from the CPU shared memory of another CPU . . . 2-59
2.7.3 Multiple CPU high-speed transmission dedicated instructions . . . 2-60
2.8 System switching instruction for a redundant system . . . 2-61
2.9 Instructions for special function modules . . . 2-62
2.9.1 Instructions for serial communication modules . . . 2-62
2.9.2 Instructions for PROFIBUS/DP interface modules . . . 2-62
2.9.3 Instructions for ETHERNET interface modules . . . 2-63
2.9.4 Instruction for MELSECNET/H . . . 2-63
2.9.5 Instructions for CC-Link . . . 2-64

3 Configuration of Instructions

3.1 The structure of an instruction . . . 3-1
3.1.1 Source of data (s) . . . 3-1
3.1.2 Destination of data (d) . . . 3-2
3.1.3 Number (n) . . . 3-2
3.2 Notation of instructions . . . 3-3
3.2.1 16/32-bit and pulse . . . 3-3
3.2.2 MELSEC and IEC . . . 3-3
3.2.3 Further characteristics of the instruction notation . . . 3-5
3.2.4 Specification of the notation . . . 3-5
3.3 Programming of dedicated instructions . . . 3-6
3.4 Programming of variables . . . 3-7
3.4.1 Programming with the GX IEC Developer . . . 3-7
3.5 Data types . . . 3-9
3.5.1 Processing of data . . . 3-11
3.5.2 Addressing of arrays and registers in the GX IEC Developer . . . 3-22
3.5.3 Usage of character string data (STRING) . . . 3-25
3.6 Index qualification . . . 3-27
3.7 Indirect designation (GX Works2 only) . . . 3-39
3.8 Reducing instruction processing time . . . 3-42
3.8.1 Subset processing . . . 3-42
3.8.2 Operation processing with standard device registers (Z) (Universal model QCPU and LCPU only) . . . 3-43

3.9 Operation errors . . . 3-44
3.9.1 Verification of the device range . . . 3-45
3.9.2 Verification of the device data . . . 3-51
3.9.3 Buffer memory access . . . 3-51
3.9.4 Multiple CPU shared memory access . . . 3-51
3.10 Execution conditions of the instructions . . . 3-52
3.10.1 Execution condition . . . 3-52
3.10.2 EN input and ENO output . . . 3-53
3.11 Number of program steps . . . 3-55
3.12 Multiple Instructions using the same device . . . 3-61
3.12.1 OUT instructions using the same device . . . 3-61
3.12.2 SET/RST instructions using the same device . . . 3-62
3.12.3 PLS instructions using the same device . . . 3-63
3.12.4 PLF instructions using the same device . . . 3-64
3.13 Precautions for use of file registers . . . 3-66

4 Layout and Structure of the Chapters

4.1 Overview of the instructions . . . 4-2
4.2 The CPU table . . . 4-2
4.3 Devices . . . 4-3
4.4 Representation format of the instruction . . . 4-4
4.4.1 Representation in the GX IEC Developer . . . 4-4
4.4.2 Representation in GX Works2 . . . 4-4
4.5 Variables . . . 4-5
4.6 Functions . . . 4-5
4.7 Notes . . . 4-6
4.8 Operation errors . . . 4-6
4.9 Program examples . . . 4-7

5 Sequence Instructions

5.1 Input instructions . . . 5-4
5.1.1 LD, LDI, AND, ANI, OR, ORI . . . 5-4
5.1.2 LDP, LDF, ANDP, ANDF, ORP, ORF . . . 5-7
5.1.3 LDPI, LDFI, ANDPI, ANDFI, ORPI, ORFI . . . 5-10
5.2 Connection instructions . . . 5-13
5.2.1 ANB, ORB . . . 5-13
5.2.2 MPS, MRD, MPP . . . 5-15
5.2.3 INV . . . 5-18
5.2.4 MEP, MEF . . . 5-20
5.2.5 EGP, EGF . . . 5-22

5.3 Output instructions 5-24
5.3.1 OUT 5-24
5.3.2 OUT T, OUTH T 5-26
5.3.3 OUT C 5-30
5.3.4 OUT F 5-33
5.3.5 SET 5-35
5.3.6 RST 5-37
5.3.7 SET F, RST F 5-40
5.3.8 PLS, PLF 5-43
5.3.9 FF 5-47
5.3.10 DELTA, DELTAP 5-49
5.4 Shift instructions 5-51
5.4.1 SFT, SFTP 5-51
5.5 Master control instructions 5-54
5.5.1 MC, MCR 5-54
5.6 Termination instructions 5-60
5.6.1 FEND 5-60
5.6.2 END 5-63
5.7 Miscellaneous instructions 5-66
5.7.1 STOP 5-66
5.7.2 NOP, NOPLF, PAGE n 5-68

6 Application Instructions, Part 1

6.1 Comparison operation instructions 6-2
6.1.1 =, < >, >, < =, <, > = ... 6-6
6.1.2 D=, D<>, D>, D<=, D<, D>= 6-9
6.1.3 E=, E<>, E>, E< =, E<, E>= 6-12
6.1.4 ED=, ED<>, ED>, ED< =, ED<, ED>= 6-16
6.1.5 $ =, $ < >, $ >, $ < =, $ <, $ > = ... 6-20
6.1.6 BKCMP, BKCMPP 6-25
6.1.7 DBKCMP, DBKCMPP 6-30
6.2 Arithmetic operation Instructions 6-35
6.2.1 +, +P, -, -P 6-38
6.2.2 D+, D+P, D-, D-P 6-42
6.2.3 x, xP, /, /P 6-45
6.2.4 Dx, DxP, D/, D/P 6-49
6.2.5 B+, B+P, B-, B-P 6-52
6.2.6 DB+, DB+P, DB-, DB-P 6-56
6.2.7 Bx, BxP, B/, B/P 6-60
6.2.8 DBx, DBxP, DB/, DB/P 6-63
6.2.9 E+, E+P, E-, E-P 6-67
6.2.10 ED+, ED+P, ED-, ED-P 6-72
6.2.11 Ex, ExP, E/, E/P 6-77
6.2.12 EDx, EDxP, ED/, ED/P 6-80

6.2.13 BK+, BK+P, BK-, BK-P 6-83
6.2.14 DBK+, DBK+P, DBK-, DBK-P 6-87
6.2.15 $+, $+P 6-91
6.2.16 INC, INCP, DEC, DECP 6-94
6.2.17 DINC, DINCP, DDEC, DDECP 6-97
6.3 Data conversion instructions 6-101
6.3.1 BCD, BCDP, DBCD, DBCDP 6-103
6.3.2 BIN, BINP, DBIN, DBINP 6-106
6.3.3 FLT, FLTP, DFLT, DFLTP 6-110
6.3.4 FLTD, FLTPD, DFLTD, DFLTPD 6-113
6.3.5 INT, INTP, DINT, DINTP 6-116
6.3.6 INTD, INTPD, DINTD, DINTPD 6-120
6.3.7 DBL, DBLP 6-123
6.3.8 WORD, WORDP 6-125
6.3.9 GRY, GRYP, DGRY, DGRYP 6-127
6.3.10 GBIN, GBINP, DGBIN, DGBINP 6-130
6.3.11 NEG, NEGP, DNEG, DNEGP 6-133
6.3.12 ENEG, ENEGP 6-136
6.3.13 EDNEG, EDNEGP 6-138
6.3.14 BKBCD, BKBCDP 6-140
6.3.15 BKBIN, BKBINP 6-143
6.3.16 ECON, ECONP 6-146
6.3.17 EDCON, EDCONP 6-148
6.4 Data transfer instructions 6-150
6.4.1 MOV, MOVP, DMOV, DMOVP 6-151
6.4.2 EMOV, EMOVP 6-154
6.4.3 EDMOV, EDMOVP 6-156
6.4.4 $MOV, $MOVP 6-158
6.4.5 CML, CMLP, DCML, DCMLP 6-161
6.4.6 BMOV, BMOVP 6-166
6.4.7 FMOV, FMOVP 6-170
6.4.8 DFMOV, DFMOVP 6-173
6.4.9 XCH, XCHP, DXCH, DXCHP 6-176
6.4.10 BXCH, BXCHP 6-179
6.4.11 SWAP, SWAPP 6-182
6.5 Program branch instructions 6-185
6.5.1 CJ, SCJ, JMP 6-186
6.5.2 GOEND 6-190
6.6 Program execution control instructions 6-192
6.6.1 DI, EI, IMASK 6-193
6.6.2 IRET 6-200
6.7 Link refresh instructions 6-202
6.7.1 RFS, RFSP 6-203

6.8 Other convenient instructions 6-205
6.8.1 UDCNT1 6-206
6.8.2 UDCNT2 6-209
6.8.3 TTMR 6-212
6.8.4 STMR 6-214
6.8.5 ROTC 6-218
6.8.6 RAMP 6-222
6.8.7 SPD 6-225
6.8.8 PLSY 6-228
6.8.9 PWM 6-230
6.8.10 MTR 6-232

7 Application Instructions, Part 2

7.1 Logical operation instructions 7-2
7.1.1 WAND, WANDP, DAND, DANDP 7-4
7.1.2 BKAND, BKANDP 7-10
7.1.3 WOR, WORP, DOR, DORP 7-13
7.1.4 BKOR, BKORP 7-18
7.1.5 WXOR, WXORP, DXOR, DXORP 7-21
7.1.6 BKXOR, BKXORP 7-26
7.1.7 WXNR, WXNRP, DXNR, DXNRP 7-29
7.1.8 BKXNR, BKXNRP 7-35
7.2 Data rotation instructions 7-38
7.2.1 ROR, RORP, RCR, RCRP 7-39
7.2.2 ROL, ROLP, RCL, RCLP 7-42
7.2.3 DROR, DRORP, DRCR, DRCRP 7-45
7.2.4 DROL, DROLP, DRCL, DRCLP 7-48
7.3 Data shift instructions 7-51
7.3.1 SFR, SFRP, SFL, SFLP 7-52
7.3.2 BSFR, BSFRP, BSFL, BSFLP 7-55
7.3.3 SFTBR, SFTBRP, SFTBL, SFTBLP 7-58
7.3.4 DSFR, DSFRP, DSFL, DSFLP 7-61
7.3.5 SFTWR, SFTWRP, SFTWL, SFTWLP 7-64
7.4 Bit processing instructions 7-67
7.4.1 BSET, BSETP, BRST, BRSTP 7-68
7.4.2 TEST, TESTP, DTEST, DTESTP 7-71
7.4.3 BKRST, BKRSTP 7-75
7.5 Data processing instructions 7-79
7.5.1 SER, SERP, DSER, DSERP 7-81
7.5.2 SUM, SUMP, DSUM, DSUMP 7-87
7.5.3 DECO, DECOP 7-90
7.5.4 ENCO, ENCOP 7-93
7.5.5 SEG, SEGP 7-96
7.5.6 DIS, DISP 7-100

7.5.7 UNI, UNIP 7-103
7.5.8 NDIS, NDISP, NUNI, NUNIP 7-106
7.5.9 WTOB, WTOBP, BTOW, BTOWP 7-111
7.5.10 MAX, MAXP, DMAX, DMAXP 7-116
7.5.11 MIN, MINP, DMIN, DMINP 7-119
7.5.12 SORT, DSORT 7-122
7.5.13 WSUM, WSUMP 7-126
7.5.14 DWSUM, DWSUMP 7-128
7.5.15 MEAN, MEANP, DMEAN, DMEANP 7-130
7.6 Structured program instructions 7-133
7.6.1 FOR, NEXT 7-134
7.6.2 BREAK, BREAKP 7-137
7.6.3 CALL, CALLP 7-140
7.6.4 RET 7-146
7.6.5 FCALL, FCALLP 7-148
7.6.6 ECALL, ECALLP 7-153
7.6.7 EFCALL, EFCALLP 7-159
7.6.8 XCALL 7-164
7.6.9 COM (Refresh) 7-170
7.6.10 COM (Selective Refresh) 7-173
7.6.11 CCOM, CCOMP 7-177
7.6.12 IX, IXEND 7-179
7.6.13 IXDEV, IXSET 7-184
7.7 Data table operation instructions 7-187
7.7.1 FIFW, FIFWP 7-188
7.7.2 FIFR, FIFRP 7-192
7.7.3 FPOP, FPOPP 7-196
7.7.4 FDEL, FDELP, FINS, FINSP 7-200
7.8 Buffer memory access instructions 7-207
7.8.1 FROM, FROMP, DFRO, DFROP 7-208
7.8.2 TO, TOP, DTO, DTOP 7-212
7.9 Display instructions 7-217
7.9.1 PR 7-218
7.9.2 PRC 7-223
7.9.3 LEDR 7-227
7.10 Failure diagnosis and debugging 7-231
7.10.1 CHKST, CHK 7-232
7.10.2 CHKCIR, CHKEND 7-240

7.11 Character string processing instructions 7-245
7.11.1 BINDA, BINDAP, DBINDA, DBINDAP 7-248
7.11.2 BINHA, BINHAP, DBINHA, DBINHAP 7-253
7.11.3 BCDDA, BCDDAP, DBCDDA, DBCDDAP 7-258
7.11.4 DABIN, DABINP, DDABIN, DDABINP 7-263
7.11.5 HABIN, HABINP, DHABIN, DHABINP 7-268
7.11.6 DABCD, DABCDP, DDABCD, DDABCDP 7-272
7.11.7 COMRD, COMRDP 7-276
7.11.8 LEN, LENP 7-280
7.11.9 STR, STRP, DSTR, DSTRP 7-283
7.11.10 VAL, VALP, DVAL, DVALP 7-290
7.11.11 ESTR, ESTRP 7-296
7.11.12 EVAL, EVALP 7-306
7.11.13 ASC, ASCP 7-312
7.11.14 HEX, HEXP 7-315
7.11.15 RIGHT, RIGHTP, LEFT, LEFTP 7-319
7.11.16 MIDR, MIDRP, MIDW, MIDWP 7-323
7.11.17 INSTR, INSTRP 7-329
7.11.18 STRINS, STRINSP 7-333
7.11.19 STRDEL, STRDELP 7-336
7.11.20 EMOD, EMODP 7-339
7.11.21 EREXP, EREXPP 7-343
7.12 Special functions 7-347
7.12.1 SIN, SINP 7-351
7.12.2 SIND, SINDP 7-354
7.12.3 COS, COSP 7-357
7.12.4 COSD, COSDP 7-360
7.12.5 TAN, TANP 7-363
7.12.6 TAND, TANDP 7-366
7.12.7 ASIN, ASINP 7-369
7.12.8 ASIND, ASINDP 7-372
7.12.9 ACOS, ACOSP 7-375
7.12.10 ACOSD, ACOSDP 7-378
7.12.11 ATAN, ATANP 7-381
7.12.12 ATAND, ATANDP 7-384
7.12.13 RAD, RADP 7-387
7.12.14 RADD, RADDP 7-390
7.12.15 DEG, DEGP 7-393
7.12.16 DEGD, DEGDP 7-396
7.12.17 POW, POWP 7-399
7.12.18 POWD, POWDP 7-402
7.12.19 SQR, SQRP 7-405
7.12.20 SQRD, SQRDP 7-408
7.12.21 EXP, EXPP 7-411

7.12.22 EXPD, EXPDP 7-414
7.12.23 LOG, LOGP 7-417
7.12.24 LOGD, LOGDP 7-420
7.12.25 LOG10, LOG10P 7-423
7.12.26 LOG10D, LOG10DP 7-426
7.12.27 RND, RNDP, SRND, SRNDP 7-429
7.12.28 BSQR, BSQRP, BDSQR, BDSQRP 7-431
7.12.29 BSIN, BSINP 7-435
7.12.30 BCOS, BCOSP 7-438
7.12.31 BTAN, BTANP 7-441
7.12.32 BASIN, BASINP 7-444
7.12.33 BACOS, BACOSP 7-447
7.12.34 BATAN, BATANP 7-450
7.13 Data control instructions 7-453
7.13.1 LIMIT, LIMITP, DLIMIT, DLIMITP 7-454
7.13.2 BAND, BANDP, DBAND, DBANDP 7-458
7.13.3 ZONE, ZONEP, DZONE, DZONEP 7-462
7.13.4 SCL, SCLP, DSCL, DSCLP 7-466
7.13.5 SCL2, SCL2P, DSCL2, DSCL2P 7-471
7.14 File register switching instructions 7-475
7.14.1 RSET, RSETP 7-476
7.14.2 QDRSET, QDRSETP 7-479
7.14.3 QCDSET, QCDSETP 7-482
7.15 Clock instructions 7-485
7.15.1 DATERD, DATERDP 7-487
7.15.2 DATEWR, DATEWRP 7-490
7.15.3 DATE+, DATE+P 7-493
7.15.4 DATE-, DATE-P 7-498
7.15.5 SECOND, SECONDP, HOUR, HOURP 7-503
7.15.6 DT=, DT<>, DT>, DT<=, DT<, DT>= 7-509
7.15.7 TM=, TM<>, TM>, TM<=, TM<, TM>= 7-514
7.16 Expansion clock instructions 7-519
7.16.1 S.DATERD, SP.DATERP 7-520
7.16.2 S.DATE+, SP.DATE+ 7-524
7.16.3 S.DATE-, SP.DATE- 7-528
7.17 Program control instructions 7-532
7.17.1 PSTOP, PSTOPP 7-534
7.17.2 POFF, POFFP 7-536
7.17.3 PSCAN, PSCANP 7-539
7.17.4 PLOW, PLOWP 7-541
7.17.5 PCHK 7-543

7.18 Other convenient instructions. 7-546
7.18.1 WDT, WDTP 7-547
7.18.2 DUTY 7-549
7.18.3 TIMCHK 7-552
7.18.4 ZRRDB, ZRRDBP 7-554
7.18.5 ZRWRB, ZRWRBP 7-558
7.18.6 ADRSET, ADRSETP 7-562
7.18.7 KEY 7-564
7.18.8 ZPUSH, ZPUSHP, ZPOP, ZPOPP 7-570
7.18.9 UNIRD, UNIRDP 7-574
7.18.10 TYPERD, TYPERDP 7-580
7.18.11 TRACE, TRACER 7-586
7.18.12 SP.FWRITE 7-588
7.18.13 SP.FREAD 7-599
7.18.14 SP.DEVST 7-612
7.18.15 S.DEVLD, SP.DEVLD 7-615
7.18.16 PLOADP 7-617
7.18.17 PUNLOADP 7-621
7.18.18 PSWAPP 7-624
7.18.19 RBMOV, RBMOVP 7-627
7.18.20 UMSG 7-634

8 Data Link Instructions

8.1 Categories of instructions. 8-1
8.2 Data refresh instructions 8-1
8.2.1 S.ZCOM, SP.ZCOM 8-2
8.3 Reading and writing routing information. 8-7
8.3.1 S.RTREAD, SP.RTREAD 8-8
8.3.2 S.RTWRITE, SP.RTWRITE 8-10

9 Multiple CPU Dedicated Instructions

9.1 Writing to the CPU shared memory of host CPU 9-2
9.1.1 S.TO, SP.TO. 9-4
9.1.2 TO, TOP, DTO, DTOP 9-8
9.2 Read from CPU shared memory of another station 9-13
9.2.1 FROM, FROMP, DFRO, DFROP 9-14

10 Multiple CPU Device Write/Read Instructions

10.1 Overview 10-1
10.2 Multiple CPU high-speed transmission instructions 10-13
10.2.1 D.DDWR, DP.DDWR 10-13
10.2.2 D.DDRD, DP.DDRD 10-18

11 Instructions for MELSEC System Q

11.1 Instruction for a redundant system 11-1
11.1.1 SP.CONTSW 11-2

12 Instructions for Special Function Modules

12.1 Instructions for serial communication modules 12-2
12.1.1 BUFRCVS 12-3
12.1.2 GETE, GETEP 12-6
12.1.3 PUTE, PUTEP 12-11
12.1.4 PRR, PRRP 12-18
12.2 Instructions for PROFIBUS/DP interface modules 12-26
12.2.1 BBLKRD, BBLKRDP 12-27
12.2.2 BBLKWR, BBLKWRP 12-30
12.3 Instructions for ETHERNET interface modules 12-33
12.3.1 BUFRCV 12-34
12.3.2 BUFRCVS 12-39
12.3.3 BUFSND 12-42
12.3.4 OPEN 12-47
12.3.5 CLOSE 12-56
12.3.6 ERRCLR 12-61
12.3.7 ERRRD 12-67
12.3.8 UINI 12-72
12.4 Instructions for MELSECNET/H 12-78
12.4.1 PAIRSET 12-79
12.5 Instructions for CC-Link 12-82
12.5.1 RLPASET 12-83
12.5.2 RIRD 12-95
12.5.3 RIWT 12-103
12.5.4 RIRCV 12-111
12.5.5 RISEND 12-119
12.5.6 RITO 12-126
12.5.7 RIFR 12-130

13 Error Codes

13.1 Error code list . . . 13-1
13.1.1 How to read the error code list . . . 13-1
13.1.2 Types of error codes . . . 13-2
13.1.3 Clearing an error . . . 13-2
13.2 Error code list (1000 to 1999) . . . 13-3
13.3 Error code list (2000 to 2999) . . . 13-19
13.4 Error code list (3000 to 3999) . . . 13-39
13.5 Error code list (4000 to 4999) . . . 13-57
13.6 Error code list (5000 to 5999) . . . 13-71
13.7 Error code list (6000 to 6999) . . . 13-73
13.8 Error code list (7000 to 10000) . . . 13-81
13.9 Error codes returned to request source . . . 13-85

A Appendix A

A.1 Definition of the processing times . . . A-1
A.2 Processing times for MELSEC System Q CPUs . . . A-2
A.2.1 Table of Processing Times . . . A-3
A.2.2 Instructions executable by the product with the first 5 digits of the serial No. "04122" or higher (Basic model QCPU) . . . A-22
A.2.3 Table of the time to be added (Basic model QCPU) . . . A-25
A.2.4 Instructions availabe from function version B (High Performance model QCPU/Process CPU/Redundant CPU) . . . A-26
A.2.5 Table of the time to be added (High Performance model QCPU/Process CPU/Redundant CPU) . . . A-27
A.2.6 Redundant system instruction . . . A-27
A.3 Operation Processing Time of Universal Model QCPU . . . A-28
A.3.1 Subset instruction processing time . . . A-28
A.3.2 Processing time of instructions other than subset instruction . . . A-42
A.4 Operation Processing Time of LCPU . . . A-79
A.4.1 Subset instruction processing time . . . A-79
A.4.2 Processing time of instructions other than subset instruction . . . A-86
A.5 Comparison of the CPUs . . . A-102
A.5.1 Available devices . . . A-102
A.5.2 I/O control modes . . . A-104
A.5.3 Data types . . . A-104
A.5.4 Timer comparison . . . A-105
A.5.5 Comparison of counters . . . A-109
A.5.6 Comparison of display instructions . . . A-110
A.5.7 QCPU, LCPU instructions whose designation format has been changed . . . A-111
A.5.8 AnACPU and AnUCPU dedicated instructions . . . A-112

A.6 Table of special relays A-113
A.6.1 Diagnostic information A-114
A.6.2 System information A-117
A.6.3 System clocks A-122
A.6.4 Scan information A-124
A.6.5 I/O refresh A-124
A.6.6 Drive information A-125
A.6.7 Instruction related special relays A-128
A.6.8 Debugging A-131
A.6.9 Conversion from A series to System Q or L series A-132
A.6.10 Built-in Ethernet port and built-in Ethernet function A-139
A.6.11 Process control instruction A-140
A.6.12 Redundant system (host system CPU information) A-140
A.6.13 Redundant system (other system CPU information) A-144
A.6.14 Redundant system (tracking information) A-145
A.6.15 Redundant power supply module information A-148
A.6.16 Built-in I/O function A-149
A.6.17 Data logging A-154
A.7 Table of special registers A-156
A.7.1 Diagnostic information A-157
A.7.2 System information A-170
A.7.3 System clocks/counters A-181
A.7.4 Scan information A-182
A.7.5 Memory cards A-185
A.7.6 Instruction related registers A-194
A.7.7 Debugging A-198
A.7.8 Redundant CPU information (host system CPU information) A-198
A.7.9 Remote password count A-198
A.7.10 Conversion from A series to System Q or L series A-199
A.7.11 Built-in Ethernet port QCPU and built-in Ethernet function A-207
A.7.12 Fuse blown module A-210
A.7.13 I/O module verification A-210
A.7.14 Process control instruction A-211
A.7.15 Redundant system (host system CPU information) A-212
A.7.16 Redundant system (other system CPU information) A-214
A.7.17 Redundant system (tracking information) A-217
A.7.18 Redundant power supply module information A-218
A.7.19 Built-in I/O function A-219
A.7.20 Data logging A-227

1 Introduction

This manual describes the programming and processing of the sequence and application instructions that are provided by the CPUs of the MELSEC System Q and L series.

1.1 Further manuals

Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)
–Description of functions, methods, and devices for programming

QnUCPU User's Manual (Function Explanation, Program Fundamentals)
–Description of functions, methods, and devices for programming

QnUCPU User's Manual (Communication via Built-in Ethernet Port)
–Description of functions for the communication via built-in Ethernet port of CPU module

MELSEC-L CPU Module User's Manual (Function Explanation, Program Fundamentals)
–Description of functions, methods, and devices for programming

MELSEC-L CPU Module User's Manual (Communication via Built-in Ethernet Port)
–Description of functions for the communication via built-in Ethernet port of CPU module

MELSEC-L CPU Module User's Manual (Data Logging Function)
–Description of data logging functionality of CPU module

MELSEC-Q/L Programming Manual (Common Instructions)
–Description of how to use sequence instructions, basic instructions, and application instructions

MELSEC-Q/L/QnA Programming Manual (SFC)
–Description of the instructions for sequential function charts (MELSAP3)

MELSEC-Q/L Programming Manual (MELSAP-L)
–Description of the instructions for sequential function charts (MELSAP-L)

MELSEC-Q/L Programming Manual (Structured Text)
–Description of programming methods using structured languages

MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)
–Description of the PID control instructions

QnPH/QnPRHCPU Programming Manual (Process Control Instructions)
–Description of the dedicated instructions for performing process control

NOTE *You can download all manuals as PDF from the MITSUBISHI ELECTRIC homepage (www.mitsubishi-automation.com).*

1.2 CPU types

The functions described in this manual can be transferred to all CPU types by the current version of the GX Works2 provided that the according CPU supports the instructions.

The different PLC types with their specific CPU are listed below in detail:

PLC Type	CPU Type	CPU Module Model
MELSEC System Q	Basic model	Q00JCPU, Q00CPU, Q01CPU
	High Performance model	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU
	Process model	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU
	Redundant model	Q12PRHCPU, Q25PRHCPU
	Universal model	Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU
L series		L02CPU, L26CPU-BT

If, e.g. in tables, QCPU or LCPU is mentioned, all CPU types of the MELSEC System Q and L series are included. Exceptions are marked separately.

1.3 Software

All the described instructions can be applied with the available software packages:

– GX Developer

– GX IEC Developer

– GX Works2

The program examples contained in this manual were created with the GX Works2.

Corresponding to the selected CPU only those instructions are available within the GX Works2 dialog box that can actually be processed by the CPU.

NOTE *The programming tool GX IEC Developer does not support the CPU modules of the L series.*

1.4 Finding an instruction

Advanced

If you are already familiar with the programming of instructions for the MELSEC System Q, look up the instruction chapters 5 through 12. The header line contains the name of the instruction as it is applied within GX Works2.

Beginners

If you are not really familiar with the handling of the instructions, proceed as follows:

- Read through chapter 3 regarding the differing representation of instructions within the MELSEC and the IEC editor.
- Read through chapter 4 regarding the consistent layout and structure of each description of instruction.
- Use

  –- the tabular overview of instruction categories with brief descriptions in chapter 2

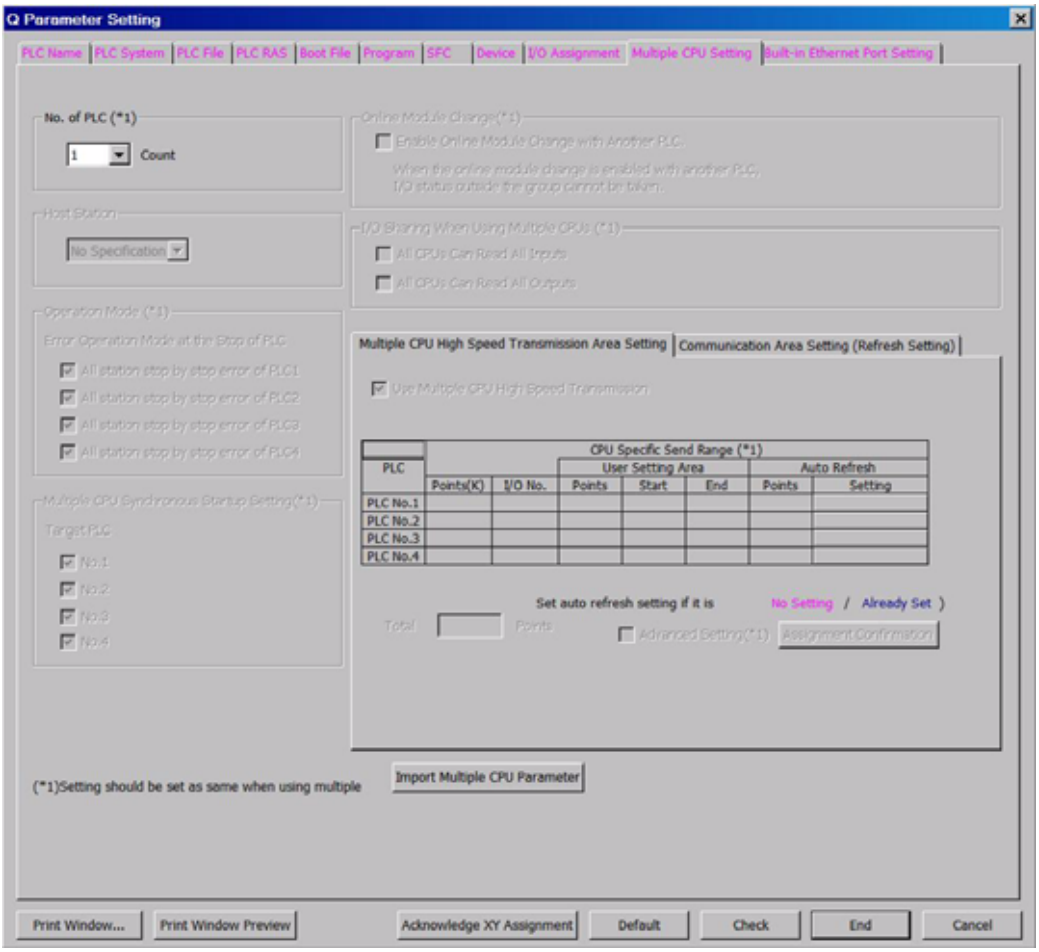
  –- the index containing the entire instructions

NOTE *All the instructions contained in this manual are also included within the online help of the GX Works2 as detailed as here.*

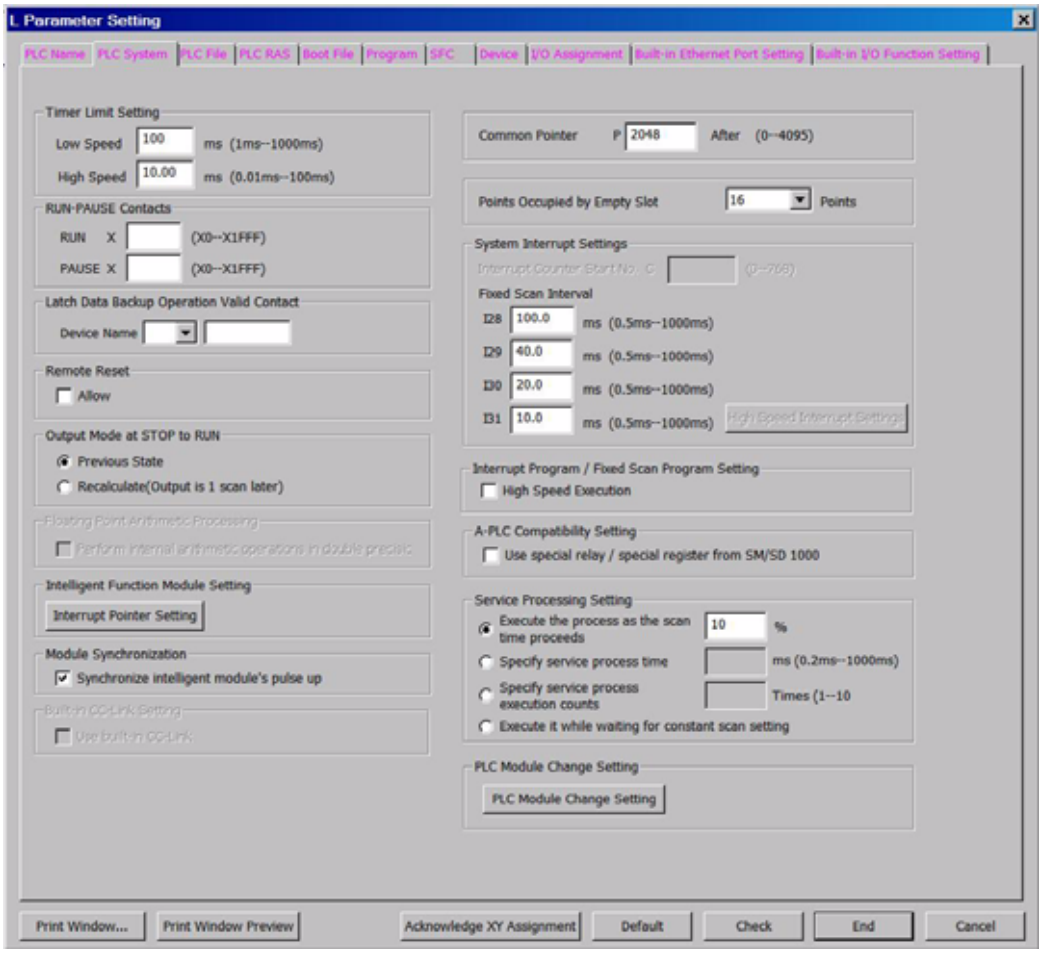
1.5 PLC parameters

Via parameters several functions, device ranges, etc. are set up. For the programming of the functions described in this manual, the parameter settings can remain preset or customised to the user´s needs. Refer to the according hardware manuals of the CPUs and programming manuals for detailed descriptions of the PLC parameter settings.

Example: MELSEC System Q



Example: L series



1.6 Comparison between the software packages

The most important features of the GX IEC Developer, the GX Developer, and the GX Works2 are listed in the following table:

GX IEC Developer	GX Developer	GX Works2
Structured use	Simple to use	Simple and structured use
Programming in comply with IEC (6)1131-3	—	—
Editors: • Instruction List • Ladder Diagram • Structured Text (ST) • Sequential Function Chart (SFC) • Function Block (FUB)	Editors: • Instruction List • Ladder Diagram • Sequential Function Chart (SFC)	Editors: • Ladder Diagram • Structured Text (ST) • Sequential Function Chart (SFC) • Structured Ladder Diagram
Functions and Function Blocks	Function Blocks (V 7.0 or later)	Functions and Function Blocks
Program modifications in online mode	Program modifications in online mode Program change in online mode	Program modifications in online mode Program change in online mode
Diagnostic functions for the PLC	Diagnostic functions for the PLC	Diagnostic functions for the PLC
Diagnostic functions for network systems	Diagnostic functions for network systems	Diagnostic functions for network systems

2 Instruction Tables

2.1 Subdivision of instructions

The instructions are subdivided into the following categories:

- Sequence instructions
- Application instructions (Part 1 and Part 2)
- Data link instructions
- Multiple CPU dedicated instruction
- Multiple CPU high-speed transmission dedicated instructions
- Redundant system instruction
- Instructions for special function modules

The categories of instructions are described in detail in the following table:

Category of Instruction		Description	Reference Section
Sequence instructions	Input instructions	Operation start, series and parallel connection of contacts	5.1
	Connection instructions	Series and parallel block connection, storage and processing of operation results, inversion of operation results, conversion of operation results into pulses, setting of edge relays	5.2
	Output instructions	Bit devices, counter and timer contacts, output, setting, and resetting of annunciators, setting and resetting of devices, leading edge and trailing edge output, bit device output inversion, generating pulses	5.3
	Shift instructions	Shifting bit devices	5.4
	Master control instructions	Setting and resetting single parts of a program	5.5
	Termination instructions	End of a part of program, end of sequence and routine programs	5.6
	Miscellaneous instructions	Sequence program stop, no operation	5.7
Application instructions Part 1	Comparison operation instructions	Compares data to data (e.g. =, >, ≥)	6.1
	Arithmetic operation instructions	Adds, subtracts, multiplies, divides, increments, and decrements BIN and BCD data, floating point data, and BIN block data, links character strings	6.2
	Data convsersion instruction	Converts data types, e.g. BCD → BIN, BIN → BCD	6.3
	Data transfer instructions	Transmits designated data	6.4
	Program branch instructions	Program jump commands	6.5
	Program execution control instructions	Enables and disables program interrupts	6.6
	Refresh instructions	Refreshes bit devices, links, and I/O interfaces	6.7
	Other convenient instructions	Count 1- or 2-phase input up or down, teaching timer, special function timer, rotary table near path rotation control, ramp signal, pulse density measurement, fixed cycle pulse output, pulse width modulation, matrix input	6.8

Category of Instruction		Description	Reference Section
Application instructions Part 2	Logical operation instructions	Logical AND / OR, logical exclusive OR / exclusive NOR	7.1
	Rotation instructions	16-bit and 32-bit data right / left rotation	7.2
	Shift instructions	Shift data by bit or word	7.3
	Bit processing instructions	Set, reset, and test bits	7.4
	Data processing instructions	Search, encode, and decode data at specified devices Disunite and unite data	7.5
	Structured program instructions	Repeated operation, subroutine program calls, subroutine calls between program files, switching between main and subprogram parts, micro computer program calls, index qualification of entire ladders, store index qualification values in data tables	7.6
	Data table operation instructions	Write to and read data from a data table, delete and insert data blocks in a data table	7.7
	Buffer memory access instructions	Buffer memory access of special function modules or remote modules	7.8
	Display instructions	Output ASCII characters to the outputs of a module or to an LED display	7.9
	Debugging and failure diagnosis instructions	Failure checks, setting and resetting status latch, sampling trace, program trace	7.10
	Character string processing instructions	Character string (ASCII code) processing	7.11
	Special function instructions	Trigonometrical functions, square root and exponential calculation with BCD data and floating point data	7.12
	Data control instructions	Upper and lower limit control and storage of checked data	7.13
	File register switching instructions	Switching between file register blocks and files	7.14
	Clock instructions	Reading/writing of the values of year, month, day, hour, minute, second, and day of the week; addition/ subtraction of the values of hour, minute, and second; conversion of the values of hour, minute, and second into second; comparison between the values of year, month, and day; and comparison between the values of hour, minute, and second.	7.15
	Expansion clock instructions	Reading of the values of year, month, day, hour, minute, second, millisecond, and day of the week; addition/ subtraction of the values of hour, minute, second, and millisecond	7.16
	Program instructions	Select different program execution modes	7.17
	Other instructions	Reset watchdog timer (WDT), pulse generation, direct read from indirect access file registers, numerical key input from keyboard, batch save or recovery of index registers, reading module information/model name, trace set/trace reset, writing to and reading from files/standard ROM, program instructions, data transfer, user message	7.18

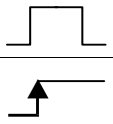
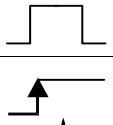
Category of Instruction		Description	Reference Section
Data link instructions	Network refresh instructions	Instructions for data refresh operations in network modules.	8.2
	Read/write routing information	Read and write routing parameters (network number and station number of relay station, station number of routing station).	8.3
Data exchange instructions in a multi-CPU system		Writing to the CPU shared memory Reading from the CPU shared memory of another CPU	9.1 9.2
Multiple CPU high-speed transmission dedicated instructions		Writes/reads devices to/from another CPU.	chapter 10
Instruction for a redundant system		System switching (Active system/standby system)	chapter 11
Instructions for special function modules		Instructions for serial communication modules, PROFIBUS/DP interface modules, ETHERNET interface modules, MELSECNET/H and CC-Link	chapter 12

2.2 Overview of instructions

2.2.1 Description of the overview tables

The following sections 2.3 through 2.6 include an overview of all instructions described in this manual.

In the following the layout of the overview table is described in detail:

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Addition and subtraction of 16-bit binary data	+	s, d	(d)+(s) → (d)		3	●	6.2.1
	+P						
	+	s1, s2, d1	(s1)+(s2) → (d1)		4	●	6.2.1
	+P						

(1) (2) (3) (4) (5) (6) (7) (8)

Explanation of the different columns:

(1) Category of instruction

(2) Specification of instruction name ("command") for the programming

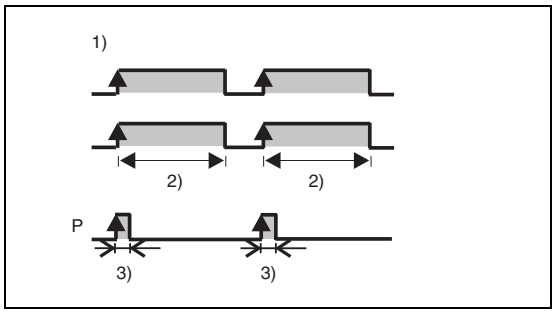
The instruction names are represented in MELSEC notation (refer to section 3.2 "Notation of instructions" for explanation of the notation).

In general, 16-bit instructions are represented. All 32-bit instructions are indicated by a leading "D".

Example: 16-bit instruction: +
32-bit instruction: D+

Pulse instructions, i.e. instructions that are only executed at leading edge of a signal are indicated by an appended "P".

Example: Execution when ON: +
Execution at leading edge: +P



[1] Execution condition of instruction
[2] One program scan
[3] One execution

Instructions processing character strings are indicated by a leading "$"
Example: Standard instructions: +
Character string instruction: $+P

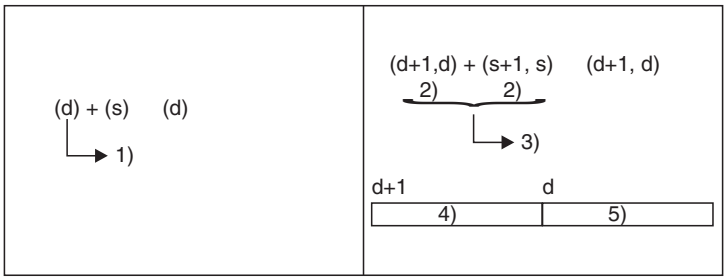
(3) Specification of variables

Here, the variables to be used are specified. The data source is represented by an "s", the data destination is represented by a "d".

Example:

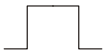
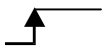
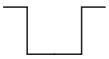
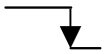
s	= if there is only one data source
s1, s2	= if there are several data sources
s+0, s+1, (s1)+0, (s1)+1	= for 32-bit instructions e.g. s1 = data register D0, (s1)+1 = data register D1
s+0, s+1, s+2, s+3	= 4 successive devices, e.g. for an array

(4) Meaning and processing of the entire control instruction



[1] Indicates 16 bits
[2] 16 bits
[3] Indicates 32 bits
[4] upper 16 bits
[5] lower 16 bits

(5) Indication of the execution condition according to the following table

Symbol	Execution condition
no indication	The instruction is executed continuously and independent from the prior execution condition. If the precondition is not set, the instruction is not executed.
	The instruction is executed as long as the precondition is ON. If the precondition is OFF, the instruction is not executed and no processing is conducted.
	This instruction is a pulsed instruction. It is only executed once and at leading edge of the input signal (when the precondition alters from OFF to ON). Afterwards, the instruction will not be executed any longer even if the input signal is still ON.
	Executed during OFF; instruction is executed only while the precondition is OFF. If the precondition is ON, the instruction is not executed, and no processing is conducted.
	This instruction is a pulsed instruction as well. It is only executed once and at trailing edge of the input signal (when the precondition alters from ON to OFF). Afterwards, the instruction will not be executed any longer even if the input signal is still OFF.

(6) Indication of the number of program steps

Indicated is the number of steps that is required for the entire execution of the instruction. Refer to section 3.11 for details.

(7) The ● mark indicates instructions for which subset processing is possible.

Refer to section 3.8.1 for details on subset processing.

(8) Indication of the reference section

Indicates the chapter and section of this manual where the instruction is described in detail.

2.3 Sequence instructions

2.3.1 Input instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Input instruction	LD	s	Operation start (Load (normally open contact))		①	●	5.1.1
	LDI		Operation start (Load (normally closed contact))				
	AND	s	Series connection (of NO contacts)				
	ANI		Series connection (of NC contacts)				
	OR	s	Parallel connection (of NO contacts)				
	ORI		Parallel connection (of NC contacts)				
	LDP	s	Pulse operation start (leading edge)		①	●	5.1.2
	LDF		Pulse operation start (trailing edge)				
	ANDP	s	Pulse series connection (leading edge)				
	ANDF	s	Pulse series connection (trailing edge)				
	ORP	s	Pulse parallel connection (leading edge)				
	ORF	s	Pulse parallel connection (trailing edge)				
Input instruction	LDPI	s	Starts leading edge pulse NOT operation		3 ②	●	5.1.3
	LDFI	s	Starts trailing edge pulse NOT operation		3 ②		
	ANDPI	s	Leading edge pulse NOT series connection		4 ②		
	ANDFI	s	Trailing edge pulse NOT series connection		4 ②		
	ORPI	s	Leading edge pulse NOT parallel connection		4 ②		
	ORFI	s	Trailing edge pulse NOT parallel connection		4 ②		

① The number of program steps depends on the devices used.
- For the use of internal devices or file registers (R0 through R32767) : 1
- For the use of a direct access input (DX) : 2
- For the use of other devices : 3

② The number of program steps depends on the devices and types of CPU modules used.
- For the use of internal devices or file registers (R0 through R32767) : 1
- For the use of a direct access input (DX) : 1
- For the use of other devices : 3

The number of program steps depends on the devices used.
- For the use of internal devices or file registers (R0 through R32767) : Number of basic steps
- Serial number access format file register (ZR), Extended data register (D), Extended link register (W), Multiple CPU shared device (U3En\G10000) : Number of basic steps + 1
- For the use of a direct access input (DX) : Number of basic steps + 1
- For the use of other devices : Number of basic steps + 2

2.3.2 Connection instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Connection instruction	ANB	—	Block series connection (Ladder block series connection)		1		5.2.1
	ORB		Block parallel connection (Ladder block parallel connection)				
	MPS	—	Operation result processing (Store operation result (memory push))		1		5.2.2
	MRD		Operation result processing (Read operation result (memory read))				
	MPP		Operation result processing (Read and clear operation result (memory pop))				
	INV	—	Operation result inversion (Inversion instruction)		1		5.2.3
	MEP	—	Operation result into pulse conversion (Pulse generation at leading edge of operation result)		1		5.2.4
	MEF		Operation result into pulse conversion (Pulse generation at trailing edge of operation result)				
	EGP	d	Setting of edge relays (Setting an edge relay with leading edge of an operation result)		1		5.2.5
	EGF		Setting of edge relays (Setting an edge relay with trailing edge of an operation result)		①		

① The number of program steps depends on the devices and types of CPU modules used.
- High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU, LCPU : 1
- Basic Model QCPU : 2

2.3.3 Output instruction

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Output instruction	OUT	d	Setting instructions for outputs		①		5.3.1
	OUT T OUTH T	d	Timers		4		5.3.2
	OUT C	d	Counter		4		5.3.3
	OUT F	d	Annunciator output		4		5.3.4
	SET	d	Setting of devices	②	③		5.3.5
	RST	d	Resetting devices	②	④		5.3.6
	SET F RST F	d	Setting and resetting the annunciators		2		5.3.7
	PLS PLF	d	Output at leading edge Output at trailing edge		2		5.3.8
	FF	s	Inversion of bit output device		2		5.3.9
	DELTA DELTAP	d	Generating pulses at direct access outputs		2		5.3.10

① The number of program steps depends on the devices and types of CPU modules used.
- When using internal device or file register (R): 1
- When using direct access outputs DY: 2
- When using serial number access format file register:
  (Universal model QCPU and LCPU): 2
  (Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU): 3
- Devices other than above: 3

② This ↑ execution condition is only applied, if the annunciator (F) is used.

③ The number of program steps depends on the devices and types of CPU modules used.
- When using internal device or file register (R0 to R32767): 1
- When using direct access outputs DY or SFC program device (BL): 2
- When using serial number access format file register:
  (Universal model QCPU and LCPU): 2
  (Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU): 3
- Devices other than above: 3

④ The number of program steps depends on the devices and types of CPU modules used.
- For bit processing
- internal device (bit to be specified by bit device or word device): 1
- Direct access output: 2
- Timer, counter: 4
- For word processing
- internal device: 2
- Index register: 2
- For bit/word processing
- When using serial number access format file register:
  (Universal model QCPU and LCPU): 2
  (Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU): 3
- Devices other than above: 3

2.3.4 Shift instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Shift instruction	SFT	d	Shifting bit devices		2		5.4.1
	SFTP						

2.3.5 Master control instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Master control instruction	MC	n, d	Activating indicated program parts		2		5.5.1
	MCR	n	Deactivating indicated program parts		1		

2.3.6 Program termination instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Termination instruction	FEND	—	End of program branches		1		5.6.1
	END		End of sequence program				5.6.2

2.3.7 Miscellaneous instructions

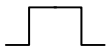
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Stop	STOP	—	Stop instruction		1		5.7.1
Other instructions	NOP	—	No operation program step		1		5.7.2
	NOPLF	—	Ignored (To change pages during printouts)				
	PAGE	n	Ignored (Subsequent programs will be controlled from step 0 of page n)				

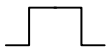
2.4 Application instructions, Part 1

2.4.1 Comparison operation instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
BIN 16-bit data comparison	LD=	s1, s2	Sets the output, if s1 = s2		3	●	6.1.1
	AND=						
	OR=						
	LD<>	s1, s2	Sets the output, if s1 ≠ s2		3	●	
	AND<>						
	OR<>						
	LD>	s1, s2	Sets the output, if s1 > s2		3	●	
	AND>						
	OR>						
	LD<=	s1, s2	Sets the output, if s1 <= s2		3	●	
	AND<=						
	OR<=						
	LD<	s1, s2	Sets the output, if s1 < s2		3	●	
	AND<						
	OR<						
	LD>=	s1, s2	Sets the output, if s1 >= s2		3	●	
	AND>=						
	OR>=						
BIN 32-bit data comparison	LDD=	s1, s2	Sets the output, if s1 = s2		①	●	6.1.2
	ANDD=						
	ORD=						
	LDD<>	s1, s2	Sets the output, if s1 ≠ s2		①	●	
	ANDD<>						
	ORD<>						
	LDD>	s1, s2	Sets the output, if s1 > s2		①	●	
	ANDD>						
	ORD>						
	LDD<=	s1, s2	Sets the output, if s1 <= s2		①	●	
	ANDD<=						
	ORD<=						
	LDD<	s1, s2	Sets the output, if s1 < s2		①	●	
	ANDD<						
	ORD<						
	LDD>=	s1, s2	Sets the output, if s1 >= s2		①	●	
	ANDD>=						
	ORD>=						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Floating point data comparison (Single precision)	LDE=	s1, s2	Sets the output, if s1 = s2		3		6.1.3
	ANDE=						
	ORE=						
	LDE<>	s1, s2	Sets the output, if s1 ≠ s2		3		
	ANDE<>						
	ORE<>						
	LDE>	s1, s2	Sets the output, if s1 > s2		3		
	ANDE>						
	ORE>						
	LDE<=	s1, s2	Sets the output, if s1 <= s2		3		
	ANDE<=						
	ORE<=						
	LDE<	s1, s2	Sets the output, if s1 < s2		3		
	ANDE<						
	ORE<						
	LDE>=	s1, s2	Sets the output, if s1 >= s2		3		
	ANDE>=						
	ORE>=						
Floating point data comparison (Double precision)	LDED=	s1, s2	Sets the output, if (s1 + 3, s1 + 2, s1 + 1, s1) = (s2 + 3, s2 + 2, s2 + 1, s2)		3		6.1.4
	ANDED=						
	ORED=						
	LDED<>	s1, s2	Sets the output, if (s1 + 3, s1 + 2, s1 + 1, s1) ≠ (s2 + 3, s2 + 2, s2 + 1, s2)		3		
	ANDED<>						
	ORED<>						
	LDED>	s1, s2	Sets the output, if (s1 + 3, s1 + 2, s1 + 1, s1) > (s2 + 3, s2 + 2, s2 + 1, s2)		3		
	ANDED>						
	ORED>						
	LDED<=	s1, s2	Sets the output, if (s1 + 3, s1 + 2, s1 + 1, s1) <= (s2 + 3, s2 + 2, s2 + 1, s2)		3		
	ANDED<=						
	ORED<=						
	LDED<	s1, s2	Sets the output, if (s1 + 3, s1 + 2, s1 + 1, s1) < (s2 + 3, s2 + 2, s2 + 1, s2)		3		
	ANDED<						
	ORED<						
	LDED>=	s1, s2	Sets the output, if (s1 + 3, s1 + 2, s1 + 1, s1) >= (s2 + 3, s2 + 2, s2 + 1, s2)		3		
	ANDED>=						
	ORED>=						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Character string data comparison	LD$=	s1, s2	Compares the character strings in s1 and s2 character by character. ② Sets the output, if s1 = s2		3		6.1.5
	AND$=						
	OR$=						
	LD$<>	s1, s2	Compares the character strings in s1 and s2 character by character. ② Sets the output, if s1 ≠ s2		3		
	AND$<>						
	OR$<>						
	LD$>	s1, s2	Compares the character strings in s1 and s2 character by character. ② Sets the output, if s1 > s2		3		
	AND$>						
	OR$>						
	LD$<=	s1, s2	Compares the character strings in s1 and s2 character by character. ② Sets the output, if s1 <= s2		3		
	AND$<=						
	OR$<=						
	LD$ <	s1, s2	Compares the character strings in s1 and s2 character by character. ② Sets the output, if s1 < s2		3		
	AND$<						
	OR$<						
	LD$>=	s1, s2	Compares the character strings in s1 and s2 character by character. ② Sets the output, if s1 >= s2		3		
	AND$>=						
	OR$>=						
BIN 16-bit block data comparison	BKCMP=	s1, s2, n, d1	It compares the nth BIN 16-bit block in s1 to the nth BIN 16-bit block in s2, beginning with the first number of device. The result of each block comparison is stored from d1 onwards.		5		6.1.6
	BKCMP<>	s1, s2, n, d1					
	BKCMP>	s1, s2, n, d1					
	BKCMP<=	s1, s2, n, d1					
	BKCMP<	s1, s2, n, d1					
	BKCMP>=	s1, s2, n, d1					
	BKCMP=P	s1, s2, n, d1					
	BKCMP<>P	s1, s2, n, d1					
	BKCMP>P	s1, s2, n, d1					
	BKCMP<=P	s1, s2, n, d1					
	BKCMP<P	s1, s2, n, d1					
	BKCMP>=P	s1, s2, n, d1					

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
BIN 32-bit block data comparisons	DBKCMP=	s1, s2, n, d1	This instruction compares BIN 32-bit data stored in n-point devices starting from the device specified by S1 with BIN 32-bit data stored in n-point devices starting from the device specified by a constant and S2, and then stores the result into the nth device specified by (D) and up.	 	5		6.1.7
	DBKCMP<>	s1, s2, n, d1					
	DBKCMP>	s1, s2, n, d1					
	DBKCMP<=	s1, s2, n, d1					
	DBKCMP<	s1, s2, n, d1					
	DBKCMP>=	s1, s2, n, d1					
	DBKCMP=P	s1, s2, n, d1					
	DBKCMP<>P	s1, s2, n, d1					
	DBKCMP>P	s1, s2, n, d1					
	DBKCMP<=P	s1, s2, n, d1					
	DBKCMP<P	s1, s2, n, d1					
	DBKCMP>=P	s1, s2, n, d1					

① The number of program steps depends on the devices used and the type of CPU.

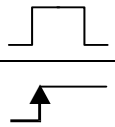
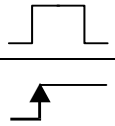
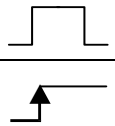
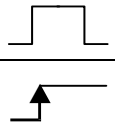
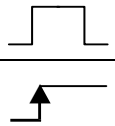
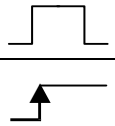
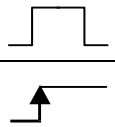
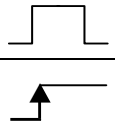
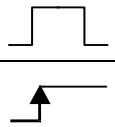
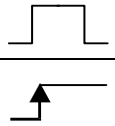
- High Performance model QCPU, Process CPU, Redundant CPU
  - – Word device: internal word devices (except for file register ZR) : 5 (NOTE 1)
  - – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 5 (NOTE 1)
  - – Constant; No limitations : 5 (NOTE 1)
  - Devices other than the above : 3 (NOTE 2)
- Basic model QCPU, Universal model QCPU, LCPU
  - – All devices that can be used : 3 (NOTE 2)

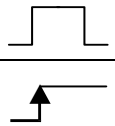
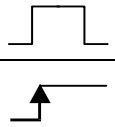
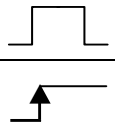
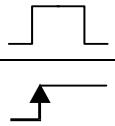
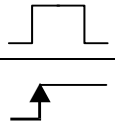
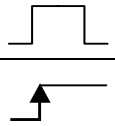
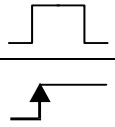
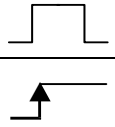
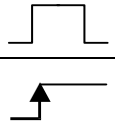
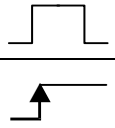
NOTE 1: For these models the number of steps increases but processing speed becomes faster.
NOTE 2: The number of steps may increase due to the conditions described in section 3.11 "Number of program steps".

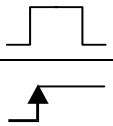
② Conditions under which the character string comparison is processed:

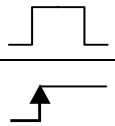
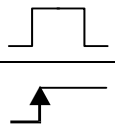
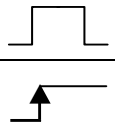
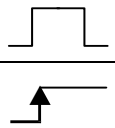
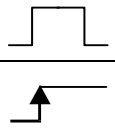
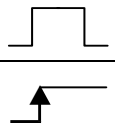
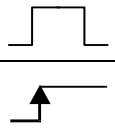
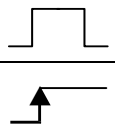
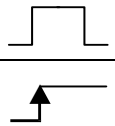
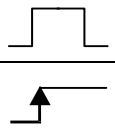
- Match : All characters in the string must match.
- Larger string : If the character strings differ, the larger string is determined.
- Smaller string : If the character strings differ, the smaller string is determined.

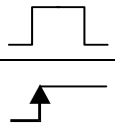
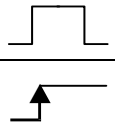
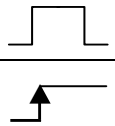
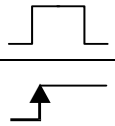
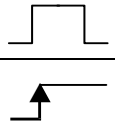
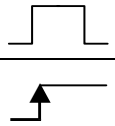
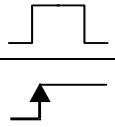
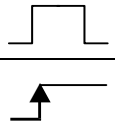
2.4.2 Arithmetic operation instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
BIN 16-bit addition and subtraction operations	+	s, d	(d)+(s) → (d)		3	●	6.2.1
	+P						
	+	s1, s2, d1	(s1)+(s2) → (d1)		4	●	
	+P						
	-	s, d	(d)-(s) → (d)		3	●	
	-P						
	-	s1, s2, d1	(s1)-(s2) → (d1)		4	●	
	-P						
BIN 32-bit addition and subtraction operations	D+	s, d	(d+1, d)+(s+1, s) → (d+1, d)		①	●	6.2.2
	D+P						
	D+	s1, s2, d1	((s1)+1, s1)+((s2) +1, s2) → ((d1)+1, d1)		②	●	
	D+P						
	D-	s, d	(d+1, d)-(s+1, s) → (d+1, d)		①	●	
	D-P						
	D-	s1, s2, d1	((s1)+1, s1)-((s2)+1, s2) → ((d1)+1, d1)		②	●	
	D-P						
BIN 16-bit multiplication and division	x	s1, s2, d1	(s1)x(s2) → ((d1)+1, d1)		③	●	6.2.3
	xP						
	/	s1, s2, d1	(s1)/(s2) → Quotient (d1), remainder ((d1)+1)		4 ④	●	
	/P						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
BIN 32-bit multiplication and division	Dx	s1, s2, d1	((s1)+1, s1)x((s2)+1, s2) → ((d1)+3, (d1)+2, (d1)+1, d1)		4 ④	●	6.2.4
	DxP						
	D/	s1, s2, d1	((s1)+1, s1)/((s2)+1, s2) → Quotient ((d1)+1, d1), remainder ((d1)+3, (d1)+2)		4 ④	●	
	D/P						
BCD 4-digit addition and substraction operations	B+	s, d	(d)+(s) → (d)		3	●	6.2.5
	B+P						
	B+	s1, s2, d1	(s1)+(s2) → (d1)		4		
	B+P						
	B-	s, d	(d)-(s) → (d)		3	●	
	B-P						
	B-	s1, s2, d1	(s1)-(s2) → (d1)		4		
	B-P						
BCD 8-digit addition and subtraction operations	DB+	s, d	(d+1, d)+(s+1, s) → (d+1, d)		3		6.2.6
	DB+P						
	DB+	s1, s2, d1	((s1)+1, s1)+((s2)+1, s2) → ((d1)+1, d1)		4		
	DB+P						
	DB-	s, d	(d+1, d)+(s+1, s) → (d+1, d)		3		
	DB-P						
	DB-	s1, s2, d1	((s1)+1, s1)+((s2)+1, s2) → ((d1)+1, d1)		4		
	DB-P						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
BCD 4-digit multiplication and division operations	B×	s1, s2, d1	(s1)x(s2) → ((d1)+1, d1)		4	●	6.2.7
	B×P						
	B/	s1, s2, d1	(s1)/(s2) → Quotient (d1), remainder ((d1)+1)		4	●	
	B/P						
BCD 8-digit multiplication and division operations	DB×	s1, s2, d1	((s1)+1, s1)x((s2)+1, s2) → ((d1)+3, (d1)+2, (d1)+1, d1)		4		6.2.8
	DB×P						
	DB/	s1, s2, d1	((s1)+1, s1)/((s2)+1, s2) → Quotient ((d1)+1, d1), remainder ((d1)+3, (d1)+2)		4	●	
	DB/P						
Floating point data addition and subtraction operations (Single precision)	E+	s, d	(d+1, d)+(s+1, s) → (d+1, d)		3	● ⑤	6.2.9
	E+P						
	E+	s1, s2, d1	((s1)+1, s1)+((s2)+1, s2) → ((d1)+1, d1)		4 ④	● ⑤	
	E+P						
	E-	s, d	(d+1, d)-(s+1, s) → (d+1, d)		3	● ⑤	
	E-P						
	E-	s1, s2, d1	((s1)+1, s1)-((s2)+1, s2) → ((d1)+1, d1)		4 ④	● ⑤	
	E-P						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Floating point data addition and subtraction operations (Double precision)	ED+	s, d	(d+3, d+2, d+1, d) +(s+3, s+2, s+1, s) → (d+3, d+2, d+1, d)		3	●	6.2.10
	ED+P						
	ED+	s1, s2, d1	((s1)+3, (s1)+2, (s1)+1, s1) +((s2)+3, (s2)+2, (s2)+1, s2) → ((d1)+3, (d1)+2, (d1)+1, d1)		4	●	
	ED+P						
	ED-	s, d	(d+3, d+2, d+1, d) – (s+3, s+2, s+1, s) → (d+3, d+2, d+1, d)		3	●	
	ED-P						
	ED-	s1, s2, d1	((s1)+3, (s1)+2, (s1)+1, s1) – ((s2)+3, (s2)+2, (s2)+1, s2) → ((d1)+3, (d1)+2, (d1)+1, d1)		4	●	
	ED-P						
Floating point data multiplication and division operations (Single precision)	Ex	s1, s2, d1	((s1)+1, s1)x((s2)+1, s2) → ((d1)+1, d1)		3	● ⑤	6.2.11
	ExP						
	E/	s1, s2, d1	((s1)+1, s1)/((s2)+1, s2) → Quotient ((d1)+1, d1)		4	● ⑤	
	E/P						
Floating point data multiplication and division operations (Double precision)	EDx	s1, s2, d1	((s1)+3, (s1)+2, (s1)+1, s1) x ((s2)+3, (s2)+2, (s2)+1, s2) → ((d1)+3, (d1)+2, (d1)+1, d1)		4	● ⑤	6.2.12
	EDxP						
	ED/	s1, s2, d1	((s1)+3, (s1)+2, (s1)+1, s1)/((s2)+3, (s2)+2, (s2)+1, s2) → Quotient ((d1)+3, (d1)+2,(d1)+1, d1)		4	● ⑤	
	ED/P						
BIN block addition and subtraction operations	BK+	s1, s2, d, n	Adds the nth 16-bit block in s1 to the nth 16-bit block in s2.		5		6.2.13
	BK+P						
	BK-	s1, s2, d, n	Subtracts the nth 16-bit block in s2 from the nth 16-bit block in s1.		5		
	BK-P						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
BIN 32-bit block addition and subtraction operations	DBK+	s1, s2, d, n	Adds the nth 32-bit block in s1 to the nth 32-bit block in s2.		5		6.2.14
	DBK+P						
	DBK-	s1, s2, d, n	Subtracts the nth 32-bit block in s2 from the nth 32-bit block in s1.		5		
	DBK-P						
Character string linking operations	$+	s, d	Character string data in s is appended to character data in d. The linked character string is stored in d.		3		6.2.15
	$+P						
	$+	s1, s2, d1	Character string data in s is appended to character data in d. The linked character string is stored in d.		4		
	$+P						
BIN increment operations	INC	d	(d)+1→ (d)		2	●	6.2.16
	INCP						
	DINC	d	(d+1, d)+1 → (d+1, d)		⑥	●	6.2.17
	DINCP						
BIN decrement operations	DEC	d	(d)-1→ (d)		2	●	6.2.16
	DECP						
	DDEC	d	(d+1, d)-1 → (d+1, d)		⑥	●	6.2.17
	DDECP						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section

① The number of program steps depends on the devices used and the type of CPU.
- High Performance model QCPU, Process CPU, Redundant CPU
  - – Word device: internal word devices (except for file register ZR) : 5 (NOTE 1)
  - – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 5 (NOTE 1)
  - – Constant; No limitations : 5 (NOTE 1)
  - Devices other than the above : 3 (NOTE 2)
- Basic model QCPU, Universal model QCPU, LCPU
  - – All devices that can be used : 3 (NOTE 2)

NOTE 1: For these models the number of steps increases but processing speed becomes faster.
NOTE 2: The number of steps may increase due to the conditions described in section 3.11 "Number of program steps".

② The number of program steps depends on the devices used and the type of CPU.
- High Performance model QCPU, Process CPU, Redundant CPU
  - – Word device: internal word devices (except for file register ZR) : 6 (NOTE 1)
  - – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 6 (NOTE 1)
  - – Constant; No limitations : 6 (NOTE 1)
  - Devices other than the above : 4 (NOTE 2)
- Basic model QCPU
  - – All devices that can be used : 4 (NOTE 2)
- Universal model QCPU, LCPU
  - – All devices that can be used : 3 (NOTE 2)

NOTE 1: For these models the number of steps increases but processing speed becomes faster.
NOTE 2: The number of steps may increase due to the conditions described in section 3.11 "Number of program steps".

③ The number of program steps depends on the devices used and the type of CPU.
- QCPU, LCPU
  - – Word device: internal word devices (except for file register ZR) : 3
  - – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 3
  - – Constant; No limitations : 3
  - Devices other than the above : 4 (NOTE 1)

NOTE 1: The number of steps may increase due to the conditions described in section 3.11 "Number of program steps".

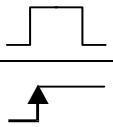
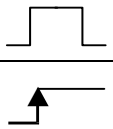
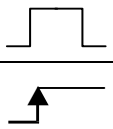
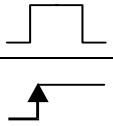
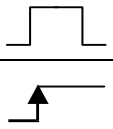
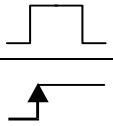
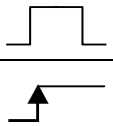
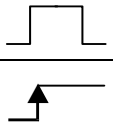
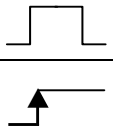
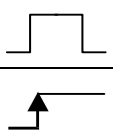
④ The number of steps is three for the Universal model QCPU and LCPU only.

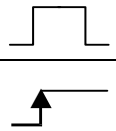
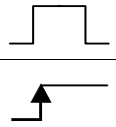
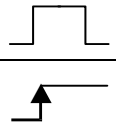
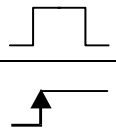
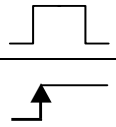
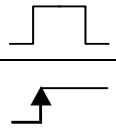
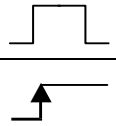
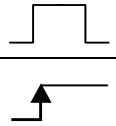
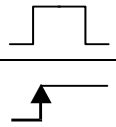
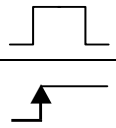
⑤ The subset is effective only with Universal model QCPU and LCPU.

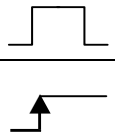
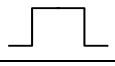
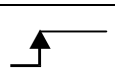
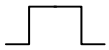
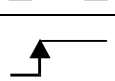
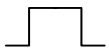
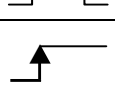
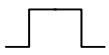
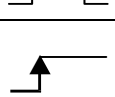
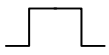
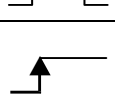
⑥ The number of program steps depends on the devices used and the type of CPU.
- High Performance model QCPU, Process CPU, Redundant CPU
  - – Word device: internal word devices (except for file register ZR) : 3 (NOTE 1)
  - – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 3 (NOTE 1)
  - – Constant; No limitations : 3 (NOTE 1)
  - Devices other than the above : 2 (NOTE 2)
- Basic model QCPU, Universal model QCPU, LCPU
  - – All devices that can be used : 2 (NOTE 2)

NOTE 1: For these models the number of steps increases but processing speed becomes faster.
NOTE 2: The number of steps may increase due to the conditions described in section 3.11 "Number of program steps".

2.4.3 Data conversion instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Conversion from BIN data into BCD data	BCD	s, d	(s) BCD conversion → (d); (s) ← BIN (0 to 9999)		3 ①	●	6.3.1
	BCDP						
	DBCD	s, d	(s+1, s) BCD conversion → (d+1, d); (s+1, s) ← BIN (0 to 99999999)		3 ①	●	
	DBCDP						
Conversion from BCD data into BIN data	BIN	s, d	(s) BIN conversion → (d); (s) ← BCD (0 to 9999)		3 ①	●	6.3.2
	BINP						
	DBIN	s, d	(s+1, s) BIN conversion → (d+1, d); (s+1, s) ← BCD (0 to 99999999)		3 ①	●	
	DBINP						
Conversion from BIN data into floating point data (Single precision)	FLT	s, d	(s+1, s) Floating point conversion → (d); (s+1, s) ← Binary value (-32768 to 32767)		3 ①	● ②	6.3.3
	FLTP						
	DFLT	s, d	(s+1, s) Floating point conversion → (d+1, d); (s+1, s) ← Binary value (-2147483648 to 2147483647)		3 ①	● ②	
	DFLTP						
Conversion from BIN data into floating point data (Double precision)	FLTD	s, d	(s) Floating point conversion → (d+3, d+2, d+1, d); (s) ← Binary value (-32768 to 32767)		4	● ②	6.3.4
	FLTPD						
	DFLTD	s, d	(s+1, s) Floating point conversion → (d+3, d+2, d+1, d); (s+1, s) ← Binary value (-2147483648 to 2147483647)		4	● ②	
	DFLTPD						
Conversion from floating point data into BIN data (Single precision)	INT	s, d	(s+1, s) BIN conversion → (d); (s+1, s) ← Floating point value (-32768 to 32767)		3 ①	● ②	6.3.5
	INTP						
	DINT	s, d	(s+1, s) BIN conversion → (d+1, d); (s+1, s) ← Floating point value (-2147483648 to 2147483647)		3 ①	● ②	
	DINTP						

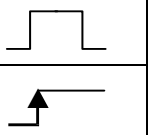
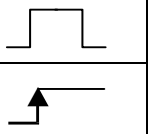
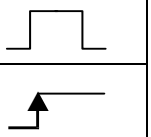
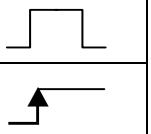
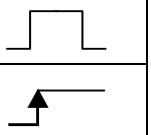
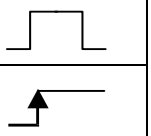
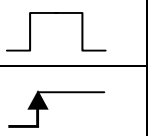
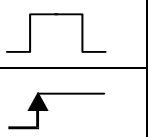
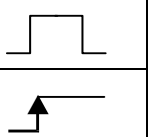
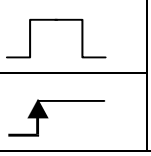
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Conversion from floating point data into BIN data (Double precision)	INTD	s, d	(s+3, s+2, s+1, s) Conversion to BIN (d) — Real number (−32768 to 32767)		3	● ②	6.3.6
	INTPD						
	DINTD	s, d	(s+3, s+2, s+1, s) Conversion to BIN (d+1,d) — Real number (–2147483648 to 2147483647)		3	● ②	
	DINTPD						
Conversion from BIN 16-bit data into BIN 32-bit data	DBL	s, d	(s) Conversion (d+1, d) — BIN (-32768 to 32767)		3		6.3.7
	DBLP						
Conversion from BIN 32-bit data into BIN 16-bit data	WORD	s, d	(s+1, s) Conversion (d) — BIN (-32768 to 32767)		3		6.3.8
	WORDP						
Conversion from BIN 16-/32-bit data into Gray code data	GRY	s, d	(s) Conversion into Gray code (d) — Binary value (-32768 to 32767)		3		6.3.9
	GRYP						
	DGRY	s, d	(s+1, s) Conversion into Gray code (d+1, d) — Binary value (-2147483648 to 2147483647)		3		
	DGRYP						
Conversion from Gray code data into BIN 16-/32-bit data	GBIN	s, d	(s) BIN conversion (d) — Gray code (-32768 to 32767)		3		6.3.10
	GBINP						
	DGBIN	s, d	(s+1, s) BIN conversion (d+1, d) — Gray code (-2147483648 to 2147483647)		3		
	DGBIN						
Sign reversal for BIN 16-/32-bit data (complement of 2)	NEG	d	(d) (d) — BIN data		2		6.3.11
	NEGP						
	DNEG	d	(d+1, d) (d+1, d) — BIN data		2		
	DNEGP						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Sign reversal for floating point data	ENEG	d	$\overline{(d+1, d)}$ → (d+1, d) ↑ Floating point value		2		6.3.12
	ENEGP						
	EDNEG	d	Floating point number $\overline{(d+3, d+2, d+1, d)}$ ⟶ (d+3, d+2, d+1, d)		3		6.3.13
	EDNEGP						
Conversion from BIN block data into BCD block data	BKBCD	s, d, n	This instruction converts each nth BIN 16-bit block in s into the nth BCD 4-digit block. Converted data is stored in d.		4		6.3.14
	BKBCDP	s, d, n					
Conversion from BCD block data into BIN block data	BKBIN	s, d, n	This instruction converts each nth BCD 4-digit block in s into the nth BIN 16-bit block. Converted data is stored in d.		4		6.3.15
	BKBINP	s, d, n					
Floating-point Conversion from single precision to double precision	ECON	s, d	32-bit floating-point real number (s+1, s) ⟶ Conversion to double precision (d+3, d+2, d+1, d)		3		6.3.16
	ECONP	s, d					
Floating-point Conversion from double precision to single precision	EDCON	s, d	64-bit floating-point real number (s+3, s+2, s+1, s) ⟶ Conversion to single precision (d+1, d)		3		6.3.17
	EDCONP	s, d					

① The number of steps is two for the Universal model QCPU and LCPU only.

② The subset is effective only with Universal model QCPU and LCPU.

2.4.4 Data transfer instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
BIN 16-bit data transfer	MOV	s, d	(s) → (d)		①	●	6.4.1
	MOVP	s, d					
BIN 32-bit data transfer	DMOV	s, d	(s+1, s) → (d+1, d)		②	●	
	DMOVP	s, d					
Floating point data transfer (Single precision)	EMOV	s, d	(s+1, s) → (d+1, d) Floating point value		②	● ①	6.4.2
	EMOVP	s, d					
Floating point data transfer (Double precision)	EDMOV	s, d	Real number data (s+3, s+2, s+1, s) → (d+3, d+2, d+1, d)		2	● ①	6.4.3
	EDMOVP	s, d					
Character string data transfer	$MOV	s, d	Transfers character string data in s to d.		3	●	6.4.4
	$MOVP	s, d					
BIN 16-bit data inversion	CML	s, d	$\overline{(s)}$ → (d)		①	●	6.4.5
	CMLP	s, d					
BIN 32-bit data inversion	DCML	s, d	$\overline{(s+1, s)}$ → (d1+1, d1)		②	●	
	DCMLP	s, d					
BIN block data transfer	BMOV	s, n, d	(s) (d) n		4	●	6.4.6
	BMOVP	s, n, d					
Identical BIN block data transfer	FMOV	s, n, d	(d) (s) n		4	●	6.4.7
	FMOVP	s, n, d					
Identical 32-bit block data transfer	DFMOV	s, n, d	(d+1, d) (s+1, s) n		4	●	6.4.8
	DFMOVP	s, n, d					

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
BIN 16-bit data exchange	XCH	d1, d2	(d1) ◄——► (d2)		3	●	6.4.9
	XCHP	d1, d2					
BIN 32-bit data exchange	DXCH	d1, d2	((d1)+1, d1) ◄——► ((d2)+1, d2)		3	●	
	DXCHP	d1, d2					
BIN block data exchange	BXCH	n, d1, d2	(d1) (d2) n		4		6.4.10
	BXCHP	n, d1, d2					
Upper and lower byte exchanges	SWAP	s	(s) b15 b8 b7 b0 8 bits 8 bits		3		6.4.11
	SWAPP	s					

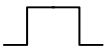
① The number of program steps depends on the devices used and the type of CPU.
- QCPU, LCPU
  – Word device: internal word devices (except for file register ZR) : 2
  – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 2
  – Constant; No limitations : 2
  Devices other than the above : 3 (NOTE 1)

NOTE 1: The number of steps may increase due to the conditions described in section 3.11 "Number of program steps".

② The number of program steps depends on the devices used and the type of CPU.
- High Performance model QCPU, Process CPU, Redundant CPU
  – Word device: internal word devices (except for file register ZR) : 3
  – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 3
  – Constant; No limitations : 3
  Devices other than the above : 3 (NOTE 1)
- Basic model QCPU
  – Word device: internal word devices (except for file register ZR) : 2
  – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 2
  – Constant; No limitations : 2
  Devices other than the above : 3 (NOTE 1)
- Universal model QCPU, LCPU
  – All devices that can be used : 2 (NOTE 1)

NOTE 1: The number of steps may increase due to the conditions described in section 3.11 "Number of program steps".

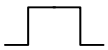
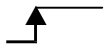
2.4.5 Program branch instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Jump instructions	CJ	p	Conditional jump (p = jump destination)		2	●	6.5.1
	SCJ	p	Conditional jump from next program scan (p = jump destination)				
	JMP	p	Jump instruction (p = jump destination)		2	●	6.5.1
	GOEND	—	Jump to the end of a program		1		6.5.2

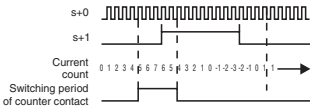
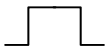
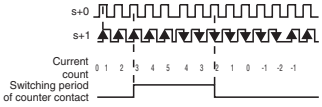
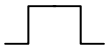
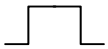
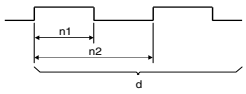
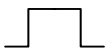
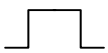
2.4.6 Interrupt program execution control instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Interrupt disabled	DI	—	Disables the execution of an interrupt program		1		6.6.1
Interrupt enabled	EI	—	Enables invoking an interrupt program		1		
Bit pattern of execution conditions of interrupt programs	IMASK	s	In the bit pattern designated by s a particular interrupt address is allocated to each bit.		2		
Return from an interrupt program to the main program	IRET	—	End of an interrupt program		1		6.6.2

2.4.7 Data refresh instructions

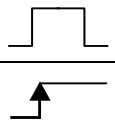
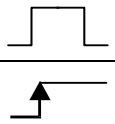
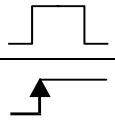
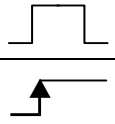
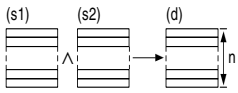
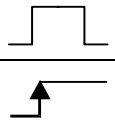
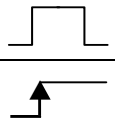
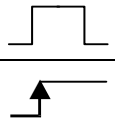
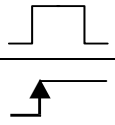
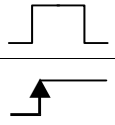
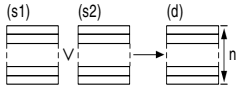
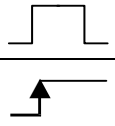
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
I/O partial refresh	RFS	s, n	The RFS instruction refreshes the inputs and outputs of the designates range of I/O devices during one program scan.		3		6.7.1
	RFSP	s, n					

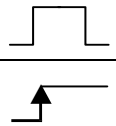
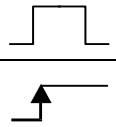
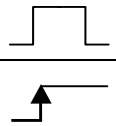
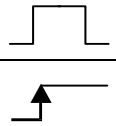
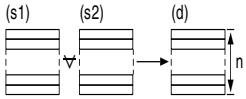
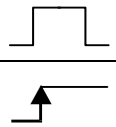
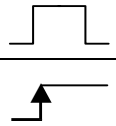
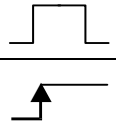
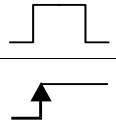
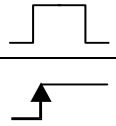
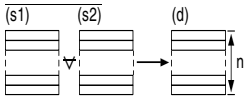
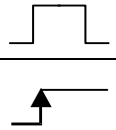
2.4.8 Other convenient instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
1-Phase Input count-up/-down Counter	UDCNT1	s, n, d			4		6.8.1
2-Phase Input count-up/-down Counter	UDCNT2	s, n, d			4		6.8.2
Programmable (teaching) Timer	TTMR	d, n	(Time, the timer is set) x n ⟶ (d) n=0:1, n=1:10, n=2:100		3		6.8.3
Special Function Timer (Timer instruction for low speed timers)	STMR	s, n, d	The STMR instruction uses outputs designated by d+0 through d+3 to perform four different timer functions: d+0: OFF delay timer output d+1: One shot timer output after OFF (Set by trailing edge) d+2: One shot timer output after ON (Set by leading edge) d+3: ON delay timer output		3		6.8.4
Positioning instruction for rotary tables	ROTC	s, n1, n2, d	The ROTC instruction rotates a sector designated by s+2 on a table with a specified number of sectors (divisions) designated by n1 to a specified position designated by s+1.		5		6.8.5
Ramp Signal	RAMP	n1, n2, n3, d1, d2	A RAMP instruction changes the content in (d1)+0 gradually from the initial value designated by n1 to the final value designated by n2.		6		6.8.6
Pulse density measurement	SPD	s, n, d	The SPD instruction counts pulses at the input designated by s for a period of time specified by n. The result of the measurement is stored in d.		4		6.8.7
Pulse output with adjustable number of pulses	PLSY	s1, s2, d	The PLSY instruction outputs a number of pulses specified by s2 at a frequency specified by s1 to an output designated by d.		4		6.8.8
Pulse width modulation	PWM	n1, n2, d			4		6.8.9
Building an input matrix	MTR	s, n, d1, d2	The MTR instruction reads the information of 16 bits beginning from the device designated by s. The number of repetitions (rows) is designated by n. The conditions of read data are stored in the device designated by d2 onwards.		5		6.8.10

2.5 Application instructions, Part 2

2.5.1 Logical operation instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Logical product	WAND	s, d	(d) ∧ (s) ⟶ (d)		3	●	7.1.1
	WANDP						
	WAND	s1, s2, d1	(s1) ∧ (s2) ⟶ (d1)		4 ①	●	
	WANDP						
	DAND	s, d	(d+1, d) ∧ (s+1, s) ⟶ (d+1, d)		②	●	
	DANDP						
	DAND	s1, s2, d	((s1)+1, s1) ∧ ((s2)+1, s2) ⟶ (d+1, d)		③	●	
	DANDP						
	BKAND	s1, s2, n, d			5		7.1.2
	BKANDP						
Logical sum	WOR	s, d	(d) ∨ (s) ⟶ (d)		3	●	7.1.3
	WORP						
	WOR	s1, s2, d1	(s1) ∨ (s2) ⟶ (d1)		4 ①	●	
	WORP						
	DOR	s, d	(d+1, d) ∨ (s+1, s) ⟶ (d+1, d)		②	●	
	DORP						
	DOR	s1, s2, d	((s1)+1, s1) ∨ ((s2)+1, s2) ⟶ (d+1, d)		③	●	
	DORP						
	BKOR	s1, s2, n, d			5		7.1.4
	BKORP						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Logical exclusive OR	WXOR	s, d	(d) ⩛ (s) ⟶ (d)		3	●	7.1.5
	WXORP						
	WXOR	s1, s2, d1	(s1) ⩛ (s2) ⟶ (d1)		4 ①	●	
	WXORP						
	DXOR	s, d	(d+1, d) ⩛ (s+1, s) ⟶ (d+1, d)		②	●	
	DXORP						
	DXOR	s1, s2, d	((s1)+1, s1) ⩛ ((s2)+1, s2) ⟶ (d+1, d)		③	●	
	DXORP						
	BKXOR	s1, s2, n, d			5		7.1.6
	BKXORP						
Logical exclusive NOR	WXNR	s, d	(d) ⩒ (s) ⟶ (d)		3	●	7.1.7
	WXNRP						
	WXNR	s1, s2, d1	$\overline{(s1) \veebar (s2)}$ ⇳ (d1)		4 ①	●	
	WXNRP						
	DXNR	s, d	$\overline{(d+1, d) \veebar (s+1, s)}$ ⇳ (d+1, d)		②	●	
	DXNRP						
	DXNR	s1, s2, d	$\overline{((s1)+1, s1) \veebar ((s2)+1, s2)}$ ⇳ (d+1, d)		③	●	
	DXNRP						
	BKXNR	s1, s2, n, d			5		7.1.8
	BKXNRP						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section

① The number of steps is three for the Universal model QCPU and LCPU only.

② The number of program steps depends on the devices used and the type of CPU.

- High Performance model QCPU, Process CPU, Redundant CPU
  - – Word device: internal word devices (except for file register ZR) : 5 (NOTE 1)
  - – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 5 (NOTE 1)
  - – Constant; No limitations : 5 (NOTE 1)
  - Devices other than the above : 3 (NOTE 2)
- Basic model QCPU, Universal model QCPU, LCPU
  - – All devices that can be used : 3 (NOTE 2)

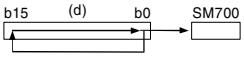
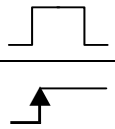
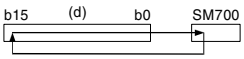
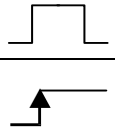
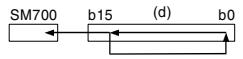
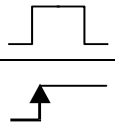
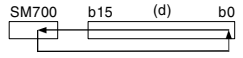
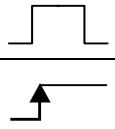
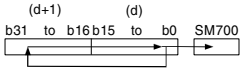
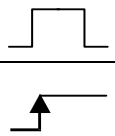
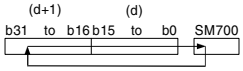
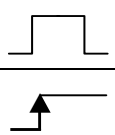
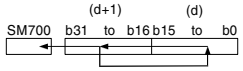
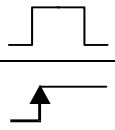
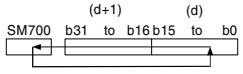
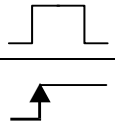
NOTE 1: For these models the number of steps increases but processing speed becomes faster.
NOTE 2: The number of steps may increase due to the conditions described in section 3.11 "Number of program steps".

③ The number of program steps depends on the devices used and the type of CPU.

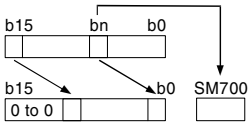
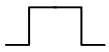
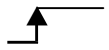
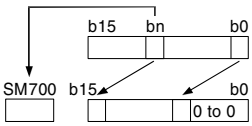
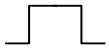
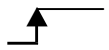
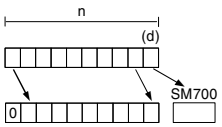
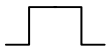
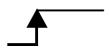
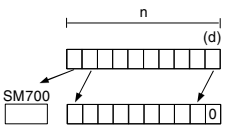
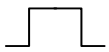
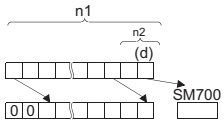
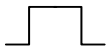
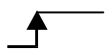
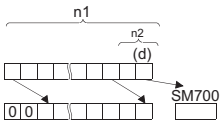
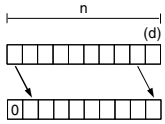
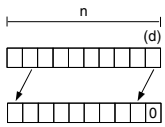
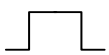
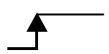
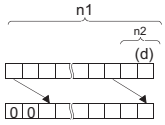
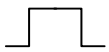
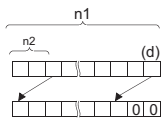
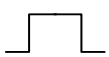
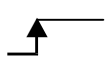
- High Performance model QCPU, Process CPU, Redundant CPU
  - – Word device: internal word devices (except for file register ZR) : 6 (NOTE 1)
  - – Bit device: whose device numbers are multiplies of 16, whose digit designation is K8, and which use no index qualification : 6 (NOTE 1)
  - – Constant; No limitations : 6 (NOTE 1)
  - Devices other than the above : 4 (NOTE 2)
- Basic model QCPU
  - – All devices that can be used : 4 (NOTE 2)
- Universal model QCPU, LCPU
  - – All devices that can be used : 3 (NOTE 2)

NOTE 1: For these models the number of steps increases but processing speed becomes faster.
NOTE 2: The number of steps may increase due to the conditions described in section 3.11 "Number of program steps".

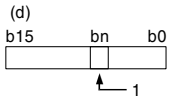
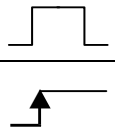
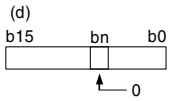
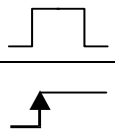
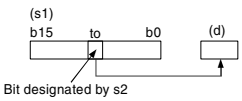
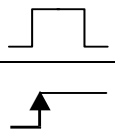
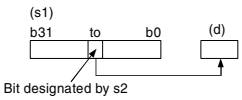
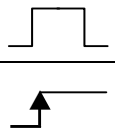
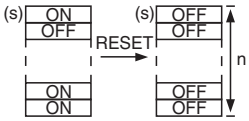
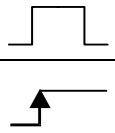
2.5.2 Rotation instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Data rotation to the right (16-bit)	ROR	n, d	 rotates by n bits to the right		3	●	7.2.1
	RORP						
	RCR	n, d	 rotates by n bits to the right		3	●	
	RCRP						
Data rotation to the left (16-bit)	ROL	n, d	 rotates by n bits to the left		3	●	7.2.2
	ROLP						
	RCL	n, d	 rotates by n bits to the left		3	●	
	RCLP						
Data rotation to the right (32-bit)	DROR	n, d	 rotates by n bits to the right		3	●	7.2.3
	DRORP						
	DRCR	n, d	 rotates by n bits to the right		3	●	
	DRCRP						
Data rotation to the left (32-bit)	DROL	n, d	 rotates by n bits to the left		3	●	7.2.4
	DROLP						
	DRCL	n, d	 rotates by n bits to the left		3	●	
	DRCLP						

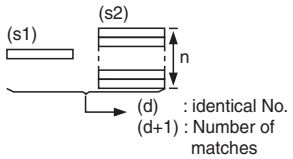
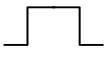
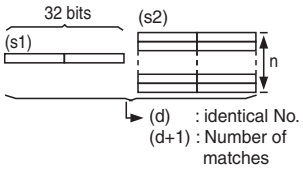
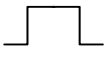
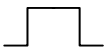
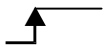
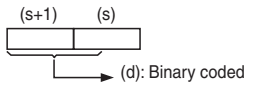
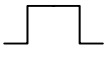
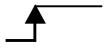
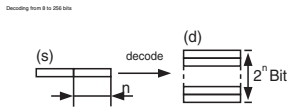
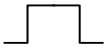
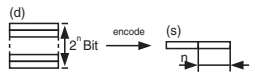
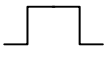
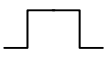
2.5.3 Shift instructions

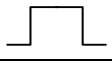
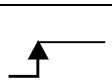
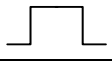
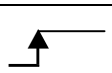
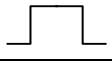
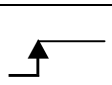
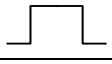
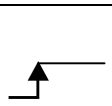
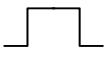
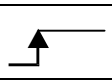
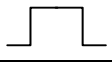
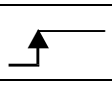
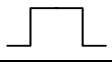
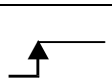
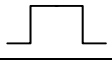
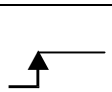
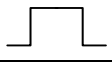
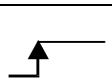
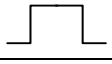
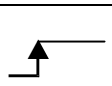
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Shift a 16-bit data word by n bits	SFR	n, d			3	●	7.3.1
	SFRP						
	SFL	n, d			3	●	
	SFLP						
Shift n bit devices by 1 bit	BSFR	n, d			3		7.3.2
	BSFRP						
	BSFL	n, d			3		
	BSFLP						
Shift n bit devices by n bits	SFTBR	n1, n2, d			4		7.3.3
	SFTBRP						
	SFTBL	n1, n2, d			4		
	SFTBLP						
Shift n word devices by one digit	DSFR	n, d			3	●	7.3.4
	DSFRP						
	DSFL	n, d			3	●	
	DSFLP						
Shift n word devices by n words	SFTWR	n1, n2, d			4		7.3.5
	SFTWRP						
	SFTWL	n1, n2, d			4		
	SFTWLP						

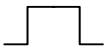
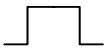
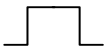
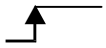
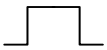
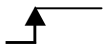
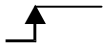
2.5.4 Bit processing instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Set / reset single bits	BSET	n, d			3	●	7.4.1
	BSETP						
	BRST	n, d			3	●	
	BRSTP						
Test condition of single bits in 16-/32-bit data words	TEST	s1, s2, d			4		7.4.2
	TESTP						
	DTEST	s1, s2, d			4		
	DTESTP						
Reset sections of bits in a batch	BKRST	s, n			3		7.4.3
	BKRSTP						

2.5.5 Data processing instructions

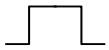
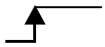
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Search 16-bit data	SER	s1, s2, d, n			5		7.5.1
	SERP						
	DSER	s1, s2, d, n			5		
	DSERP						
Check data bits (16-/32-bit)	SUM	s, d			3	●	7.5.2
	SUMP						
	DSUM	s, d			3	●	
	DSUMP						
Decoding data	DECO	s, d, n			4		7.5.3
	DECOP						
Encoding data	ENCO	s, d, n	Encoding from 256 to 8 bits 		4		7.5.4
	ENCOP						
7-segment decoding	SEG	s, d			3	●	7.5.5
	SEGP						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Disunite/unite 16-bit data words	DIS	s, n, d	The DIS instruction disunites a 16-bit data value to groupings of 4 bits. The data value to be disunited in s, the number of 4-bit groupings in n, and the first number of destination device in d must be specified.		4		7.5.6
	DISP						
	UNI	s, n, d	The UNI instruction separates each 4 lowest bits of up to four 16-bit data values and unites their conditions in one 16-bit data value.		4		7.5.7
	UNIP						
	NDIS	s1, s2, d	The NDIS instruction disunites data in devices specified from s1 on to bit groupings with a number of bits specified by s2. The disunited bit groupings are stored separately in the device specified by d onwards.		4		7.5.8
	NDISP						
	NUNI	s1, s2, d	The NUNI instruction separates bit groupings of a size specified by s2 from devices specified by s1 and unites these bit groupings in one data value. The bit groupings are stored successively from the device specified by d onwards.				
	NUNIP						
	WTOB	s, n, d	For this instruction the data values in s to be disunited, the number of byte units in n, and the first number of destination device in d must be specified.		4		7.5.9
	WTOBP						
	BTOW	s, n, d	The initial number of data value in s to be united, the number of byte units n, and destination device in d must be specified.				
	BTOWP						
Search maximum values in 16-/ 32-bit data	MAX	s, n, d	The MAX instruction searches for maximum values in 16-bit data blocks. The number of data blocks to be searched through is specified by n. The greatest value found in s through s+(n-1) is stored in d.		4		7.5.10
	MAXP						
	DMAX	s, n, d	The DMAX instruction searches for maximum values in 32-bit data blocks. The number of data blocks to be searched through is specified by n. The greatest value found in s through s+(n-1) is stored in d.				
	DMAXP						
Searching minimum values in 16-/ 32-bit data	MIN	s, n, d	The MIN instruction searches for minimum values in 16-bit data blocks. The number of data blocks to be searched through is specified by n. The smallest value found in s through s+(n-1) is stored in d.		4		7.5.11
	MINP						
	DMIN	s, n, d	The DMIN instruction searches for minimum values in 32-bit data blocks. The number of data blocks to be searched through is specified by n. The smallest value found in s through s+(n-1) is stored in d.				
	DMINP						

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Sorting 16-/ 32-bit data	SORT	s1, n, s2, d1, d2	The SORT instruction sorts 16-bit data specified by s1 in ascending or descending order. The number of data to be sorted is specified by n.		6		7.5.12
	DSORT		The DSORT instruction sorts 32-data specified by s1 in ascending or descending order. The number of data to be sorted is specified by n.				
Calculating totals of 16-/ 32-bit BIN data blocks	WSUM	s, n, d	The WSUM instruction calculates the total of 16-bit data blocks in the device specified by s. The result is stored in the device specified by d and d+1.		4		7.5.13
	WSUMP						
	DWSUM	s, n, d	The DWSUM instruction calculates the total of 32-bit data blocks in the device specified by s and s+1. The result is stored in d through d+3.		4		7.5.14
	DWSUMP						
Calculation of averages	MEAN	s, n, d	Calculates the mean of n-point devices (in 16-bit units) starting from the device specified by (s), and then stores the result into the device specified by (d).		4		7.5.15
	MEANP						
	DMEAN	s, n, d	Calculates the mean of n-point devices (in 32-bit units) starting from the device specified by (s), and then stores the result into the device specified by (d).		4		
	DMEANP						

2.5.6 Structured program instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Repetition instructions	FOR	n	The FOR/NEXT loop repeats single program sequences without setting an input condition. The program sequence located between the FOR and the NEXT command is repeated for n times.		2		7.6.1
	NEXT				1		
	BREAK	p, d	The BREAK instruction terminates a FOR/NEXT loop execution and jumps to the pointer specified by p.		3		7.6.2
	BREAKP						
Subroutine program calls	CALL	p p, s1 to sn	The CALL instruction calls a subroutine program specified by a pointer p. s1 to sn are arguments sent to subroutine program (n <= 5).		2+n ①	●③	7.6.3
	CALLP						
	RET		The RET instruction marks the end of a subroutine program.		1		7.6.4
	FCALL	p p, s1 to sn	On resetting the execution condition for the FCALL instruction, the contacts and coils in the subroutine program specified in p (pointer/ label) are treated as if the execution condition of the according instruction was not set. s1 to sn are arguments sent to subroutine program (n <= 5).		2+n ①		7.6.5
	FCALLP						
Subroutine program calls between program files	ECALL	file name, p file name, p, s1 to sn	The ECALL instruction calls a subroutine program specified by a pointer address (label) in a program file specified by a file name. s1 to sn are arguments sent to subroutine program (n <= 5).		2+n ②		7.6.6
	ECALLP						
Subroutine program calls between program files	EFCALL	file name, p	On resetting the execution condition for the EFCALL instruction, the contacts and coils in the subroutine program specified in p (pointer/ label) are treated as if the execution condition of the according instruction was not set.		3+n ②		7.6.7
	EFCALLP						
	XCALL	p, s1 to sn	• Executes subroutine program p when input condition is met. • Performs non-execution processing of subroutine program p if input conditions have not been met. (s1 to sn are arguments sent to subroutine program. n <= 5)		2+n ①		7.6.8

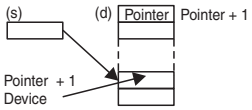
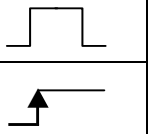
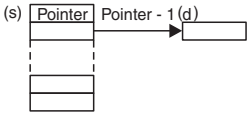
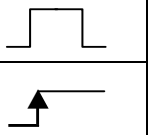
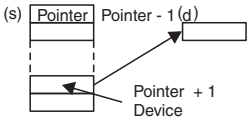
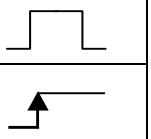
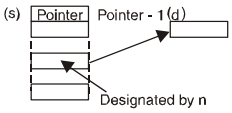
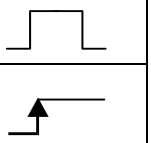
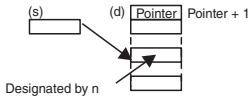
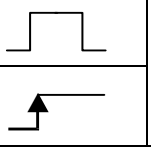
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Select refresh	COM		Performs auto refresh of intelligent function modules, link refresh, auto refresh of CPU shared memory, and communications with peripherals.		1		7.6.9
	CCOM		Performs auto refresh of intelligent function modules, auto refresh of CPU shared memory, and communications with peripherals after the input conditions are met.		1		7.6.11
	CCOMP				1		7.6.11
Index qualification of entire ladders	IX	s	The IX and IXEND instructions perform index qualification on those devices in the program part located between the IX and IXEND instructions.		2		7.6.12
	IXEND				1		
Designation of qualification values in index qualification of entire ladders	IXDEV		The IXDEV and IXSET instructions read the addresses of the devices in the offset designation area and write these offset numbers to an index table in the device designated by d.		1		7.6.13
	IXSET	p, d			3		

① n indicates number of arguments for subroutine program.

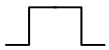
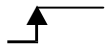
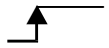
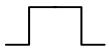
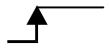
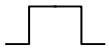
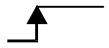
② n indicates the total of the number of arguments used in the subroutine program and the number of program name steps. The number of program name steps is calculated as "number of characters in the program/2" (decimal fraction is rounded up).

③ The subset is effective only with the Universal model QCPU and LCPU.

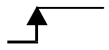
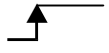
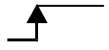
2.5.7 Data table operation instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Write data to a data table	FIFW FIFWP	s, d			3		7.7.1
Read data entered first from data table	FIFR FIFRP	s, d			3		7.7.2
Read data entered last from data table	FPOP FPOPP	s, d			3		7.7.3
Delete specified data blocks from data table	FDEL FDELP	s, n, d			4		7.7.4
Insert specified data blocks in data table	FINS FINSP						

2.5.8 Buffer memory access instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Reading data from a special function module	FROM	n1, n2, n3, d	The FROM instruction reads 1-word data (16-bit) from the buffer memory of a special function module.		5		7.8.1
	FROMP						
	DFRO		The DFRO instruction reads 2-word data (32-bit) from the buffer memory of a special function module.				
	DFROP						
Writing data to a special function module	TO	s, n1, n2, n3	The TO instruction writes 1-word data (16-bit) from the memory of the CPU to the buffer memory of a special function module.		5		7.8.2
	TOP						
	DTO		The DTO instruction writes 2-word data (32-bit) from the memory of the CPU to the buffer memory of a special function module.				
	DTOP						

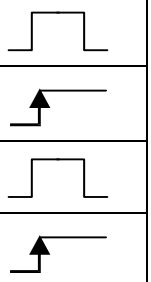
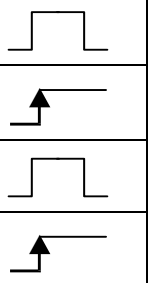
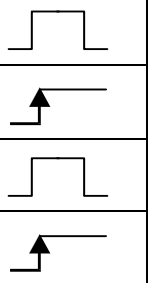
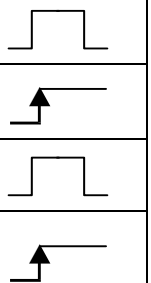
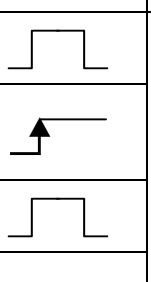
2.5.9 Display instructions

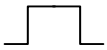
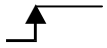
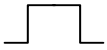
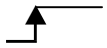
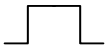
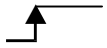
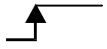
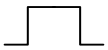
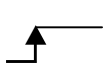
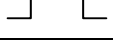
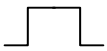
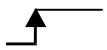
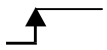
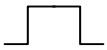
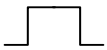
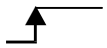
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
ASCII character output	PR	s, d	SM701 set (1): Output of an ASCII character string of 16 characters to an output module. The character string, divided into twice 8 characters, is read from the address area s and output to the outputs specified by d. SM701 not set (0): Output of ASCII character string data up to the character code "00_H" in hexadecimal format from the address area s to the outputs specified by d.		3		7.9.1
	PRC	s, d	The PRC instruction outputs a comment of a device (in ASCII code) to an output module. If SM701 is set (1), 16 characters are output; if SM701 is not set (0), 32 characters are output.		3		7.9.2
Clear display	LEDR		Resetting annunciators and error displays		1		7.9.3

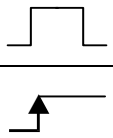
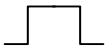
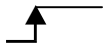
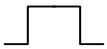
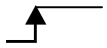
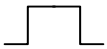
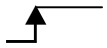
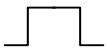
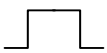
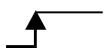
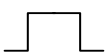
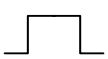
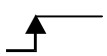
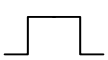
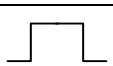
2.5.10 Debugging and failure diagnosis instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Failure check	CHKST		The CHKST instruction starts the execution of the CHK instruction. If the execution condition for the CHKST instruction is not set (0), the program step following the CHK instruction will be executed.		1		7.10.1
	CHK		The CHK instruction supports failure check operations for contact circuits. Once an error occurs within such a circuit, the device in d1 is set and the corresponding error code is stored in d2.				
	CHKCIR		The CHKCIR instruction generates error check circuits for the CHK instruction and starts the program section with the generated error check circuits.		1		7.10.2
	CHKEND		End instructions for a program part with generated check circuits.				

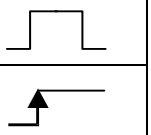
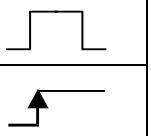
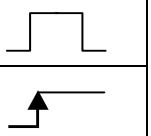
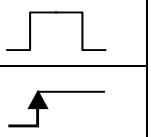
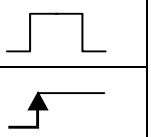
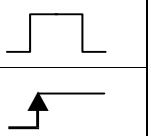
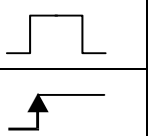
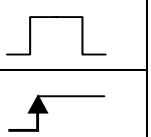
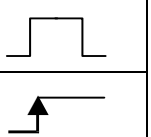
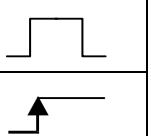
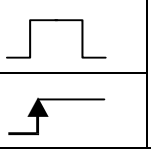
2.5.11 Character string processing instructions

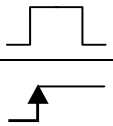
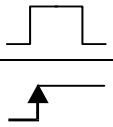
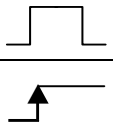
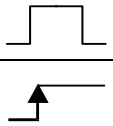
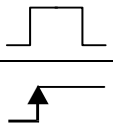
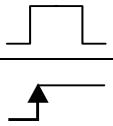
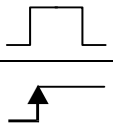
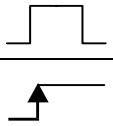
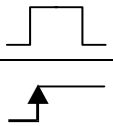
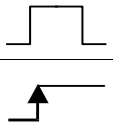
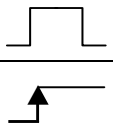
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Conversion of 16-/32-bit binary data into decimal values in ASCII code	BINDA	s, d	The BINDA instruction converts a 16-bit binary value specified by s into a 5-digit decimal value in ASCII code and stores it in the device specified in d.		3		7.11.1
	BINDAP						
	DBINDA		The DBINDA instruction converts 32-bit binary data specified by s into a 10-digit decimal value in ASCII code and stores it in the device specified in d.				
	DBINDAP						
Conversion of 16-/32-bit binary data into hexadecimal values in ASCII code	BINHA	s, d	The BINHA instruction converts 16-bit binary data specified by s into a 4-digit hexadecimal value in ASCII code and stores it in the devices specified by d.		3		7.11.2
	BINHAP						
	DBINHA		The DBINHA instruction converts 32-bit binary data specified by s into a 8-digit hexadecimal value in ASCII code and stores it in the devices specified by d.				
	DBINHAP						
Conversion of 4-/8-digit BCD data into ASCII code	BCDDA	s, d	The BCDDA instruction converts 4-digit BCD data specified by s into the ASCII format and stores it in the devices specified by d.		3		7.11.3
	BCDDAP						
	DBCDDA		The DBCDDA instruction converts 8-digit BCD data specified by s into the ASCII format and stores it in the devices specified by d.				
	DBCDDAP						
Conversion of decimal ASCII data into BIN 16-/32-bit binary data	DABIN	s, d	The DABIN instruction converts the 5-digit decimal ASCII data specified by s into the BIN 16-bit format and stores it in the devices specified by d.		3		7.11.4
	DABINP						
	DDABIN		The DDABIN instruction converts the 10-digit decimal ASCII data specified by s into the BIN 32-bit format and stores it in the devices specified by d.				
	DDABINP						
Conversion of hexadecimal ASCII data into 16-/32-bit binary data	HABIN	s, d	The HABIN instruction converts the 4-digit hexadecimal ASCII data in the device specified by s into the BIN 16-bit binary format and stores it in the devices specified by d.		3		7.11.5
	HABINP						
	DHABIN		The DHABIN instruction converts the 8-digit hexadecimal ASCII data specified in the area s into the BIN 32-bit format and stores it in the devices specified by d.				
	DHABINP						

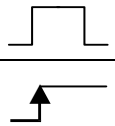
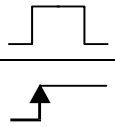
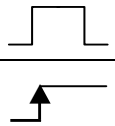
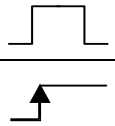
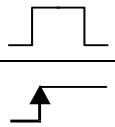
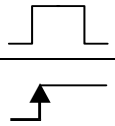
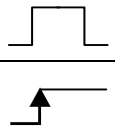
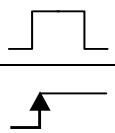
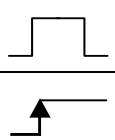
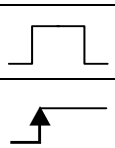
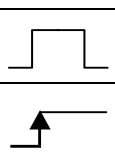
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Conversion of decimal ASCII data into 4-/8-digit BCD data	DABCD	s, d	The DABCD instruction converts the decimal ASCII data in s into the 4-digit BCD data format and stores it in the devices specified by d.		3		7.11.6
	DABCDP						
	DDABCD		The DDABCD instruction converts the decimal ASCII data specified by s into the 8-digit BCD format and stores it in the devices specified in d.				
	DDABCDP						
Read-out of comment data	COMRD	s, d	The COMRD instruction reads comment data from the device specified by s and stores it as ASCII code in the area d.		3		7.11.7
	COMRDP						
Detection of character string length	LEN	s, d	The length instruction detects the length of a character string specified in s and stores the result in the device specified by d.		3		7.11.8
	LENP						
Conversion of BIN 16-/32-bit binary data into character string data	STR	s1, s2, d	Adds a decimal point to the BIN 16-bit binary value in the device specified by s2 to the digit specified by s1, converts the data into a character string, and stores it in the area of the devices specified by d.		4		7.11.9
	STRP						
	DSTR		Adds a decimal point to the BIN 32-bit binary value in the device specified by s2 to the digit specified by the device s1, converts the data into a character string, and stores it in the area of the devices specified by d.				
	DSTRP						
Conversion of character string data into BIN 16-/32-bit binary data	VAL	s, d1, d2	Converts the character strings stored in the area s into BIN 16-bit data. The number of digits and the binary value are stored in d1 and d2.		4		7.11.10
	VALP						
	DVAL		Converts the character strings stored in s into BIN 32-bit data. The number of digits and the binary value are stored in d1 and d2.				
	DVALP						
Conversion of floating point data into character string data	ESTR	s1, s2, d	Converts the floating point data in s1 into character string data. The data format of the character string is specified in s2. The result is stored in d.		4		7.11.11
	ESTRP						
Conversion of character string data into decimal floating point data	EVAL	s, d	Converts the character string in s into a decimal floating point number (real number). The result is stored in d.		3		7.11.12
	EVALP						
Conversion of 16-bit data into ASCII code	ASC	s, n, d	Converts the 16-bit binary data stored from s onwards into the hexadecimal ASCII format and stores the result considering the number of characters specified by n from d onwards.		4		7.11.13
	ASCP						

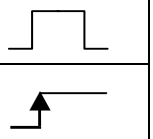
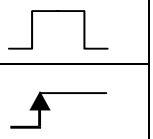
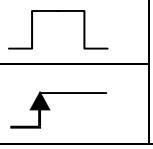
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Conversion of hexadecimal ASCII values into binary values	HEX	s, n, d	Converts the hexadecimal ASCII characters from s onwards into binary values. The number of characters to be converted is specified by n. The result is stored from d onwards.		4		7.11.14
	HEXP						
Extraction of character string data (right part of character string)	RIGHT	s, n, d	Stores n characters from the right side of the character string (end of character string) in s. The characters are stored in d.		4		7.11.15
	RIGHTP						
Extraction of character string data (left part of character string)	LEFT	s, n, d	Stores n characters from the left side of the character string (beginning of character string) in s. The characters are stored in d.				
	LEFTP						
Selecting and moving parts of character strings into a character string	MIDR	s1, s2, d	Stores a specified part of the character string stored in s. The first character of the part to be stored is specified in s2.		4		7.11.16
	MIDRP						
	MIDW	s1, s2, d	Stores a part of specified length of the character string stored in s1 in the area specified in d. The first address of the storage area in d is specified in s2.				
	MIDWP						
Search for character strings	INSTR	s1, s2, n, d	Searches the character string specified in s1 within the character string data specified by s2. The search begins with the character specified in n.		5		7.11.17
	INSTRP						
Insert character strings	STRINS	s, n, d	Inserts the character string data specified by (S) to the (n)th character (insert position) from the initial character string data specified by (D).		4		7.11.18
	STRINSP						
Delete character strings	STRDEL	d n1, n2	Deletes the (n2) characters data specified by (d) starting from the device(insert position) specified by n1.		4		7.11.19
	STRDELP						
Floating point data conversion with BCD representation	EMOD	s1, s2, d1	Calculates the BCD format from the floating point number in s1 considering the decimal point shift to the right specified in s2. The result is stored in d1.		4		7.11.20
	EMODP						
BCD data conversion with decimal floating point format	EREXP	s1, s2, d1	Calculates the decimal format of the floating point data from the floating point data in BCD format in s1, considering the decimal places specified in s2. The result is stored in d1.		3		7.11.21
	EREXPP						

2.5.12 Special function instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Sine calculation (Floating point single precision)	SIN SINP	s, d	SIN(s+1, s) ⟶ (d+1, d)		3		7.12.1
Cosine calculation (Floating point single precision)	COS COSP	s, d	COS(s+1, s) ⟶ (d+1, d)		3		7.12.3
Tangent calculation (Floating point single precision)	TAN TANP	s, d	TAN(s+1, s) ⟶ (d+1, d)		3		7.12.5
Arcus sine calculation (Floating point single precision)	ASIN ASINP	s, d	ASIN(s+1, s) ⟶ (d+1, d)		3		7.12.7
Arcus cosine calculation (Floating point single precision)	ACOS ACOSP	s, d	ACOS(s+1, s) ⟶ (d+1, d)		3		7.12.9
Arcus tangent calculation (Floating point single precision)	ATAN ATANP	s, d	ATAN(s+1, s) ⟶ (d+1, d)		3		7.12.11
Sine calculation (Floating point double precision)	SIND SINDP	s, d	SIN(s+3, s+2, s+1, s) ⟶ (d+3, d+2, d+1, d)		3		7.12.2
Cosine calculation (Floating point double precision)	COSD COSDP	s, d	COS(s+3, s+2, s+1, s) ⟶ (d+3, d+2, d+1, d)		3		7.12.4
Tangent calculation (Floating point double precision)	TAND TANDP	s, d	TAN(s+3, s+2, s+1, s) ⟶ (d+3, d+2, d+1, d)		3		7.12.6
Arcus sine calculation (Floating point double precision)	ASIND ASINDP	s, d	ASIN(s+3, s+2, s+1, s) ⟶ (d+3, d+2, d+1, d)		3		7.12.8
Arcus cosine calculation (Floating point double precision)	ACOSD ACOSDP	s, d	ACOS(s+3, s+2, s+1, s) ⟶ (d+3, d+2, d+1, d)		3		7.12.10

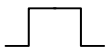
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Arcus tangent calculation (Floating point double precision)	ATAND ATANDP	s, d	ATAN(s+3, s+2, s+1, s) $\longrightarrow$ (d+3, d+2, d+1, d)		3		7.12.12
Conversion from degrees into radian	RAD RADP	s, d	(s+1, s) $\longrightarrow$ (d+1, d) Conversion from degrees into radian		3		7.12.13
	RADD RADDP	s, d	(s+3, s+2, s+1, s) $\longrightarrow$ (d+3, d+2, d+1, d) Conversion from degrees into radian		3		7.12.14
Conversion from radian into degree	DEG DEGP	s, d	(s+1, s) $\longrightarrow$ (d+1, d) Conversion from radian into degree		3		7.12.15
	DEGD DEGDP	s, d	(s+3, s+2, s+1, s) $\longrightarrow$ (d+3, d+2, d+1, d) Conversion from radian into degree		3		7.12.16
Exponentiation	POW POWP	s1, s2, d	$(s1+1, s1)^{(s2+1, s2)}$ $\longrightarrow$ (d+1, d)		4		7.12.17
	POWD POWDP	s1, s2, d	$(s1+3, s1+2, s1+1, s1)^{(s2+3, s2+2, s2+1, s2)}$ $\longrightarrow$ (d+3, d+2, d+1, d)		4		7.12.18
Square root calculation	SQR SQRP	s, d	$\sqrt{(s+1, s)}$ $\longrightarrow$ (d+1, d)		3		7.12.19
	SQRD SQRDP	s, d	$\sqrt{(s+3, s+2, s+1, s)}$ $\longrightarrow$ (d+3, d+2, d+1, d)		3		7.12.20
Floating point value as exponent of e	EXP EXPP	s, d	$e^{(s+1, s)}$ $\longrightarrow$ (d+1, d)		3		7.12.21
	EXPD EXPDP	s, d	$e^{(s+3, s+2, s+1, s)}$ $\longrightarrow$ (d+3, d+2, d+1, d)		3		7.12.22

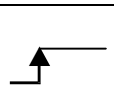
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Logarithm (natural) calculation	LOG	s, d	LOG e(s+1, s) ⟶ (d+1, d)		3		7.12.23
	LOGP						
	LOGD	s, d	LOG e(s+3, s+2, s+1, s) ⟶ (d+3, d+2, d+1, d)		3		7.12.24
	LOGDP						
Common logarithm	LOG10	s, d	log10 (s+1, s) ⟶ (d+1, d)		3		7.12.25
	LOG10P						
	LOG10D	s, d	log10 (s+3, s+2, s+1, s) ⟶ (d+3, d+2, d+1, d)		3		7.12.26
	LOG10DP						
Randomize value	RND	d	Stores the generated random value in d.		2		7.12.27
	RNDP						
Update random values	SRND	s	Updates the series of random values stored in s.		2		
	SRNDP						
Square root calculation from 4-digit BCD data	BSQR	s, d	$\sqrt{(s)}$ → (d) +0 Integer; +1 Decimal place		3		7.12.28
	BSQRP						
Square root calculation from 8-digit BCD data	BDSQR	s, d	$\sqrt{(s+1, s)}$ → (d) +0 Integer; +1 Decimal place		3		
	BDSQRP						
Sine calculation from BCD data	BSIN	s, d	sin (s) → (d) +0 Sign character; +1 Integer; +2 Decimal place		3		7.12.29
	BSINP						
Cosine calculation from BCD data	BCOS	s, d	cos (s) → (d) +0 Sign character; +1 Integer; +2 Decimal place		3		7.12.30
	BCOSP						
Tangent calculation from BCD data	BTAN	s, d	tan (s) → (d) +0 Sign character; +1 Integer; +2 Decimal place		3		7.12.31
	BTANP						

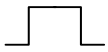
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Arcus sine calculation from BCD data	BASIN BASINP	s, d	$\sin^{-1}(s) \rightarrow$ (d) +0 Character sign; +1 Integer; +2 Decimal place		3		7.12.32
Arcus cosine calculation from BCD data	BACOS BACOSP	s, d	$\cos^{-1}(s) \rightarrow$ (d) +0 Character sign; +1 Integer; +2 Decimal place		3		7.12.33
Arcus tangent calculation from BCD data	BATAN BATANP	s, d	$\tan^{-1}(s) \rightarrow$ (d) +0 Character sign; +1 Integer; +2 Decimal place		3		7.12.34

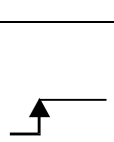
2.5.13 Data control instructions

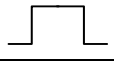
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section

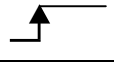
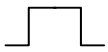
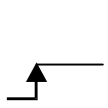
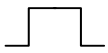
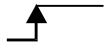
Upper and lower limit controls for BIN 16-/32-bit data	LIMIT	s1, s2, s3, d	If (s3)<(s1)
			the data value in s1 is stored in d.
			If (s1)≤(s3)≤(s2)
			the data value in s3 is stored in d.
			If (s2)<(s3)
			the data value in s2 is stored in d.
					5		7.13.1

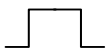
	LIMITP						

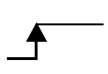
	DLIMIT	s1, s2, s3, d	If
			((s3)+1, s3)<((s1)+1, s1)
			the data value in ((s1)+1, s1) is stored in (d+1, d).
			If ((s1)+1, s1)≤
			((s3)+1, s3) <((s2)+1,s2)
			the data value in ((s3)+1, s3) is stored in (d+1, d).
			If ((s2)+1, s2)<
			((s3)+1, s3)<((s2)+1, s2)
			the data value in ((s2)+1, s2) is stored in (d+1, d).
							

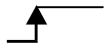
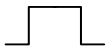
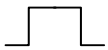
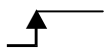
	DLIMITP						

Dead band controls for BIN 16-/32-bit data	BAND	s1, s2, s3, d	If (s1)≤(s3)≤(s2)
			0 → (d)
			If (s3)<(s1)
			(s3)–(s1)→ (d)
			If (s2)<(s3)
			(s3)–(s2)→ (d)
					5		7.13.2

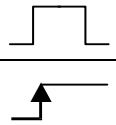
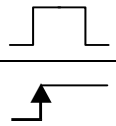
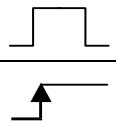
	BANDP						
	DBAND	s1, s2, s3, d	If ((s1)+1, s1)≤((s3)+1, s3) ≤((s2)+1, s2) 0 → (d+1, d) If ((s3)+1, s3)<(s1+1, s1) ((s3)+1, s3) – ((s1)+1, s1) →(d+1, d) If ((s2)+1, s2)<((s3)+1, s3) ((s3)+1, s3) – ((s2)+1, s2) →(d+1, d)				
	DBANDP						
Zone control for BIN 16-/32-bit data	ZONE	s1, s2, s3, d	If s3 = 0: 0 → (d) If s3 > 0: s3 + s2 → (d) If s3 < 0: s3 – s1 → (d)		5		7.13.3
	ZONEP						

	DZONE	s1, s2, s3, d	If ((s3)+1, s3) = 0
			0 → (d+1, d)
			If ((s3)+1, s3) > 0
			((s3)+1, s3) + ((s2)+1, s2)
			→(d+1, d)
			If ((s3)+1, s3) < 0
			((s3)+1, s3) + ((s1)+1, s1)
			→(d+1, d)
							

	DZONEP						

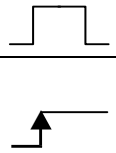
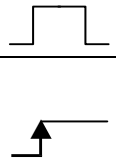
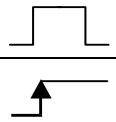
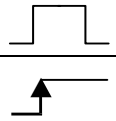
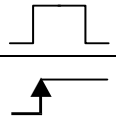
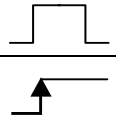
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Point-by point coordinate data	SCL	s1, s2, d	Executes scaling for the scaling conversion data (16-bit data units) specified by (s2) with the input value specified by (s1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (s2) and up.		4		7.13.4
	SCLP						
	DSCL	s1, s2, d	Executes scaling for the scaling conversion data (32-bit data units) specified by (s2) with the input value specified by (s1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (s2) and up.		4		
	DSCLP						
X or Y coordinate data	SCL2	s1, s2, d	Executes scaling for the scaling conversion data (16-bit data units) specified by (s2) with the input value specified by (s1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (s2) and up.		4		7.13.5
	SCL2P						
	DSCL2	s1, s2, d	Executes scaling for the scaling conversion data (32-bit data units) specified by (s2) with the input value specified by (s1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (s2) and up.		4		
	DSCL2P						

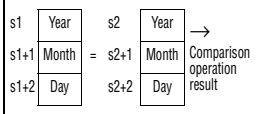
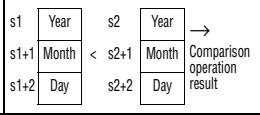
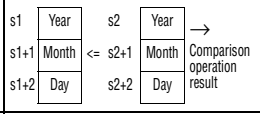
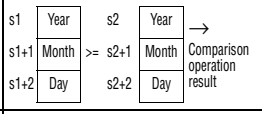
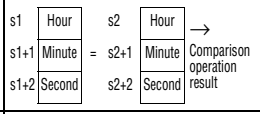
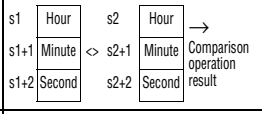
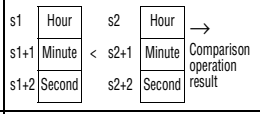
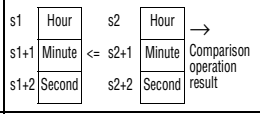
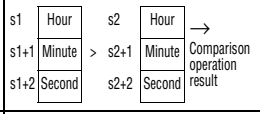
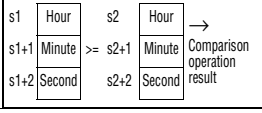
2.5.14 File register switching instructions

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Switch instruction for file register blocks	RSET RSETP	s	The RSET instruction switches from a file register block being in use by a program to a file register block with the number specified by s.		2		7.14.1
Switch instruction for file register files	QDRSET QDRSETP	s	The QDRSET instruction switches from a file register file being in use by a program to a file register file specified by s.		2 + n ①		7.14.2
Switch instruction for comment files	QCDSET QCDSETP	s	The QCDSET instruction switches from a comment file being in use by a program to a comment file specified by s.		2 + n ①		7.14.3

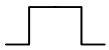
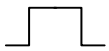
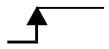
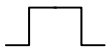
① n = (number of program name characters)/2 = Number of additional steps (Decimal fractions are rounded up)

2.5.15 Clock instructions

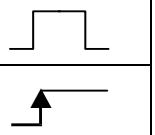
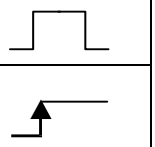
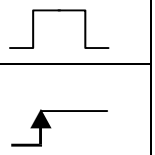
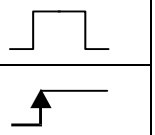
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Reading clock data	DATERD DATERDP	d	(Clock element) → d+0 Year, d+1 Month, d+2 Day, d+3 Hour, d+4 Minute, d+5 Sec., d+6 Day of the week		2		7.15.1
Writing clock data	DATEWR DATEWRP	s	s+0 Year, s+1 Month, s+2 Day, s+3 Hour, s+4 Minute, s+5 Sec., s+6 Day of the week → (Clock element)		2		7.15.2
Adding clock data	DATE+ DATE+P	s1, s2, d	s1 (Hour, Minute, Second) + s2 (Hour, Minute, Second) → d (Hour, Minute, Second)		4		7.15.3
Subtracting clock data	DATE- DATE-P	s1, s2, d	s1 (Hour, Minute, Second) - s2 (Hour, Minute, Second) → d (Hour, Minute, Second)		4		7.15.4
Changing clock data format from hh:mm:ss to seconds	SECOND SECONDP	s, d	s (Hour, Minute, Second) → d (Second)		3		7.15.5
Changing clock data format from seconds to hh:mm:ss	HOUR HOURP	s, d	s (Second) → d (Hour, Minute, Second)				

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Date comparison	LDDT=	s1, s2, n			4		7.15.6
	ANDDT=						
	ORDT=						
	LDDT<>	s1, s2, n			4		
	ANDDT<>						
	ORDT<>						
	LDDT>	s1, s2, n			4		
	ANDDT>						
	ORDT>						
	LDDT<=	s1, s2, n			4		
	ANDDT<=						
	ORDT<=						
	LDDT<	s1, s2, n			4		
	ANDDT<						
	ORDT<						
	LDDT>=	s1, s2, n			4		
	ANDDT>=						
	ORDT>=						
Clock comparison	LDTM=	s1, s2, n			4		7.15.7
	ANDTM=						
	ORTM=						
	LDTM<>	s1, s2, n			4		
	ANDTM<>						
	ORTM<>						
	LDTM>	s1, s2, n			4		
	ANDTM>						
	ORTM>						
	LDTM<=	s1, s2, n			4		
	ANDTM<=						
	ORTM<=						
	LDTM<	s1, s2, n			4		
	ANDTM<						
	ORTM<						
	LDTM>=	s1, s2, n			4		
	ANDTM>=						
	ORTM>=						

2.5.16 Expansion clock instructions

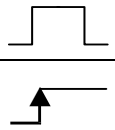
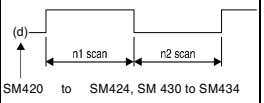
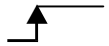
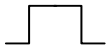
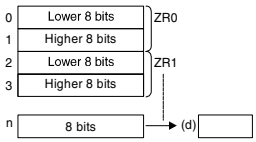
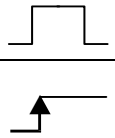
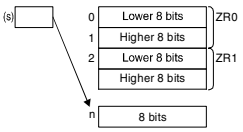
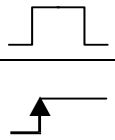
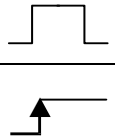
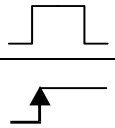
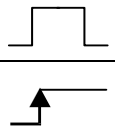
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Reading clock data of expansion block	S.DATERD	d	(Clock element) →d+0 Year; d+1 Month; d+2 Day; d+3 Hour; d+4 Minute; d+5 Sec.; d+6 Day of the week; d+7 1/1000 sec.		6		7.16.1
	SP.DATERD						
Adding clock data of expansion block	S.DATE+	s1, s2, d	(s1) Hour, Minute, Sec., —, 1/1000 sec. + (s2) Hour, Minute, Sec., —, 1/1000 sec. → (d) Hour, Minute, Sec., —, 1/1000 sec.		8		7.16.2
	SP.DATE+						
Subtracting clock data of expansion block	S.DATE-	s1, s2, d	(s1) Hour, Minute, Sec., —, 1/1000 sec. – (s2) Hour, Minute, Sec., —, 1/1000 sec. → (d) Hour, Minute, Sec., —, 1/1000 sec.		8		7.16.3
	SP.DATE-						

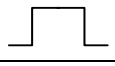
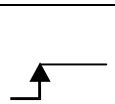
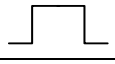
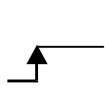
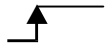
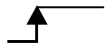
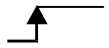
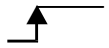
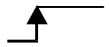
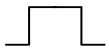
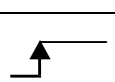
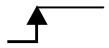
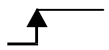
2.5.17 Program instructions

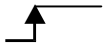
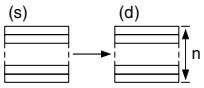
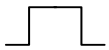
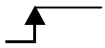
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Switching programs into stand-by mode	PSTOP PSTOPP	s	The PSTOP instruction sets the program specified by the device in s into the stand-by mode.		2 + n ①		7.17.1
Switching programs into stand-by mode and reset of outputs	POFF POFFP	s	The POFF instruction sets the program specified by the device in s into the stand-by mode and resets the outputs addressed by the program.		2 + n ①		7.17.2
Switching programs into scan execution mode	PSCAN PSCANP	s	The PSCAN instruction sets the program specified by the device in s into the scan execution mode. In this mode the program is only executed once during one program scan.		2 + n ①		7.17.3
Switching programs into low-speed execution mode	PLOW PLOWP	s	The PLOW instruction sets the program specified by the device in s into the low-speed execution mode.		2 + n ①		7.17.4
Checking the program execution status	LDPCHK ANDPCHK CRPCHK	s	In conduction when program specified by the device in s is being executed.		2 + n ①		7.17.5

① n = (number of program name characters)/2 = Number of additional steps (Decimal fractions are rounded up)

2.5.18 Other instructions

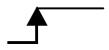
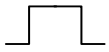
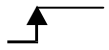
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Reset watchdog timer	WDT WDTP		The WDT instruction resets the watchdog timer (WDT) during execution of a sequence program.		1		7.18.1
Preset number of execution scans	DUTY	n1, n2, d			4		7.18.2
Time check	TIMCHK	s1, s2, d	Turns ON device specified by (d) if measured ON time of input condition is longer than preset time continuously.		4		7.18.3
Direct read of one byte	ZRRDB ZRRDBP	n, d			3		7.18.4
Direct write of one byte	ZRWRB ZRWRBP	n, s			3		7.18.5
Storing of an indirect adress	ADRSET ADRSETP	s, d	Stores the indirect adress of the device designated by s at d and d+1. This adress is used when a indirect device read is performed.		3		7.18.6
Numerical key input from keyboard	KEY	s, n, d1, d2	The KEY instruction supports the key input of 8 ASCII characters at the inputs specified by s (X). The values entered at the inputs are encoded in hexadecimal format and stored in the devices specified by d1.		5		7.18.7
Batch save of index register contents	ZPUSH ZPUSHP	d	The ZPUSH instruction saves the contents of the index registers Z0 through Z15 in d.		2		7.18.8
Batch recovery of index register contents	ZPOP ZPOPP	d	The ZPOP instruction recovers the contents of the index registers Z0 through Z15 in d.		2		7.18.8

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Reading module information	UNIRD	n1, d, n2	Reads the module information stored in the area starting from the I/O No. designated by n1 and stores it in the area starting from the device designated by d. The number of points is designated by n2.		4		7.18.9
	UNIRDP						
Reading module model name	TYPERD	n, d	This instruction reads the module information stored in the area starting from the I/O number specified by "n", and stores it in the area starting from the device specified by (D).		3		7.18.10
	TYPERDP						
Trace set/ reset	TRACE		Stores trace data set at a peripheral device to trace file in IC memory card by the designated number when SM800, SM801, and SM802 turns ON.		1		7.18.11
	TRACER		Resets the data set by the TRACE instruction		1		
Writing data to a designated file	SP.FWRITE	u0, s0, d0, s1, s2, d1	Writes data to a designated file		11		7.18.12
Reading data from a designated file	SP.FREAD	u0, s0, d0, s1, d1, d2	Reads data from a designated file		11		7.18.13
Writing data to standard ROM	S.DEVST	n1, s, n2, d	Writes data to the device data storage file in the standard ROM.		9		7.18.14
Reading data from standard ROM	S.DEVLD	n1, d, n2	Reads data from the device data storage file in the standard ROM.		8		7.18.15
	SP.DEVLD						
Loading program from memory	PLOADP	s, d	Transfers the program stored in a memory card or standard memory card (other than drive 0) to drive 0 and places the program in standby status.		3		7.18.16
Unloading program from program memory	PUNLOADP	s, d	Deletes the standby program stored in standard memory (drive 0)		3		7.18.17

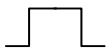
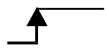
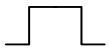
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Load and unload	PSWAPP	s1, s2, d	Deletes standby program stored in standard memory (drive 0) designated by s1.Then the program (s2) stored in a memory card or standard memory (other than drive 0) is transfered to drive 0 and placed in standby status.		4		7.18.18
Highspeed block transfer of file register	RBMOV	s, d, n			4		7.18.19
	RBMOVP	s, d, n					
User message	UMSG	s	Displays the specified character strings on the display unit as a user message.		2		7.18.20

2.6 Data link instructions

2.6.1 Instructions for network refresh

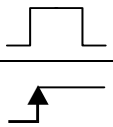
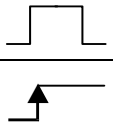
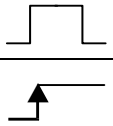
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Link instruction: Network refresh	S.ZCOM	Jn	Refreshes the designated network.		5		8.2.1
	SP.ZCOM						
	S.ZCOM	Un					
	SP.ZCOM						

2.6.2 Read/write routing information

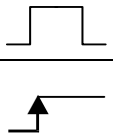
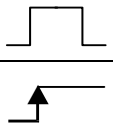
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Read/Write routing information	S.RTREAD	n, d	Reads data set at routing parameters.		7		8.3.1
	SP.RTREAD						
	S.RTWRITE	n, s	Writes routing data to the area designated by routing parameters.		8		8.3.2
	SP.RTWRITE						

2.7 Multiple CPU dedicated instruction

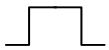
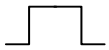
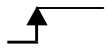
2.7.1 Instructions for writing to the CPU shared memory of host CPU

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Write to CPU shared memory	S.TO	n1, n2, n3, n4, d	Writes device data of the host station to the host CPU shared memory.		5		9.1.1
	SP.TO						
	TO	n1, n2, s, n3	Writes device data of the host station to the host CPU shared memory.		5		9.1.2
	TOP						
	DTO		Writes device data of the host station to the host CPU shared memory in 32-bit units		5		
	DTOP						

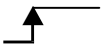
2.7.2 Instructions for reading from the CPU shared memory of another CPU

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Read from the shared memory of another CPU	FROM	n1, n2, d, n3	Reads data from the shared memory of another CPU and stores the data in the device memory of the CPU performing the FROM instruction.		5		9.2.1
	FROMP						
	DFRO		Reads data from the shared memory of another CPU in 32-bit units and stores the data in the host station.		5		
	DFROP						

2.7.3 Multiple CPU high-speed transmission dedicated instructions

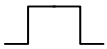
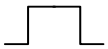
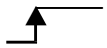
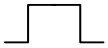
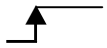
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Writing devices to another CPU	D.DDWR	n, s1, s2, d1, d2	In multiple CPU system, data stored in a device specified by host CPU (s2) or later is stored by the number of write points specified by (d2+1) into a device specified by another CPU (n) (d1) or later.		10		10.2.1
	DP.DDWR						
Reading devices from another CPU	D.DDRD		In multiple CPU system, data stored in a device specified by another CPU (d1) or later is stored by the number of read points specified by (s1+1) into a device specified by host CPU (s2) or later.		10		10.2.2
	DP.DDRD						

2.8 System switching instruction for a redundant system

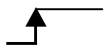
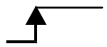
Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
System switching	SP.CONTSW	s, d	Switches between the control system and standby system at the END processing of the scan executed with the SP.CONTSW instruction.		8		11.1.1

2.9 Instructions for special function modules

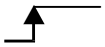
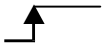
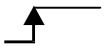
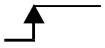
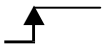
2.9.1 Instructions for serial communication modules

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Reading of data from a serial communication module	BUFRCVS	Un, n1, d1	Reading of received data from a serial communication module QJ71C24 to the PLC CPU in an interrupt program.				12.1.1
Reading of user registered frames	GETE	Un, s1, s2, d	User registered frames are read from a serial communication module				12.1.2
	GETEP						
Registration or deletion of user frames	PUTE	Un, s1, s2, d	User frames are registered to or deleted from a serial communication module				12.1.3
	PUTEP						
Transmission of data	PRR	Un, s, d	Sending of data via the serial communication module using user frames				12.1.4
	PRRP						

2.9.2 Instructions for PROFIBUS/DP interface modules

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Reading of data	BBLKRD	Un, n1, n2, d	Data is read from the buffer memory of a PROFIBUS/DP interface module and stored in the PLC CPU				12.2.1
	BBLKRDP						
Writing of data	BBLKWR	Un, n1, n2, s	Data stored in the PLC CPU is written to the buffer memory of a PROFIBUS/DP interface module				12.2.2
	BBLKWR						

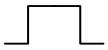
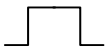
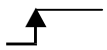
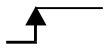
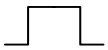
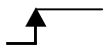
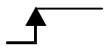
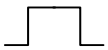
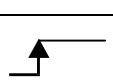
2.9.3 Instructions for ETHERNET interface modules

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Reading from fixed buffer	BUFRCV	Un, s1, s2, d1, d2	Data received during fixed buffer communication is read from the ETHERNET interface module				12.3.1
	BUFRCVS	Un, s1, d1					12.3.2
Writing to fixed buffer	BUFSND	Un, s1, s2, s3, d1	Data stored in the PLC CPU is moved to a fixed buffer of an ETHERNET interface module				12.3.3
Open connection	OPEN	Un, s1, s2, d1	Open processing for a connection				12.3.4
Close connection	CLOSE	Un, s1, s2, d1	Close processing for a connection				12.3.5
Error clear	ERRCLR	Un, s1, d1	Error codes stored in the buffer memory of the ETHERNET interface module are cleared and the "ERR." LED is switched off.				12.3.6
Reading of an error code	ERRRD	Un, s1, d1	Error codes stored in the buffer memory of the ETHERNET interface module are read to the PLC CPU				12.3.7
Re-initialization	UINI	Un, s1, d1	Re-initial processing of an ETHERNET interface module				12.3.8

2.9.4 Instruction for MELSECNET/H

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Subset	Reference Section
Pairing setting	PAIRSET	Jn, s1	Setting of stations for duplex network				12.4.1

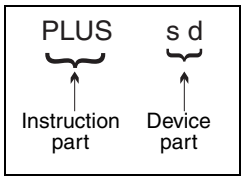
2.9.5 Instructions for CC-Link

Category	Instruction	Variables	Meaning	Execution Condition	Number of steps	Reference Section
Parameter setting	RLPASET	Un, s1 to s5, d1	Transfer of the parameter settings to the master station of CC-Link			12.5.1
	RLPASET_P					
Reading from the buffer memory or from the device memory of a CPU	RIRD	Un, s, d1, d2	Data is read from the buffer memory of another stations CC-Link module or from the device memory of that stations PLC CPU		8	12.5.2
	RIRD_P					
Writing to the buffer memory or to the device memory of a CPU	RIWT	Un, s, d1, d2	Data is written to the buffer memory of another stations CC-Link module or to the device memory of that stations PLC CPU		8	12.5.3
	RIWT_P					
Reading from an intelligent device station	RIRCV	Un, s1, s2, d1, d2	Data is read with handshake from the buffer memory of an intelligent device station connected to CC-Link		10	12.5.4
	RIRCV_P					
Writing to an intelligent device station	RISEND	Un, s1, s2, d1, d2	Data is written with handshake to the buffer memory of an intelligent device station connected to CC-Link		10	12.5.5
	RISEND_P					
Writing to automatic updating buffer memory	RITO	Un, n1, n2, n3, d	Data is moved from the device memory of the PLC CPU to the automatic updating buffer memory of the master station. This data is then transferred to another station connected to CC-Link.		9	12.5.6
	RITO_P					
Reading from automatic updating buffer memory	RIFR	Un, n1, n2, n3, d	Data transmitted from another station to the automatic updating buffer memory of the master station is moved to the device memory of the PLC CPU.		9	12.5.7
	RIFR_P					

3 Configuration of Instructions

3.1 The structure of an instruction

Most of the instructions consist of an instruction part and a device part. Other instructions do not require a device part and thus only consist of the instruction part.



Instruction part

The instruction part describes the functions of the instruction.

PLUS $\hat{=}$ Addition

Device part

The device part describes the constants or variables to be specified. The device part can comprise three items: the source of data (s), the destination of data (d), and the number (n).

3.1.1 Source of data (s)

- The data source designates the devices to be processed by the instruction.
  For 16-bit instructions the notation of the data source is s.
  For 32-bit instructions its notation is s+1 and s.
- Within the data source constants or variables can be specified.

Constants

Constants specify a constant numerical value to be processed by the instruction. This value is constantly set by the user written program and cannot be altered during program execution. It is recommended to index qualify each variable to be used as constant.

Variables

Variables specify a device storing data to be processed by the instruction (also refer to section 3.4 "Programming of variables").

Before an instruction is executed, the data must be stored in the device. The data stored in variables can be altered during program execution.

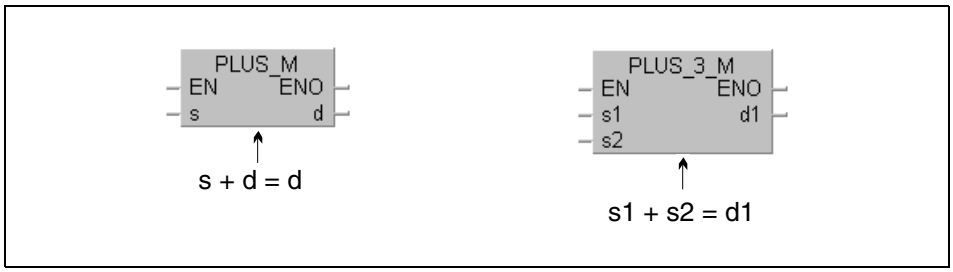
3.1.2 Destination of data (d)

- The data destination designates the devices to store the data after being processed by the instruction.

  For 16-bit instructions the notation of the data destination is d.

  For 32-bit instructions its notation is d+1 and d. However, some instructions with 2 devices require a value to be processed stored in the data destination d before the instruction is executed. In this case, the result of the operation will be stored in the same device as well.

  Example: The addition instruction for BIN 16-bit data.
  Here, d first stores data for the operation and then the operation result:

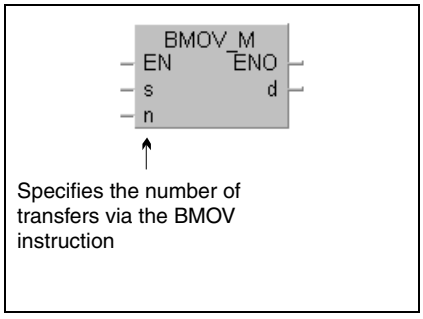


- A device for the storage of data has always to be set as data destination.

3.1.3 Number (n)

- The number n specifies how many devices are to be used or how often an instruction is to be executed.

  Example: The BMOV instruction for block data transfer:



- The value n may range from 0 to 32767. If n is specified 0, the instruction will not be executed.

3.2 Notation of instructions

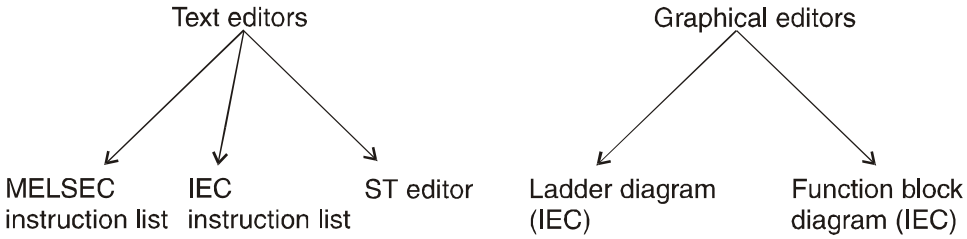
From the notation certain characteristics of the instructions can be derived.

3.2.1 16/32-bit and pulse

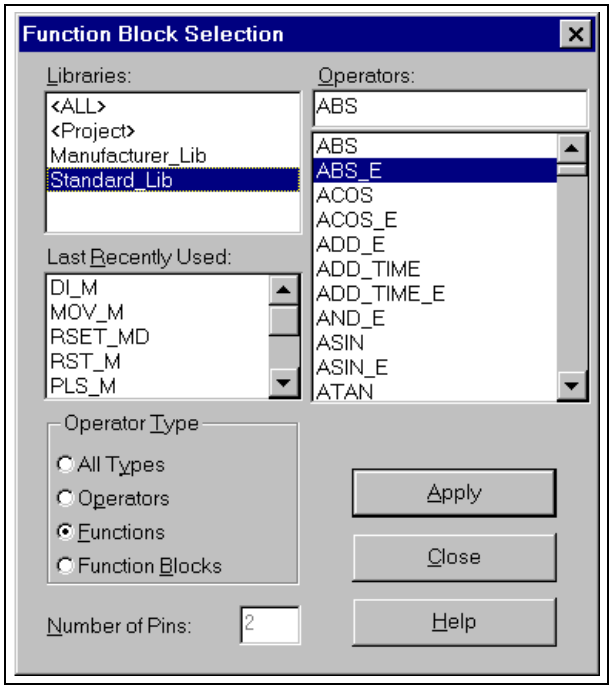
SORT	16-bit processing
SORTP	16-bit processing with pulse
DSORT	32-bit processing
DSORTP	32-bit processing with pulse

3.2.2 MELSEC and IEC

The GX IEC Developer includes several editors for the instructions:



Within these editors the instructions are represented in different notations.



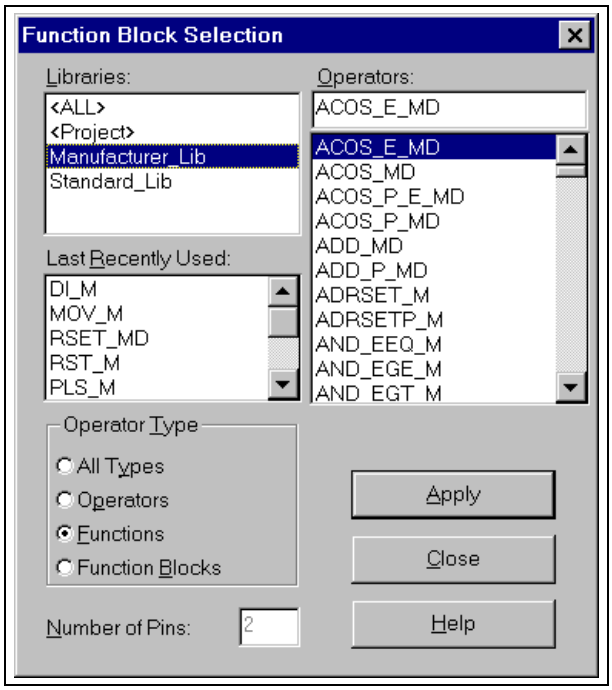
For the selection of an instruction in the GX IEC Developer this dialog box will appear.
Depending on the selected library different instructions can be chosen:

ALL: MELSEC and IEC instructions

Project: Functions and Function Blocks created by the user

Manufacturer: MELSEC instructions

Standard: IEC instructions



For example, this dialog box will appear when the the manufacturer library is selected. This listing contains the "adapted" MELSEC instructions.

The functions of the "pure" and "adapted" instructions are identical. Only their notation differs.

Legend of the extensions within the IEC editor:

Extension in IEC Editor	Meaning
_M	MELSEC instruction
_P_M	Pulse execution of an instruction
_MD	Dedicated MELSEC instruction (also refer to section 3.3 "Programming of dedicated instructions")
_P_MD	Pulse execution of a dedicated instruction
_K_MD	Use of a constant in a dedicated instruction
_K_P_MD	Use of a constant and pulse execution in a dedicated instruction.
_S_MD	Dedicated MELSEC instruction for CPUs of MELSEC System Q
_P_S_MD	Pulse execution of a dedicated MELSEC instruction for CPUs of MELSEC System Q

3.2.3 Further characteristics of the instruction notation

The table below contains the symbols that represent several functions within the MELSEC editor. The column on the right shows the according instruction names within the IEC editor.

Example: MELSEC editor IEC editor

LD$> LD_STRING_GT_M

MELSEC Editor	IEC Editor
$	STRING
=	EQ
<>	NE
<=	LE
<	LT
>=	GE
>	GT
+	PLUS
-	MINUS
x	MULTI
/	DIVID

3.2.4 Specification of the notation

The chapters 5 through 12 that give a detailed description of the instructions contain illustrations of both editors, i.e. both notations. The header line contains the "pure" MELSEC instruction as it occurs in the MELSEC instruction list.

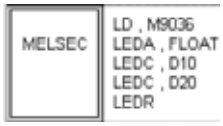
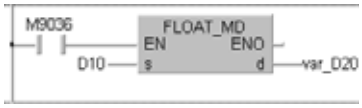
NOTE *The tabular overview at the beginning of each instruction category always represents both notations.*

3.3 Programming of dedicated instructions

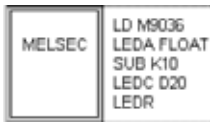
The dedicated instructions are customised instructions that do not only differ in notation from the pure MELSEC instructions. They also require a particular programming technique for the different CPUs.

In the MELSEC editor the FLOAT_MD instruction has to be programmed in combination with the LEDA, LEDC, LEDR instructions. In the IEC editors the dedicated instructions can be programmed as usual.

Example: Programming of the FLOAT_MD instruction (common execution 16-bit)

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

Example: Programming of the FLOAT_P_MD instruction
(pulse execution 16-bit, use of a constant in device s)

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

Refer to the following manuals for further information on the programming of dedicated instructions:

- GX IEC Developer Reference Manual
- Programming Manual (Dedicated Instructions)

3.4 Programming of variables

3.4.1 Programming with the GX IEC Developer

The majority of instructions besides the instruction part also require a device part with specified variables. These variables contain the values for the execution of the instruction.

According to the selected editor in the GX IEC Developer a different method of programming of the variables is required.

In the MELSEC editor:

The data registers D100 and D10 can be assigned directly to the variable designation D100 and D10.

The connected PLC automatically detects that the following devices are designated:

D100 = D100 and D101
D10 = D10, D11, D12, D13

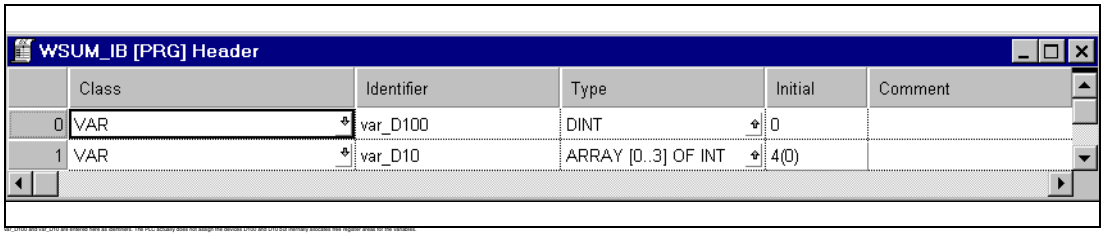
In the IEC editor:

In the IEC editor direct devices can only be entered, if actually only this device is to be designated.

Example: AND D10

Before a DWSUMP_M instruction can be processed, the variables have to be defined in the header of the program organisation unit (POU).

Example: Header of the IEC IL



Example: DWSUMP

	DWSUMP	var_D100,	4,	var_D10
		s	n	d
		↑	↑	↑
		32-bit	16-bit or constant	array

The variable var_D100 is of type DINT (32-bit). The variable var_D10 is of type ARRAY. The array contains four 16-bit registers of type INT (also refer to section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer").

Specification of the notation

The designation var_D100 or var_D10 in the screenshots indicate that not direct devices are designated but identifiers. In these cases the variable definition is compulsory! If an instruction can only be programmed over a variable definition this is explicitely noted.

NOTE *As identifier any name can be entered (e.g. Motor1, Indicator). The names var_D100 or var_D10 were selected here for a clear comparison to the programming in the MELSEC editor.*

The table of variables at the beginning of any instruction gives an overview of the data types of the devices for each instruction (the example shows the DWSUM instruction in section 7.5.14).

Variables

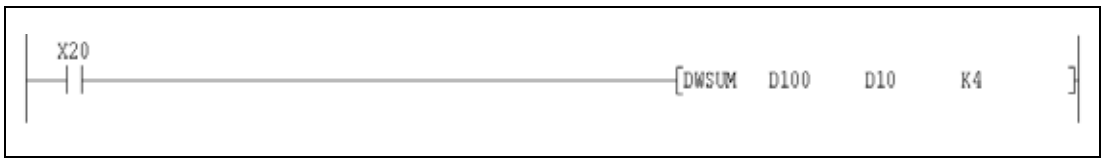
Set Data	Meaning	Data Type	
		MELSEC	IEC
s	First number of device storing data to be added.	BIN 32-bit	ANY32
d	First number of device storing result.	BIN 64-bit	Array [1..4] of ANY16
n	Number of data blocks to be added.	BIN 16-bit	ANY16

In GX Works2

The data registers D100 and D10 can be assigned directly to the variable designation D100 and D10.

The connected PLC automatically detects that the following devices are designated:

D100 = D100 and D101
D10 = D10, D11, D12, D13



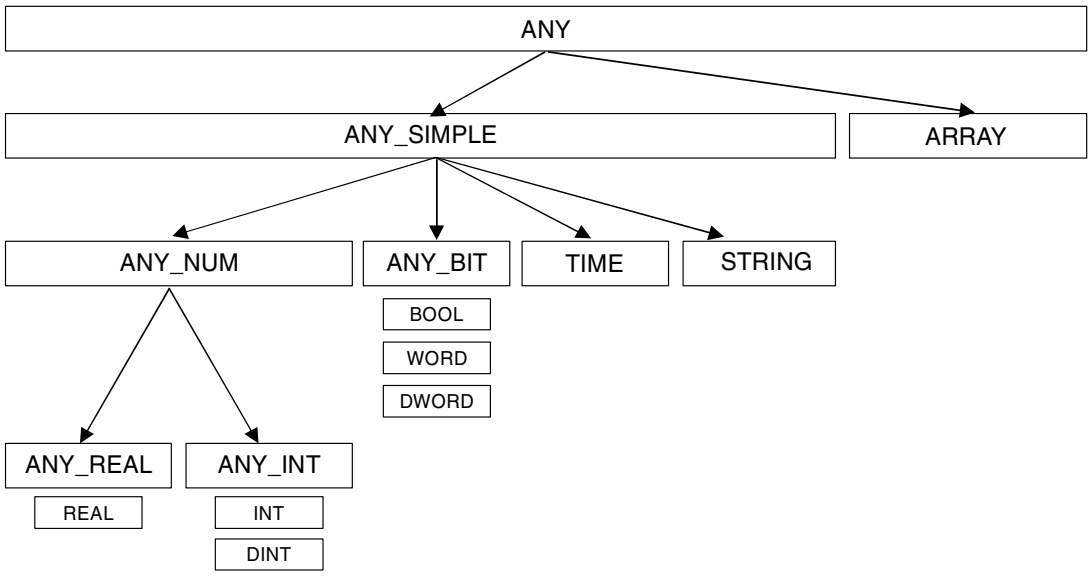
3.5 Data types

The data type determines the number and processing of bits as well as the value range of the variables.

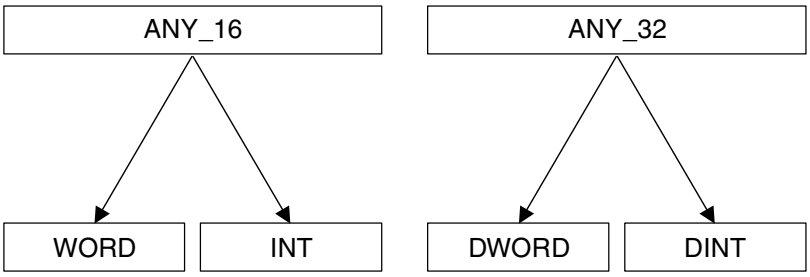
The following data types exist:

Data Type		Value Range	Number of bits
BOOL	Boolean	0 (FALSE), 1 (TRUE)	1 bit
INT	INTEGER	-32768 through 32767	16 bits
DINT	Double INTEGER	-2147483648 through 2147483647	32 bits
WORD	Bit string 16	0 through 65535	16 bits
DWORD	Bit string 32	0 through 4294967295	32 bits
REAL	Floating point number	Single precision: $-2^{128} < \text{Value} \leq -2^{-126}$, 0, $2^{-126} \leq \text{Value} < 2^{128}$	32 bits
		Double precision: $-2^{1024} < \text{Value} \leq -2^{-1022}$, 0, $2^{-1022} \leq \text{Value} < 2^{1024}$	64 bits
TIME	Time value	T#-24d-0h31m23s648.00ms through T#24d20h31m23s647.00ms	32 bits
STRING	Character string	max. 50 characters	

Hierarchy of data types ANY



Hierarchy of data types ANY16 and ANY32



Data type	Meaning
ANY	Any data type
ANY_SIMPLE	Simple data type
ANY_NUM	Numeric data type
ANY_REAL	Floating point number
ANY_INT	Integer data type
ANY_BIT	Bit processing data type
ANY_16	Any 16-bit data type
ANY_32	Any 32-bit data type
TIME	Time
STRING	Character string
REAL	Floating point number
INT	Integer value
DINT	Double integer value
BOOL	Boolean value
WORD	Word (16 bits)
DWORD	Double word (32 bits)
ARRAY	Array

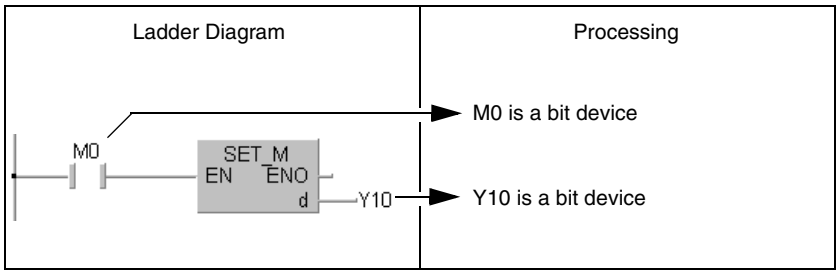
3.5.1 Processing of data

Processing of bit data

A bit device (X, Y, M, K, S, B or F) can obtain two states (ON = 1 or OFF = 0). Its status therefore can be represented by one bit (1 or 0). Bit processing is always performed, if a specified bit device is addressed by the program. For the processing of 16-bit or 32-bit instructions several bit devices are grouped in blocks of 16 or 32 device numbers (i.e. 16 or 32 addresses).

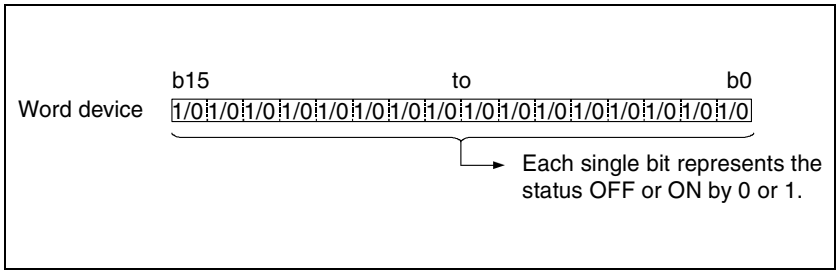
- Usage of bit devices

  A bit device (e.g. inputs, outputs, relays) consists of one bit.

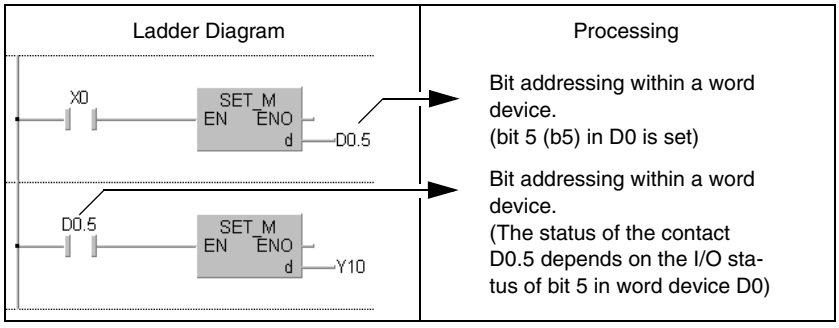


- Usage of word devices

  The CPUs of the MELSEC System Q and L series support the addressing of each single bit in a word device.



The bits have to be addressed in hexadecimal format. For example, the bit 5 (b5) in D0 is addressed D0.5. Bit 10 in D0 is addressed D0.A.
Single bits of timers, counters, and retentive timers can not be addressed.



- Usage of bit blocks

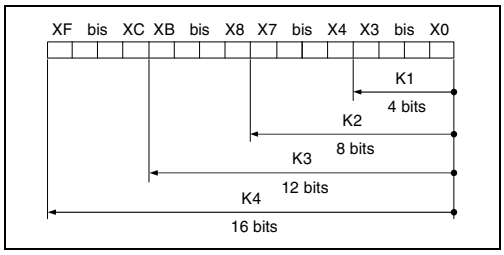
  Single bits can be grouped in blocks of four and thus process word data. The detailed description is given in the following sections, "Processing of word data (16/32 bits)".

Processing of word data (16 bits)

- Usage of bit devices

  Bit devices are capable of processing word data provided that the number of bit devices (addresses) is determined. Up to 16 bits can be processed in blocks of 4 bits each. The length of each block (i.e. the digit designation) is determined by K1 to K4.

  K1X0 4 addresses from X0 through X3
  K2X0 8 addresses from X0 through X7
  K3X0 12 addresses from X0 through XB
  K4X0 16 addresses from X0 through XF

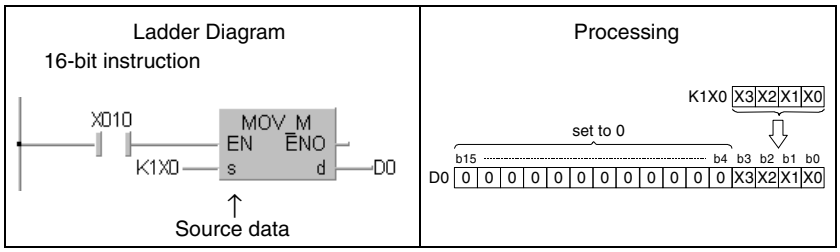


– Designation of bit blocks for s

The table below shows the range of values processed as source data for the digit designation of source data (s)

Digit Designation	16-bit instruction
K1 (4 digits)	0 to 15
K2 (8 digits)	0 to 255
K3 (12 digits)	0 to 4095
K4 (16 digits)	-32768 to 32767

The bit addresses not used are set to 0.

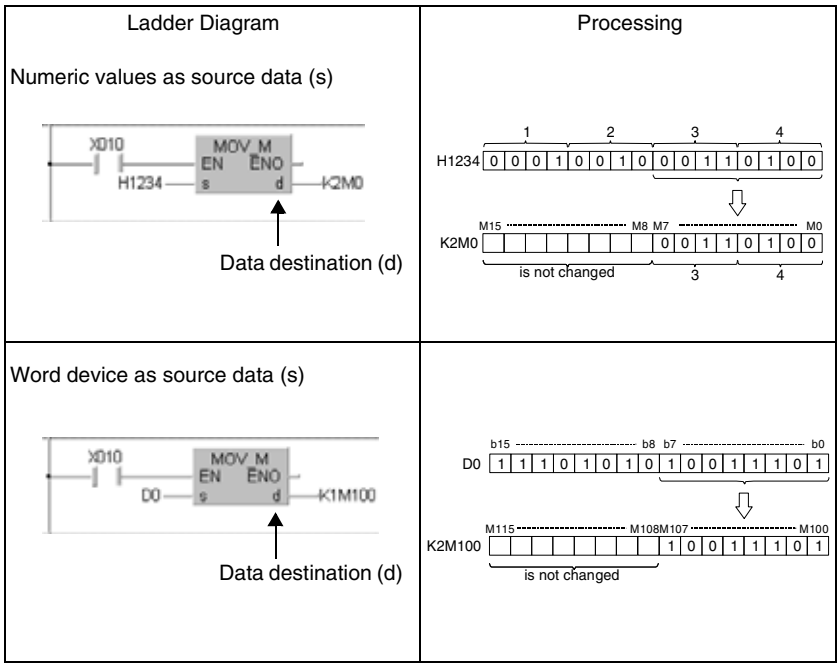


NOTE *For the block by block addressing of bit devices the number of the first bit device (initial device number) can be designated at any random value.*

Block by block addressing cannot be made for the direct access I/Os (DX, DY).

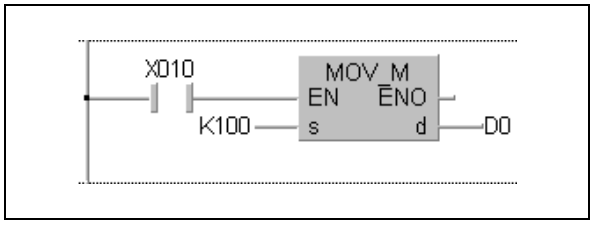
– Designation of bit blocks for d

The digit designation for the destination data (d) determines the address range the data is to be written to. The bit addresses exceeding the determined address range remain ignored.



● Usage of word devices

Word devices are determined by an address. This address comprises 16 bits.

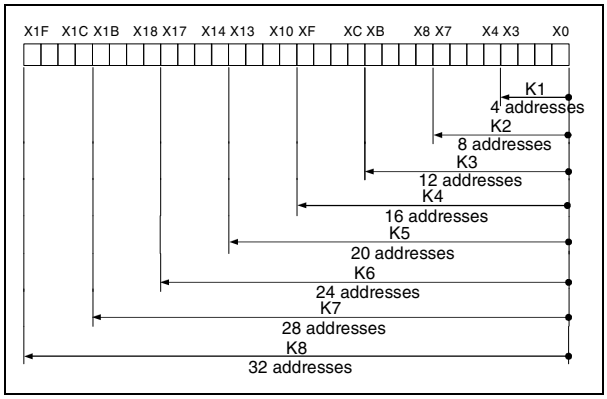


Processing of double word data (32 bits)

- Usage of bit devices

  Bit devices are capable of processing word data provided that the number of bit devices (addresses) is determined. Up to 32 bits can be processed in blocks of 4 bits each. The length of each block (i.e. the digit designation) is determined by K1 to K8.

  K1X0 4 addresses from X0 through X3
  K2X0 8 addresses from X0 through X7
  K3X0 12 addresses from X0 through XB
  K4X0 16 addresses from X0 through XF
  K5X0 20 addresses from X0 through X13
  K6X0 24 addresses from X0 through X17
  K7X0 28 addresses from X0 through X1B
  K8X0 32 addresses from X0 through X1F

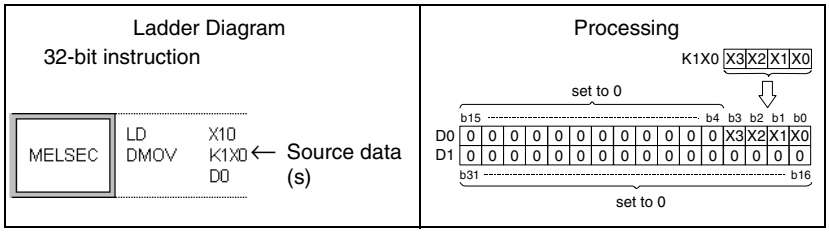


– Designation of bit blocks for s

For a specification of the digit designation the range of the values processed as source data is listed in the table below:

Digit Designation	32-bit Instruction
K1 (4 digits)	0 to 15
K2 (8 digits)	0 to 255
K3 (12 digits)	0 to 4095
K4 (16 digits)	-32768 to 32767
K5 (20 digits)	0 to 1048575
K6 (24 digits)	0 to 16777215
K7 (28 digits)	0 to 268435455
K8 (32 digits)	-2147483648 to 2147483647

The bit addresses not used are set to 0.

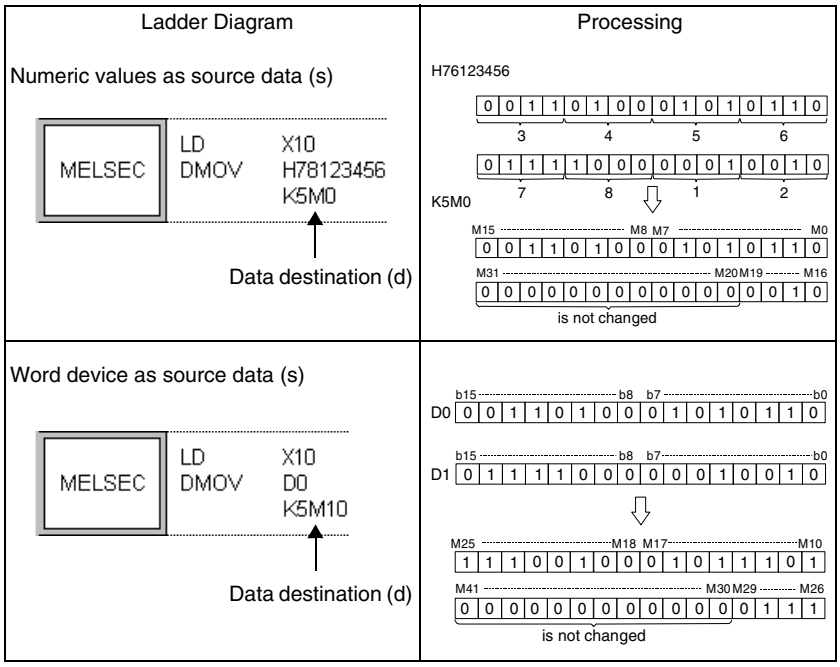


NOTE *For the block by block addressing of bit devices the number of the first bit device (initial device number) can be designated at any random value.*

Block by block addressing cannot be made for the direct access I/Os (DX, DY).

– Designation of bit blocks for d

The digit designation for the destination data (d) determines the address range the data is to be written to. The bit addresses exceeding the determined address range remain ignored.

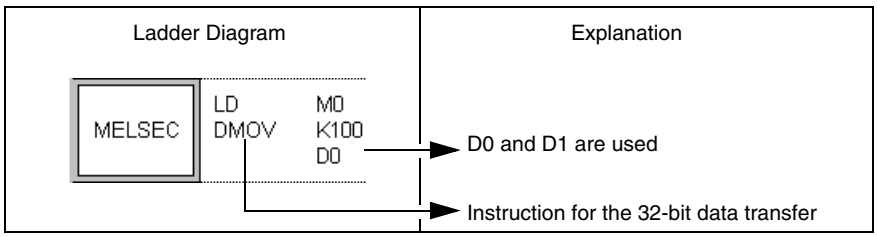


- Usage of word devices

  Double word devices comprise two 16-bit devices.

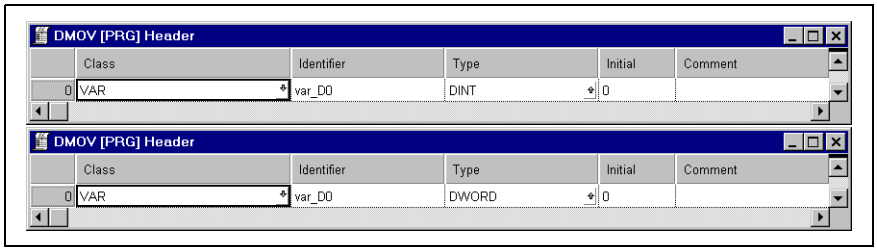
  According to the programming software and selected editor double word devices are programmed differently.

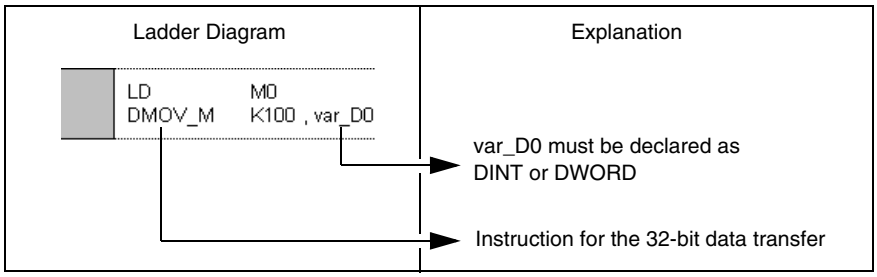
  – In the MELSEC editor of the GX IEC Developer



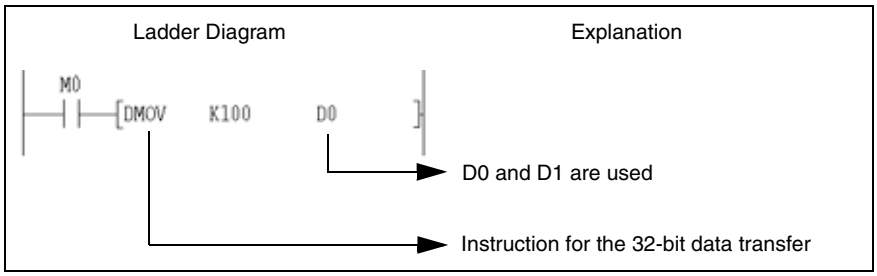
  – In the IEC editor of the GX IEC Developer

  Before a 32-bit device can be programmed in the IEC editor of the GX IEC Developer, the variables have to be defined in the header of the program organisation unit (POU).
  The data types DWORD and DINT are of the 32-bit type.





  – In the editor of the GX Works2

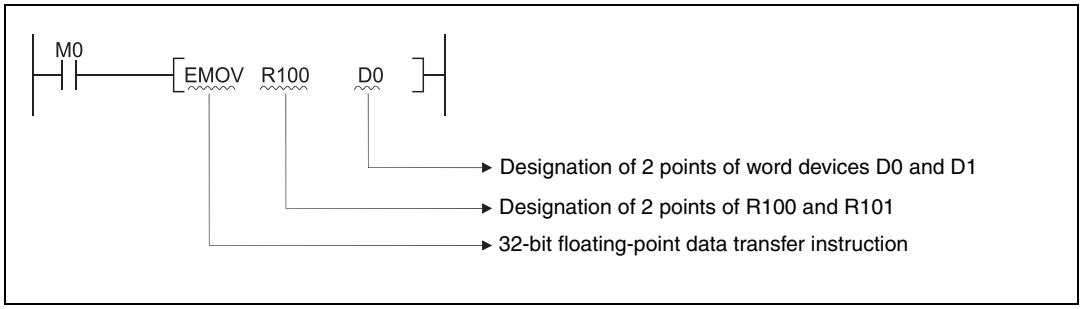


Processing of data of the data type REAL

Data of the REAL type are floating-point numbers. Whether instructions processing floating-point numbers should be performed with single precision (32-bit) or double precision (64-bit) can be set in the PLC parameters. Only word devices are capable of storing floating-point numbers.

- Single precision floating-point data

  Instructions which deal with single precision floating-point data designate devices which are used for the lower 16 bits of data. The 32-bit floating-point number is stored in two successive 16-bit registers (designated device number) and (designated device number + 1).

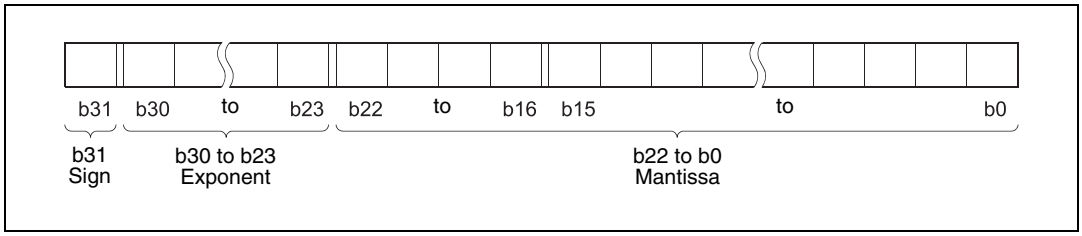


NOTES *Instructions processing floating-point numbers begin with an E (e.g. EMOV).*

Two word devices are required for storing a single precision floating-point number. Therefore, it is divided into the following components:

[Sign] 1.[Mantissa] x $2^{[Exponent]}$

The bit configuration of the registers and their contents are shown in the figure below:



– Sign of the floating-point number: The sign is stored in b31.
  0 = Positive
  1 = Negative

– Exponent: The n from 2^n is binary stored from b23 through b30.
  The meaning of the binary value n is shown in the following figure.

b23 to b30	FF$_H$	FE$_H$	FD$_H$			81$_H$	80$_H$	7F$_H$	7E$_H$			02$_H$	01$_H$	00$_H$
n	free	127	126			2	1	0	-1			-125	-126	free

– Mantissa: The 23 bits from b0 to b22, represents the XXXXXX... at binary 1.XXXXXX....

Example: Representation of the value "10" as floating-point number.
The "x" in (nnn)x designates the base of the number system.

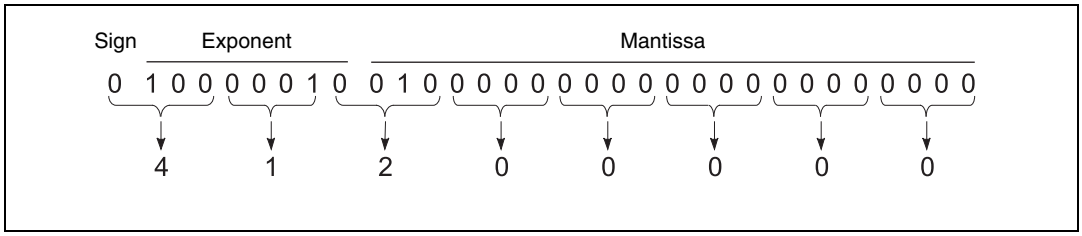
$(10)_{10} \rightarrow (1010)_2 \rightarrow (1.010000... \times 2^3)_2$

Sign: Positive → 0

Exponent: 3 → 82_H → $(10000010)_2$

Mantissa: $(010\ 00000\ 00000\ 00000\ 00000)_2$

The value "10" will be stored as 41200000_H (see following figure).



Example: Representation of the value "0.75" as floating-point number.
The "x" in (nnn)x designates the base of the number system.

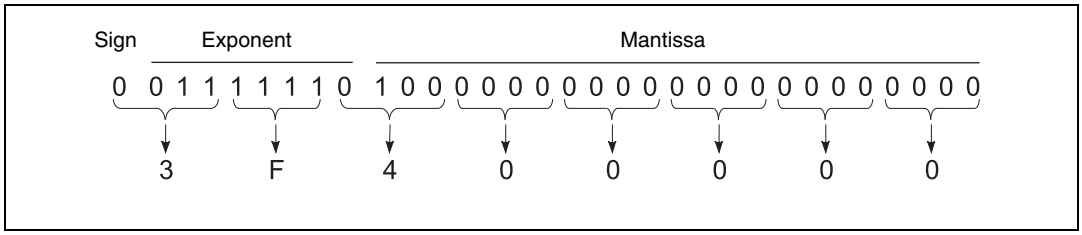
$(0.75)_{10} \rightarrow (0.11)_2 \rightarrow (1.100... \times 2^{-1})_2$

Sign: Positive → 0

Exponent: -1 → $7E_H$ → $(01111110)_2$

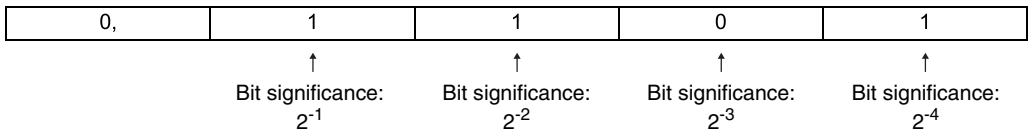
Mantissa: $(100\ 00000\ 00000\ 00000\ 00000)_2$

The value "0.75" will be stored as $3F400000_H$ (see following figure).



NOTE *Post decimal positions for binary data are represented as follows:*

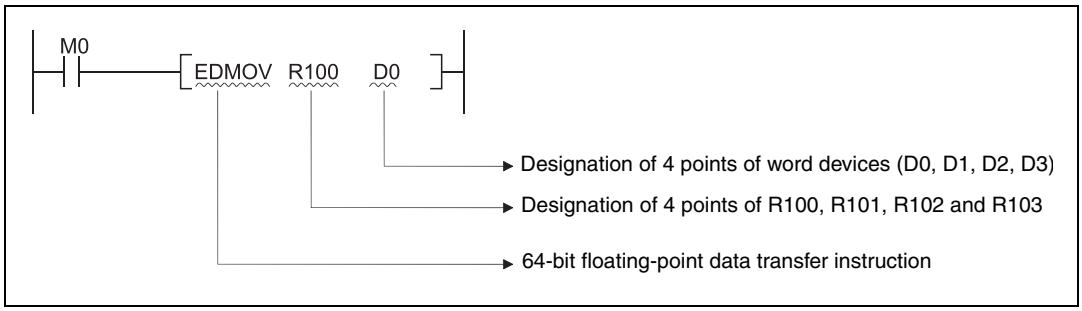
Example: $(0.1101)_2$



$(0.1101)_2 = 2^{-1}+2^{-2}+2^{-4} = 0.5 + 0.25 + 0.0625 = (0.8125)_{10}$

- Double precision floating-point data

  Instructions which deal with double precision floating-point data designate devices which are used for the lower 16 bits of data. The 64-bit floating-point number is stored in four successive 16-bit registers (designated device number) to (designated device number + 3)

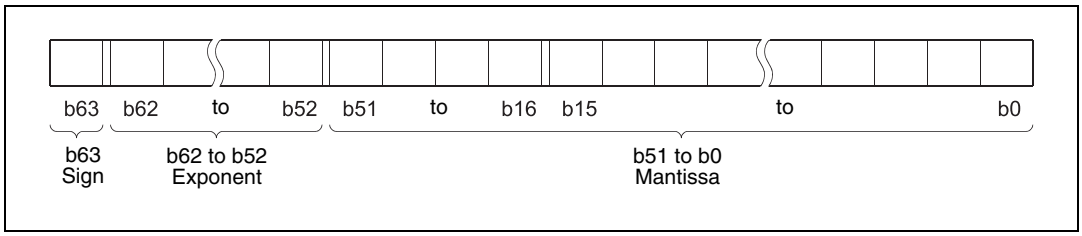


NOTE *Instructions processing floating-point numbers begin with an E (e.g. EMOV).*

Four word devices are required for storing a double precision floating-point number. Therefore, it is divided into the following components:

[Sign] 1.[Mantissa] x $2^{[Exponent]}$

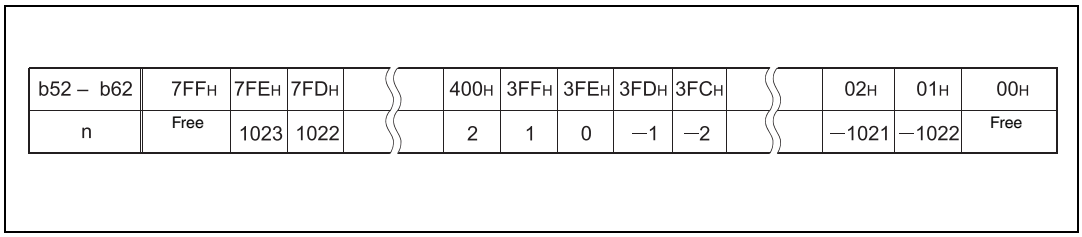
The bit configuration of the registers and their contents are shown in the figure below:.



– Sign of the floating-point number: The sign is stored in b63.

  0 = Positive
  1 = Negative

– Exponent: The n from 2^n is binary stored from bits b52 through b62.
  The meaning of the binary value n is shown in the following figure.



b52 – b62	7FF$_H$	7FE$_H$	7FD$_H$		400$_H$	3FF$_H$	3FE$_H$	3FD$_H$	3FC$_H$		02$_H$	01$_H$	00$_H$
n	Free	1023	1022		2	1	0	−1	−2		−1021	−1022	Free

– Mantissa: The 52 bits from b0 to b51, represents the XXXXXX... at binary 1.XXXXXX....

NOTES

The CPU module floating decimal point data can be monitored using the monitoring function of a peripheral device.

When floating-point data is used to express 0, the following bits are turned to 0:
Single precision floating-point data: bits b0 to b31
Double precision floating-point data: bits b0 to b63

The setting range of floating decimal point data is as follows:
Single precision floating-point data: $-2^{128} < Value \leq -2^{-126}$, 0, $2^{-126} \leq Value < 2^{128}$
Double precision floating-point data: $-2^{1024} < Value \leq -2^{-1022}$, 0, $2^{-1022} \leq Value < 2^{1024}$
For operations when a real number is out of range and operations when an invalid value is input, an error occurs. For more informations refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

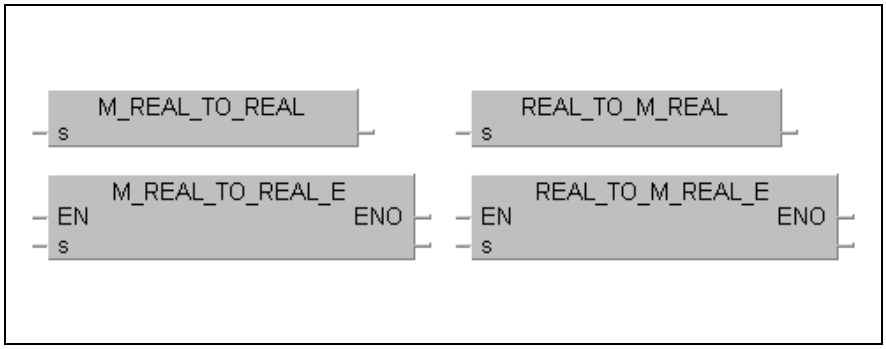
Do not specify "-0" in floating-point data. (In this case the most significant bit of the floating-point real number is "1"). An operation error will occur with the following CPU modules, if floating-point operation is performed with "–0".

- *Basic model QCPU (CPUs with first five digits of serial No. are "04122 or higher can perform floating-point operation)*
- *High Performance model QCPU where internal operation is set to single precision (setting in PLC parameter dialog box of the PLC system)*
- *Process CPU of the MELSEC System Q*
- *Redundant CPU of the MELSEC System Q*
- *Universal model QCPU of the MELSEC System Q*
- *L-series CPUs*

The High Performance model QCPU with the internal processing set to "double precision" (double precision is set by default for the floating-point operation processing) internally convert the value "–0" to 0 to perform a floating-point operation. Therefore an operation error does not occur.

- Floating-point data in the IEC Editor

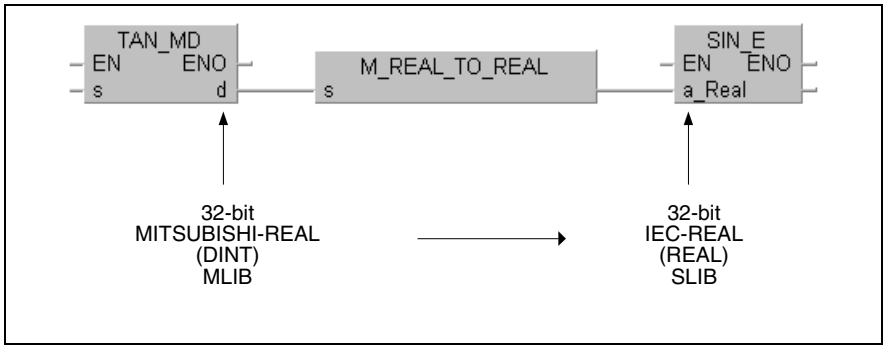
  Since the REAL IEC function uses the data type REAL as input/output but the MELSEC instructions use the data type DINT, the following functions are provided to compensate this difference:



The conversion from the IEC data type REAL into the MELSEC data type is performed by the instruction REAL_TO_M_REAL (REAL_TO_M_REAL_E).

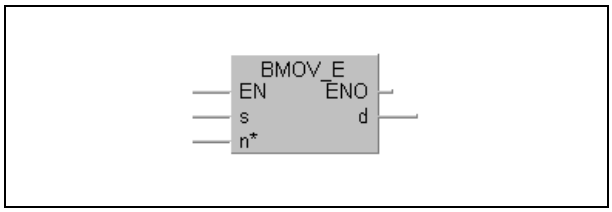
The conversion from the MELSEC data type into the IEC data type is performed by the instruction M_REAL_TO_REAL (M_REAL_TO_REAL_E).

Example: For the application of dedicated instructions that process the data type REAL and for IEC instructions the REAL to REAL conversion ist required.



When programming in in GX IEC Developer the BMOV_E instruction can be used to switch off the variable check. No additional code is created.

Any type of data can be specified in s, even arrays are possible. n holds the number of 16-bit data to copy.



3.5.2 Addressing of arrays and registers in the GX IEC Developer

Addressing of 32-bit registers

The addressing of 32-bit registers (data type DINT, DWORD) requires a variable definition in the header of the program organisation unit (POU).

In the following example the DMOV instruction requires two 16-bit registers for moving one 32-bit data word. For the addressing in the MELSEC editor of the GX IEC Developer only the initial registers (here D10, D20) are designated. Each required second 16-bit register (D11, D21) is addressed automatically by the compiler.

In the IEC editor of the GX IEC Developer instead of the initial register a variable (here var_D10, var_D20) with a specific data type (here DINT (32 bits)) has to be defined in the header of the program organisation unit according to the header of the instruction. For these variables the compiler assigns corresponding addresses internally.

MELSEC Instruction List

Ladder Diagram

IEC Instruction List

Header of the DMOV instruction

	Class	Identifier	Type	Initial	Comment
0	VAR_INPUT	s	ANY32	0	
1	VAR_OUTPUT	d	ANY32	0	

Header of the program organisation unit (POU)

	Class	Identifier	Type	Initial	Comment
0	VAR	var_D10	DINT	0	
1	VAR	var_D20	DINT	0	

Addressing of arrays

For the programming of instructions that use an array with array elements as input or output devices (16-bit registers) the variables in the header of the program organisation unit have to be defined according to the header of the instruction.

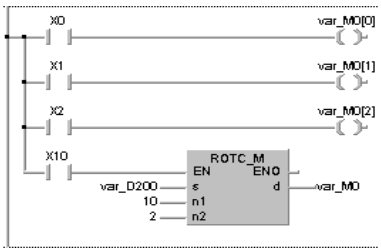
The individual array elements are addressed by specifying the array and the array element in square parentheses (var_xx[x]).

The figures below show the addressing via arrays for the positioning instruction for rotary tables (ROTC):

MELSEC Instruction List

```
MELSEC   LD    X0
         OUT   M0
         LD    X1
         OUT   M1
         LD    X2
         OUT   M2
         LD    X10
         ROTC  D200
               K10
               K2
               M0
```

Ladder Diagram



IEC Instruction List

```
LD      X0
ST      var_M0[0]
LD      X1
ST      var_M0[1]
LD      X2
ST      var_M0[2]
LD      X10
ROTC_M  var_D200 , 10 , 2 , var_M0
```

Header of the ROTC instruction

	Class	Identifier	Type	Initial	Comment
0	VAR_INPUT	s	ARRAY [1.. 3] OF ANY16	0,0,0	
1	VAR_INPUT	n1	ANY16	0	
2	VAR_INPUT	n2	ANY16	0	
3	VAR_OUTPUT	d	ARRAY [1.. 8] OF BOOL	8(FALSE)	

Header of the program organisation unit (POU)

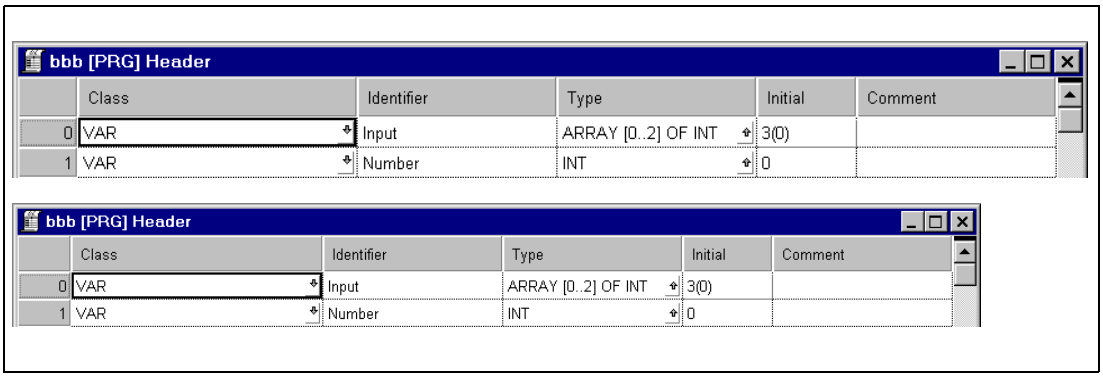
	Class	Identifier	Type	Initial	Comment
0	VAR	var_D200	ARRAY [0..2] OF INT	3(0)	
1	VAR	var_M0	ARRAY [0..7] OF BOOL	8(FALSE)	

You can infer from the header of the ROTC instruction that the input device range s consists of 3 array elements of the type ANY16 and the output device range consists of 8 array elements of the type BOOL.

In the GX Works2 and in the MELSEC editor of the GX IEC Developer for the input/output device ranges s and d only each of the initial devices D200 and M0 is specified. The compiler addresses the registers D200 through D202 for s and M0 through M7 for d.

In the IEC editors arrays must be defined for s and d. The input array s is defined as var_D200. It consists of 3 array elements (var_D200[0] – var_D200[2]) of the type INT (16-bit integer). The output array d is defined as var_M0. It consists of 8 array elements (var_M0[0] – var_M0[7]) of the type BOOL (bit). For these variables the compiler assigns corresponding addresses internally.

NOTE *Arrays can also be addressed variably. In this case instead of the array element number in square brackets any identifier for example [Number] is entered. "Number" must be declared in the header of the program organisation unit. Then a value corresponding to the according array element can be moved to the register "Number".*



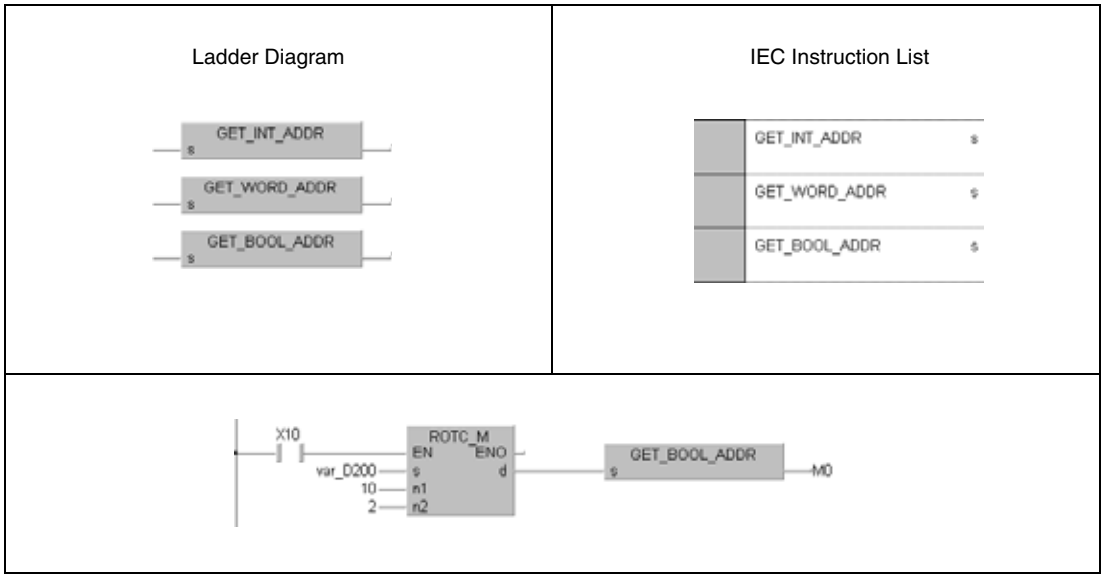
Instructions for the array address/ initial address conversion

The instruction set for the conversion of an output array into an initial address of a device range comprises three instructions.

The instruction GET_INT_ADDR converts an output array with array elements of the type INT (16-bit integer) into an initial address of a device range.

The instruction GET_WORD_ADDR converts an output array with array elements of the type WORD (16-bit word) into an initial address of a device range.

The instruction GET_BOOL_ADDR converts an output array with array elements of the type BOOL (bit) into an initial address of a device range.



After the conversion the array elements can be processed as individual devices. Therefore, the variable definition in the header of the program organisation unit is not required.

In the program with the ROTC instruction shown above instead of the array elements var_M0[0] – var_M0[7] the relays M0 through M7 can be used.

The methods of addressing devices in GX Works2 and the GX IEC Developer are identical.

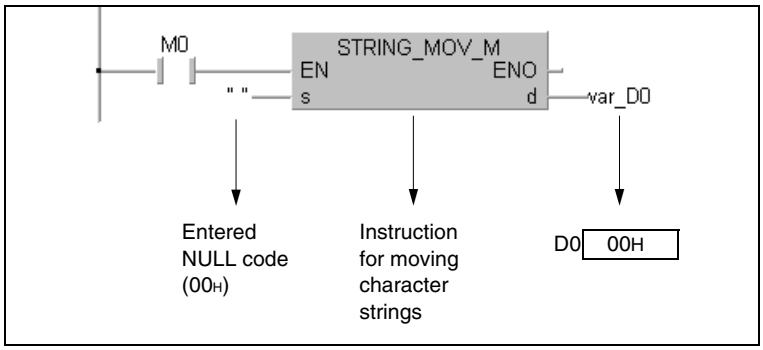
These instructions only convert output arrays. Input arrays must be addressed and declared as previously described.

3.5.3 Usage of character string data (STRING)

The data string STRING ($) processes character strings.
Character strings are all entered characters (max. 50 characters) up to the NULL code (00H).

- If the entered character is the NULL code (00H)

  For the storage of the NULL code a data word (register) is required.

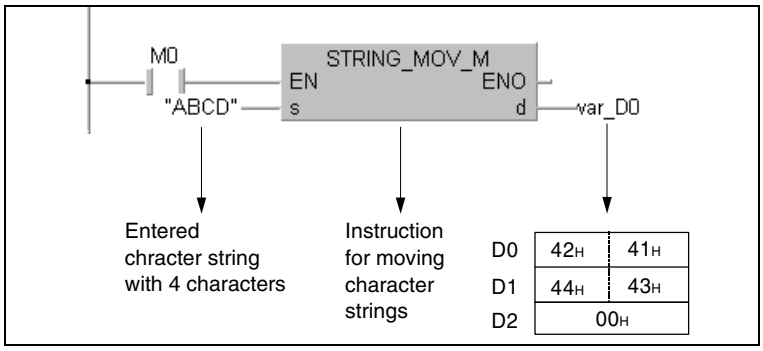


- If the number of characters contained in the string is even

  The storage of character strings with an even number of characters requires a number of data words calculated by the following formula:

  (Number of characters / 2) + 1

  If for example the character string "ABCD" is to be moved to D0, the registers D0 through D1 are required for the string and the register D2 is required for the NULL code indicating the end of string.

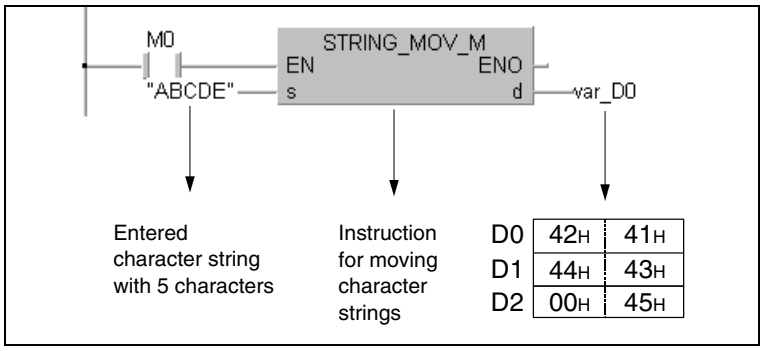


- If the number of characters contained in the character string is odd

  The storage of character strings with an uneven number of characters requires a number of data words calculated by the following formula:

  (Number of characters +1) / 2

  If for example the character string "ABCDE" is to be moved to D0, the registers D0 through D2 are required for the character string. The NULL code indicating the end of string is written to the upper byte of D2.



3.6 Index qualification

Overview of indexing

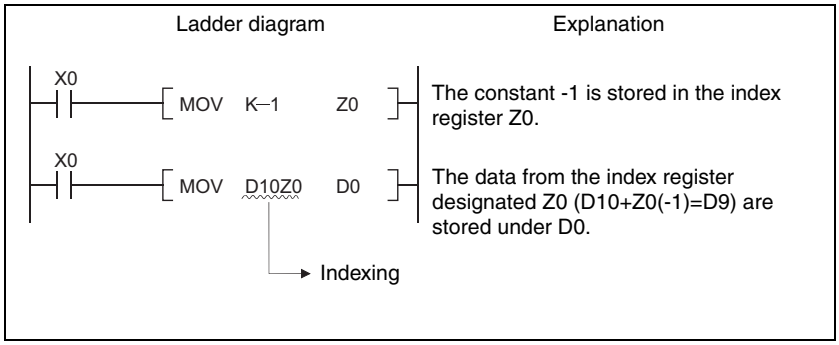
- Index qualification is an indirect addressing method of a device through an index register. For the index qualification within a program the device obtains the directly entered device number plus the contents of the index register as adress.
- Indexing with 32-bit index registers in addition to 16-bit index registers is available with the Universal model QCPU and LCPU.

Indexing with 16-bit index registers

- Example of indexing

  Each index register can be set between –32768 and 32767.

  The program shown below gives an example of the index qualification. In the first program line the value –1 is assigned to the index register Z0. This register serves as index for D10 in the second program line. Therefore, D0 stores the value of D9 (D10Z = D(10-1) = D9).



- Devices that can be designated by index qualification

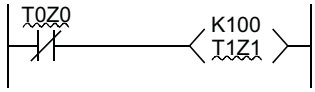
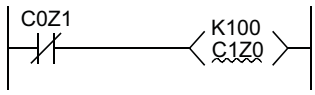
  With the exception of the restrictions noted below, Indexing can be used with devices used with contacts, coils, basic instructions, and application instructions.

  – Devices that can not be designated by index qualification

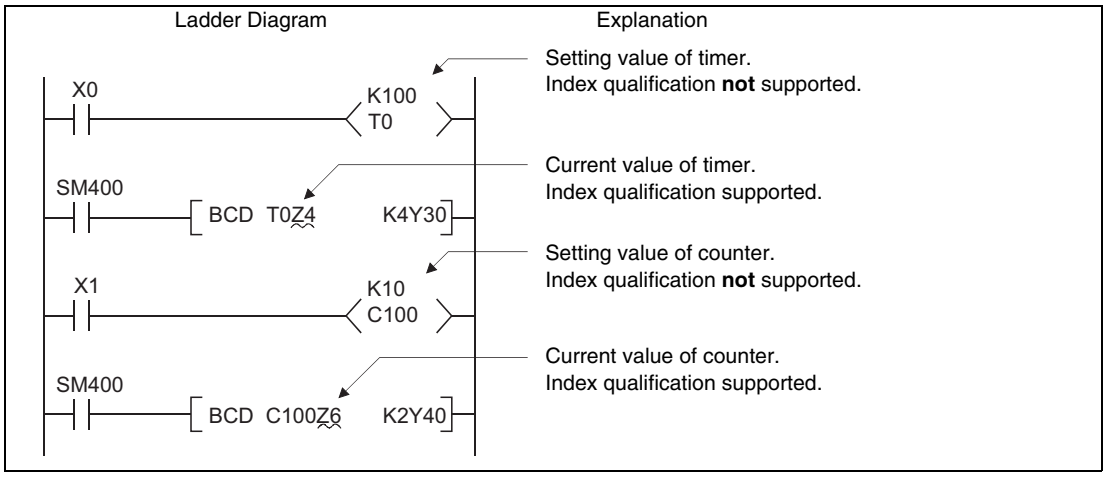
Device	Meaning
E	Floating point number
$	Character string
□.□	Bit addressing of word devices
FX, FY, FD	Function devices
P	Pointers used as label
I	Interrupt pointers used as label
Z	Index registers
S	Step relays
TR	SFC transfer devices [1)]
BL	SFC block devices [1)]

[1] SFC transfer devices and SFC block devices are devices for SFC use.
Refer to the following manual for how to use these devices: MELSEC-Q / L / QnA Programming Manual (SFC)

– Devices with limits for use with index registers

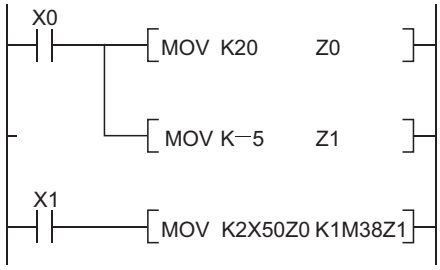
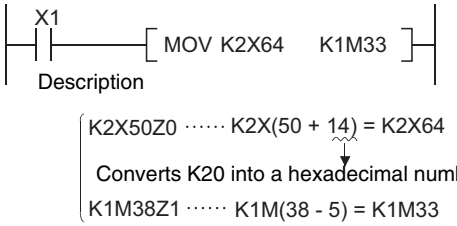
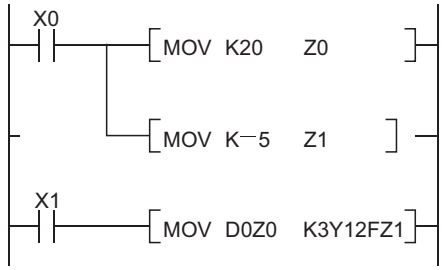
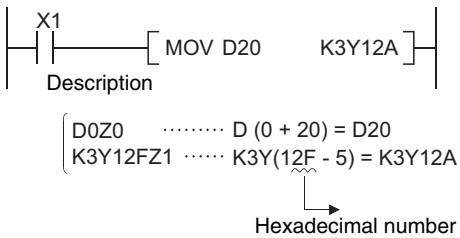
Device	Meaning	Application Example
T	Only the registers Z0 and Z1 can be used for addressing timer contacts and coils.	
C	Only the registers Z0 and Z1 can be used for addressing counter contacts and coils.	

NOTES *There are no restrictions on the addressing of current values of timers and counters.*



- A case where indexing has been performed, and the actual process device, would be as follows:

  (When Z0 = 20 and Z1 = 5)

Ladder example	Actual Process Device
	
	

Indexing with 32-bit (Universal model QCPU (excluding Q00UJCPU) and LCPU)

A method of speciyfing index registers in indexing with 32-bit can be selected from the following two methods.

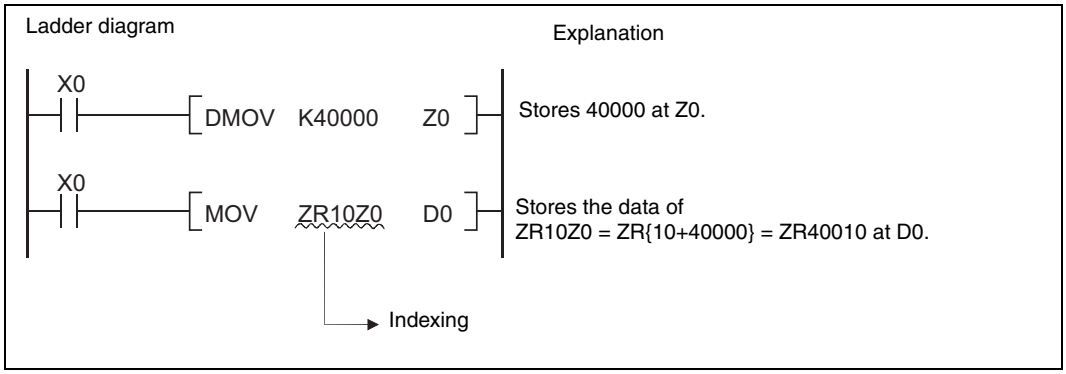
- Specifing the index registers' range used for indexing with 32-bit.
- Specifing the 32-bit indexing using "ZZ" specification.

NOTES *32-bit indexing with the "ZZ" specification is only available for the following CPU modules. See the programming tool operating manual for the available programming tools:*

- *The first five digits of the serial No. for QnU(D)(H)CPU is "10042" or higher (excluding Q00UJCPU)*
- *QnUDE(H)CPU*
- *LCPU*

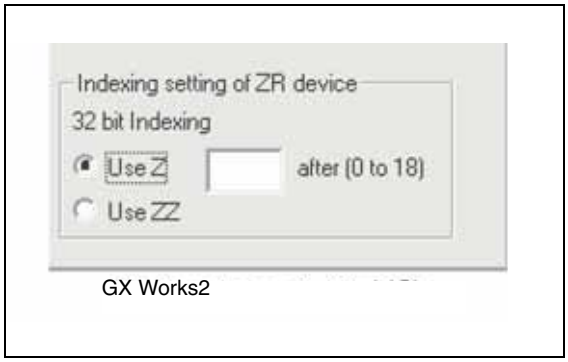
- Example of specifying the range of index registers for use of 32-bit indexing.

  Each index register can be set between -2147483648 and 2147483647.



– Specification method

For indexing with a 32-bit index register, specify the head number of an index register to be used on the Device tab of the Q parameter setting screen.



NOTES *When the head number of the index register used is changed on the Device tab of the Q parameter setting screen, do not change the parameters only or do not write only the parameters into the programmable controller. Be sure to write the parameter into the programmable controller with the program.*
When the parameter is forced to be written into the programmable controller, an error of CAN'T EXE. PRG. occurs. (Error code: 2500)

– Device that indexing can be used

Indexing can be used only for the device shown below.

Device	Meaning
ZR	Serial number access format file register
D	Extended data register (D)
W	Extended link register (W)

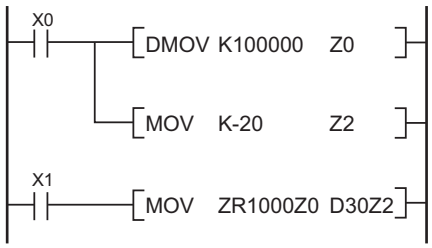
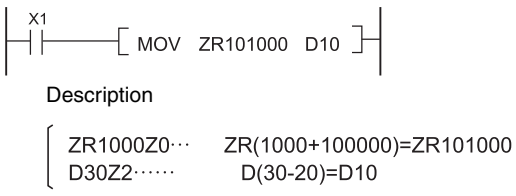
– Usable range of index registers

The following table shows the usable range of index registers for indexing with 32-bit index registers. For indexing with 32-bit index registers, the specified index register (Zn) and the next index register of the specified register (Zn+1) are used. Be sure not to overlap index registers to be used.

Setting Value	Index Registers to be used	Setting Value	Index Registers to be used
Z0	Z0, Z1	Z10	Z10, Z11
Z1	Z1, Z2	Z11	Z11, Z12
Z2	Z2, Z3	Z12	Z12, Z13
Z3	Z3, Z4	Z13	Z13, Z14
Z4	Z4, Z5	Z14	Z14, Z15
Z5	Z5, Z6	Z15	Z15, Z16
Z6	Z6, Z7	Z16	Z16, Z17
Z7	Z7, Z8	Z17	Z17, Z18
Z8	Z8, Z9	Z18	Z18, Z19
Z9	Z9, Z10	Z19	Cannot be specified

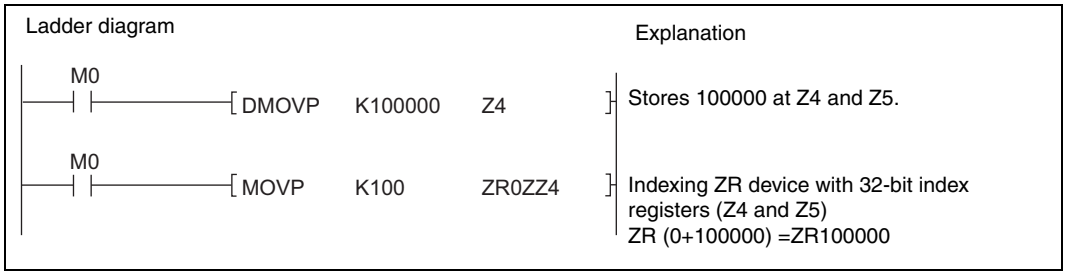
– An example of indexing and the actual process device are as follows.

(When Z0 (32-bit) = 100000 and Z2 (16-bit) = –20)

Ladder example	Actual Process Device
	

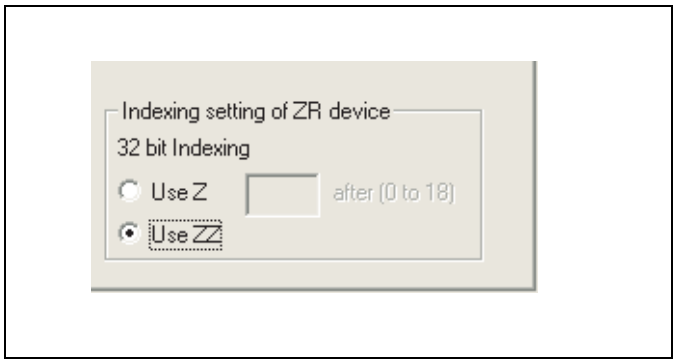
- Example of specifying 32-bit indexing with “ZZ” specification.

  One index register can specify 32-bit indexing by using “ZZ” specification such as “ZR0ZZ4”. Following figure shows an example.



– Specification method

To perform 32-bit indexing by using “ZZ” specification, select “Use of ZZ” in “Indexing Setting for ZR Device” in PC parameter.



– Device that indexing can be used

The following device is available for indexing.

Device	Meaning
ZR	Serial number access format file register
D	Extended data register (D)
W	Extended link register (W)

– Usable range of index registers

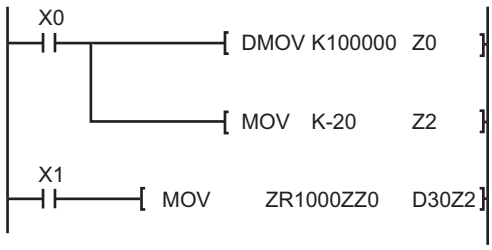
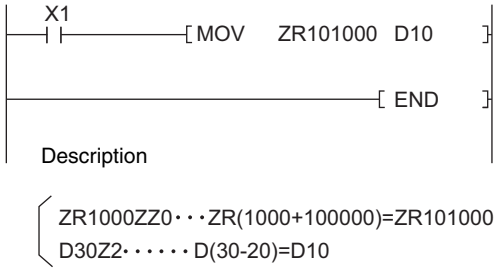
The following table shows the usable range of index registers in 32-bit indexing used “ZZ” specification. The 32-bit indexing with “ZZ” specification is specified as the format ZRmZZn. Specifying ZRmZZn enables Zn and Zn+1 of 32-bit values to index the device number.

"ZZ" specification [1)]	Index Registers to be used	"ZZ" specification [1)]	Index Registers to be used
□ZZ0	Z0, Z1	□ZZ10	Z10, Z11
□ZZ1	Z1, Z2	□ZZ11	Z11, Z12
□ZZ2	Z2, Z3	□ZZ12	Z12, Z13
□ZZ3	Z3, Z4	□ZZ13	Z13, Z14
□ZZ4	Z4, Z5	□ZZ14	Z14, Z15
□ZZ5	Z5, Z6	□ZZ15	Z15, Z16
□ZZ6	Z6, Z7	□ZZ16	Z16, Z17
□ZZ7	Z7, Z8	□ZZ17	Z17, Z18
□ZZ8	Z8, Z9	□ZZ18	Z18, Z19
□ZZ9	Z9, Z10	□ZZ19	Cannot be specified

[1] refers to device name (ZR) for indexing target

– Following example shows the 32-bit indexing using the “ZZ” specification and the actual processing device:

(When Z0 (32-bit) = 100000 and Z2 (16-bit) = –20)

Ladder example	Actual Process Device
	

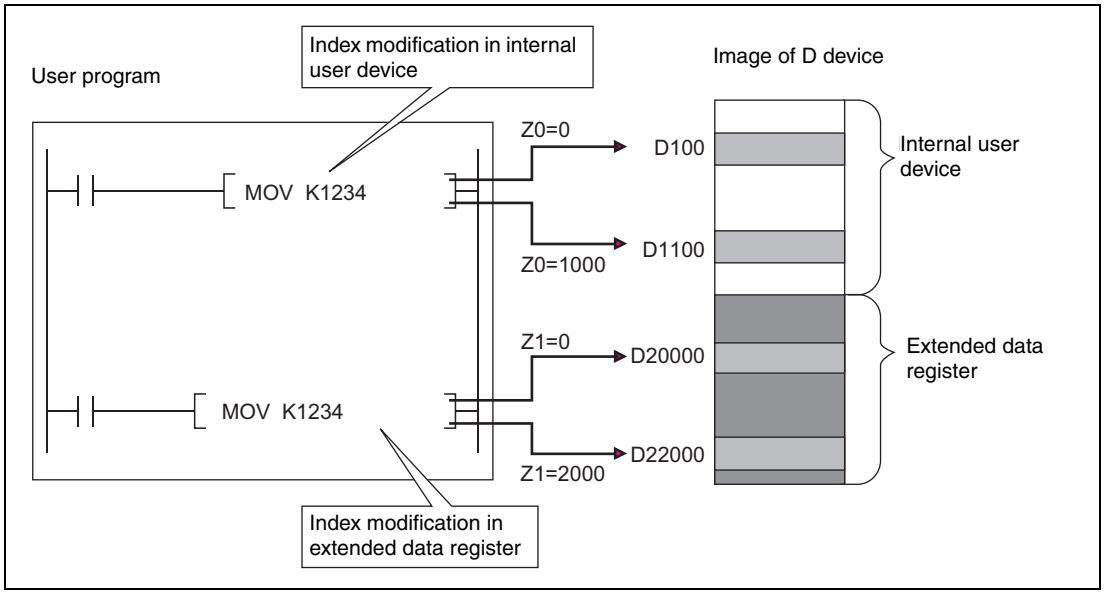
– Available functions for “ZZ” specification

- Specifying devices in program instruction
- Monitoring device registrations
- Testing devices execution type
- Testing devices with conditions
- Setting monitor conditions
- Tracing sampling (Trace point (specifing devices), trace target device)
- Data logging function (Sampling interval (specifying devices), logging target data)

NOTES *ZZn cannot be used alone as a device like “DMOV K100000 ZZ0”. When setting values of index registers to specify 32-bit indexing with “ZZ” specification, set the value of Zn (Z0~Z19). ZZn alone cannot be used as target for data transfer.*

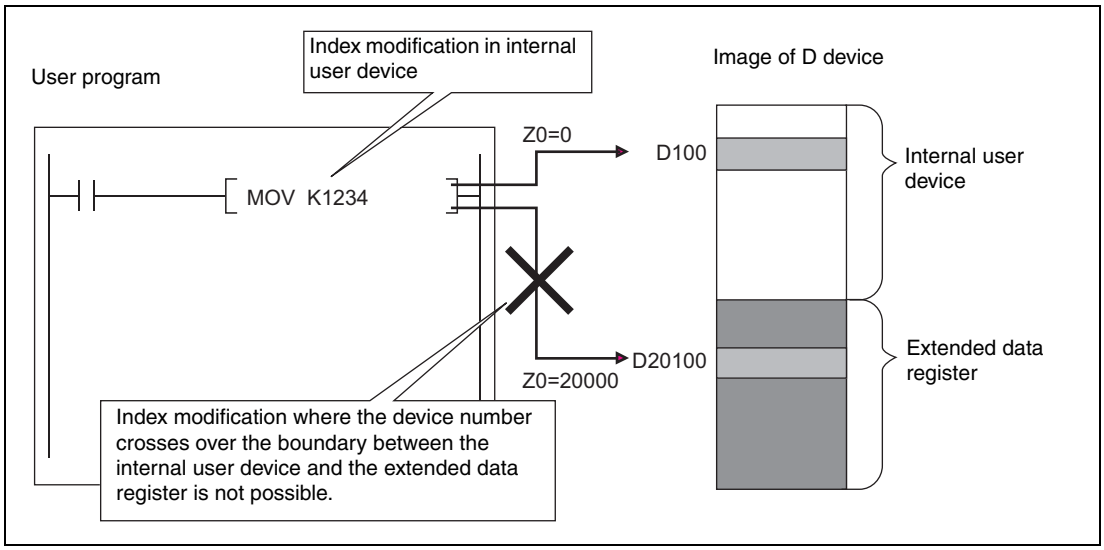
Index modification using extended data register (D) and extended link register (W) (Universal model QCPU (excluding Q00UJCPU) and LCPU)

Like index modification using data register (D) and link register (W) of internal user device, a device can be specified by index modification within the range of the extended data register (D) and extended link register (W).



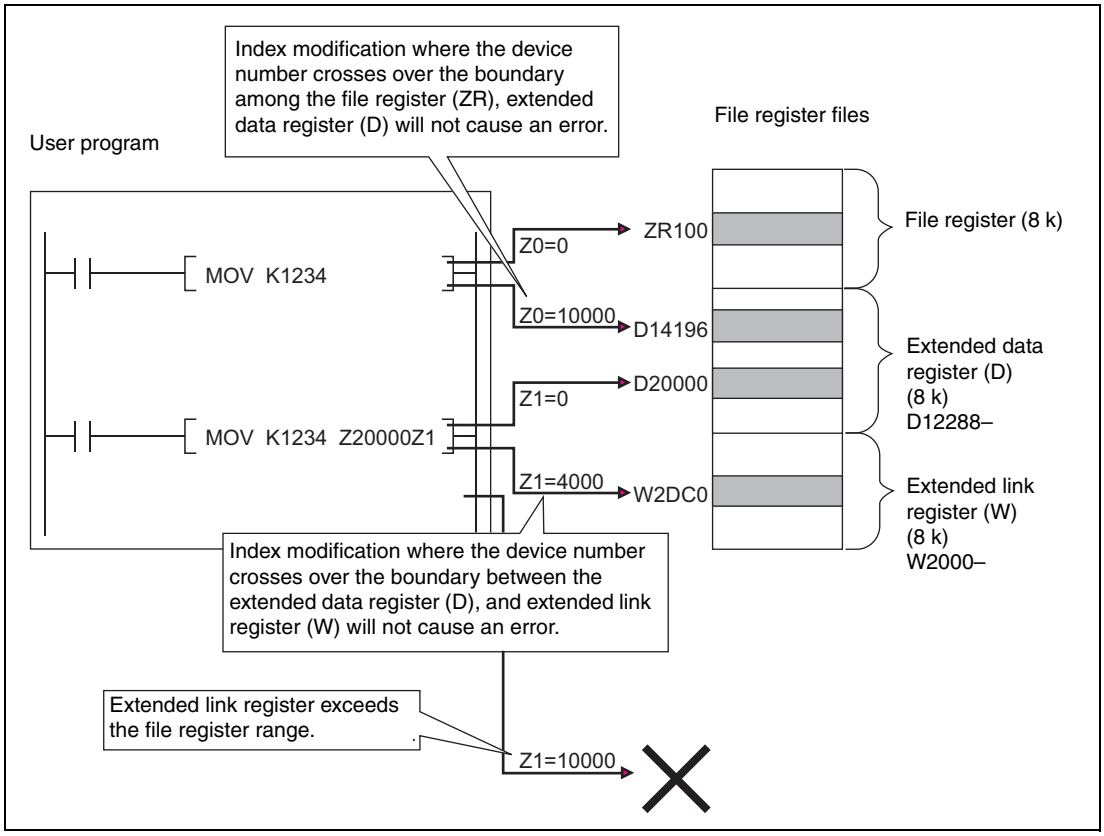
- Index modification where the device number crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W)

  The specification of index modification where the device number crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W) cannot be made. If doing so, an error occurs when the device range check is enabled at index modification (Error code 4101).



- Index modification where the device number crosses over the boundary among the file register (ZR), extended data register (D), and extended link register (W)

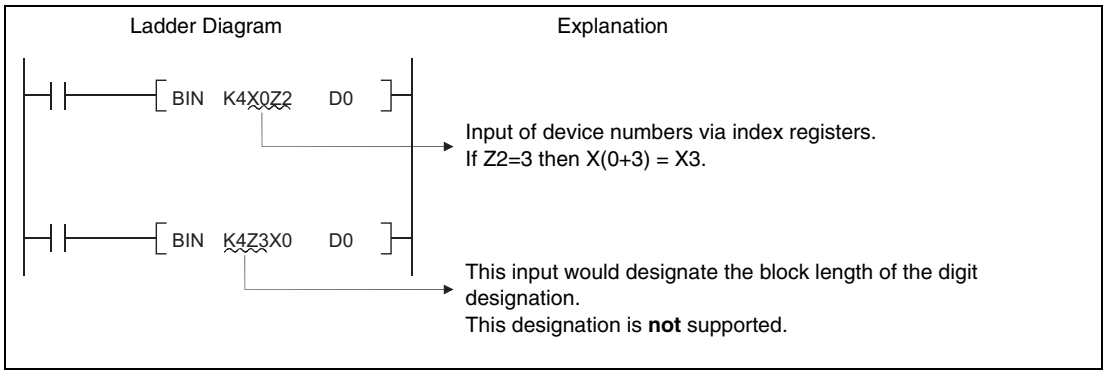
  Index modification where the device number crosses over the boundary among the file register (ZR), extended data register (D), and extended link register (W) will not cause an error. However, an error occurs if the index modification result of file register (ZR), extended data register (D), and extended link register exceeds the file register range (Error code 4101).



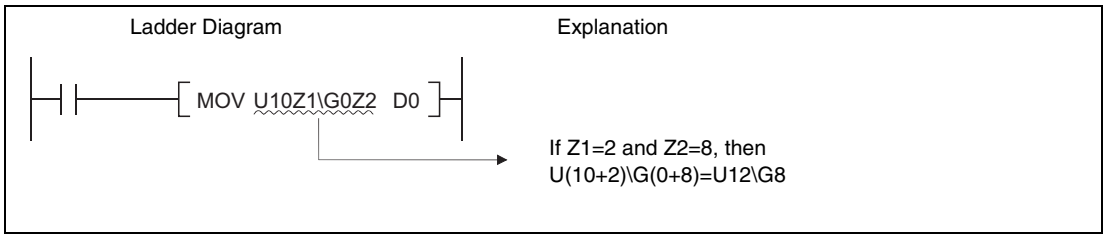
Other index modifcations

- Bit data

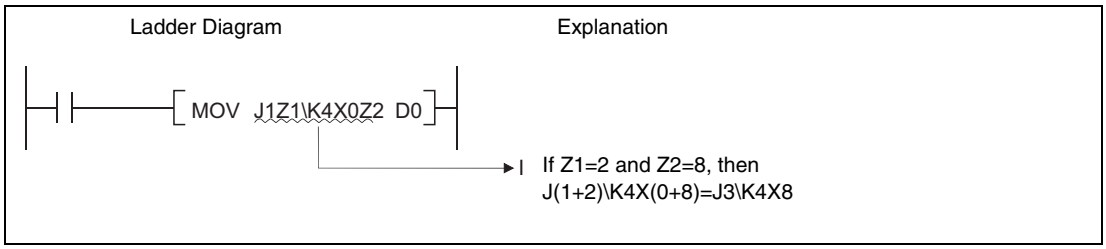
  Devices can as well be index qualified for the digit designation. The block length of the digit designation can not be affected.



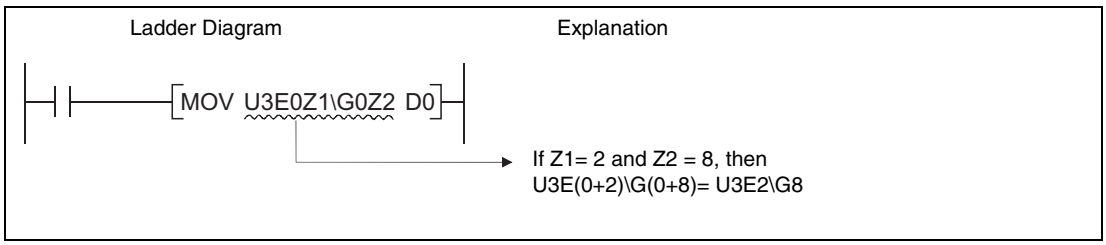
- Both I/O numbers and buffer memory number can be performed indexing with intelligent function module devices[1)]



- Both network numbers and device numbers can be performed indexing with link direct devices[1)]



- When indexing is used for multiple CPU shared devices, indexing for the head I/O numbers of CPU modules and indexing for the CPU shared memory address are automatically executed.



NOTE *For the intellingent function module device, link direct device and the multiple CPU shared device refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manuall (Function Explanation, Program Fundamentals).*

- Index modification using extended data register (D) and extended link register (W) by 32 bits (Universal model QCPU(except Q00UJCPU) and LCPU)

  Like index modification using file register (ZR), index modification using extended data register (D) and extended link register (W) by 32 bits can be performed by the following two methods:

  – Specifing the index registers' range used for indexing with 32-bit.

  – Specifing the 32-bit indexing using "ZZ" specification.

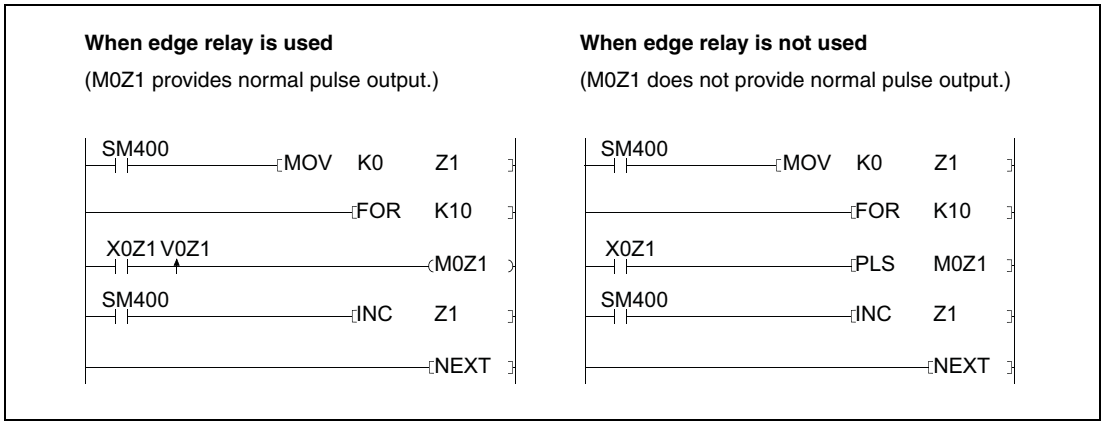
NOTES *32-bit indexing with the "ZZ" specification is only available for the following CPU modules (also refer to the User's manuals of the programming tool used):*

- *QnU(D)(H)CPU with first five digits of the serial No. is "10042" or higher (excluding Q00UJCPU)*
- *QnUDE(H)CPU*
- *LCPU*

Precautions on performing indexing

- Performing indexing between the FOR and NEXT instructions

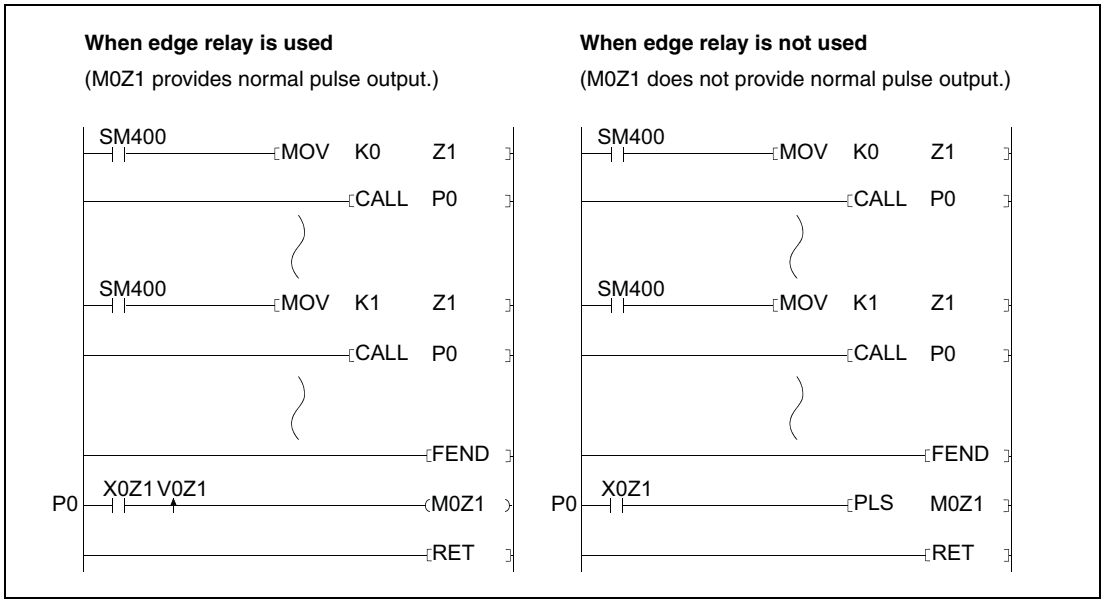
  Pulses can be output between the FOR and NEXT instructions by use of the edge relay (V). However, pulse output using the PLS/PLF/pulse (P) instruction is not allowed.



NOTES *The ON/OFF data of X0Z1 is stored by the edge relay V0Z1. For example, the ON/OFF data of X0 is stored by V0, and that of X1 by V1.*

- Performing indexing with the CALL instruction

  Pulses can be output with the CALL instruction by use of the edge relay (V). However, pulse output using the PLS/PLF/pulse (P) instruction is not allowed.



- Device range check during indexing
  - Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

    Device range checks are not conducted during indexing.
    Therefore, when the data after index modification exceed the user specified device range, the data is written to another device without causing an error.
    Note, however, that when the data after index modification is written to the device for system use exceeding the user specified device range, an error occurs. (Error code: 1103)
    Take extra precaution when using indexing in programming.

  - Universal model QCPU and LCPU

    The device range is checked for indexing.
    With changing the settings of the PLC parameter, the device range is not checked.
- Changing indexing with 16-bit index register for indexing with 32-bit index register

  For changing indexing with 16-bit index register for indexing with 32-bit index register, check if the program has enough spaces for indexing. For indexing with 32-bit index registers, the specified index register (Zn) and the next index register of the specified register (Zn+1) are used. Be sure not to overlap index registers to be used.

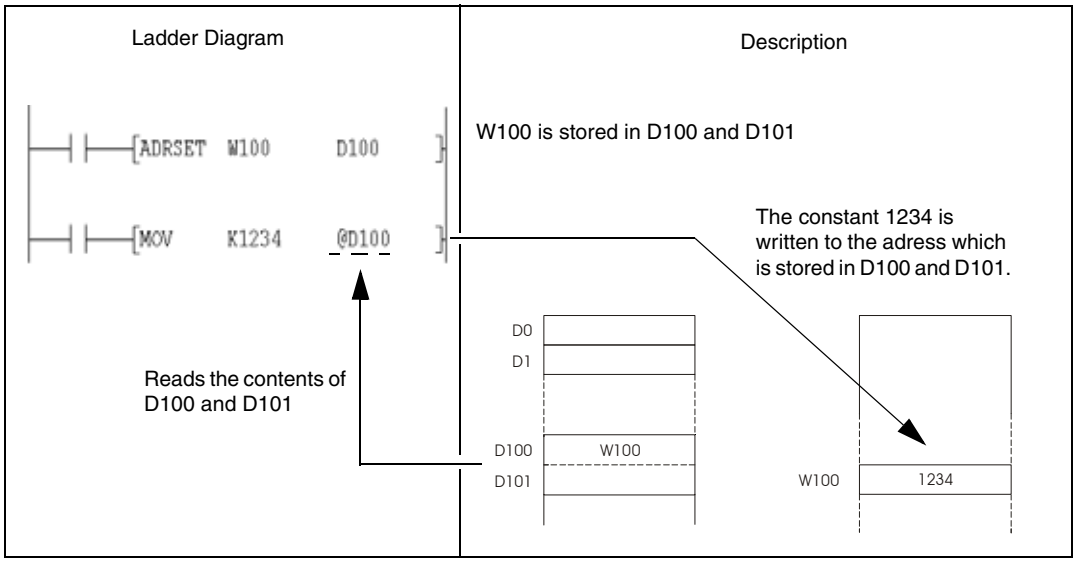
3.7 Indirect designation (GX Works2 only)

With indirect designation, a device address is stored in a word device. In the sequence program the device address is not directly designated. For operations concerning this device address the word device is used instead. This method can be used when the index register is insufficient.

The device which contains the device address for indirect designation has the prefix "@". For example, designation of @D100 will make the contents of D100 and D101 the device address.

The address of the device performing indirect designation can be stored in the word device with the ADRSET instruction.

NOTE *The ADRSET instruction is not supported by the GX IEC Developer.*

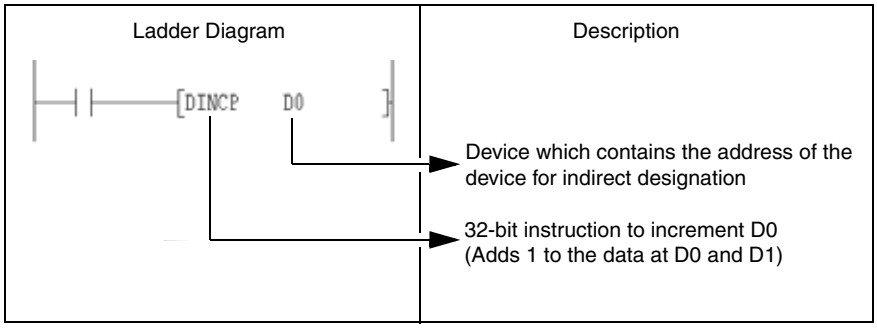


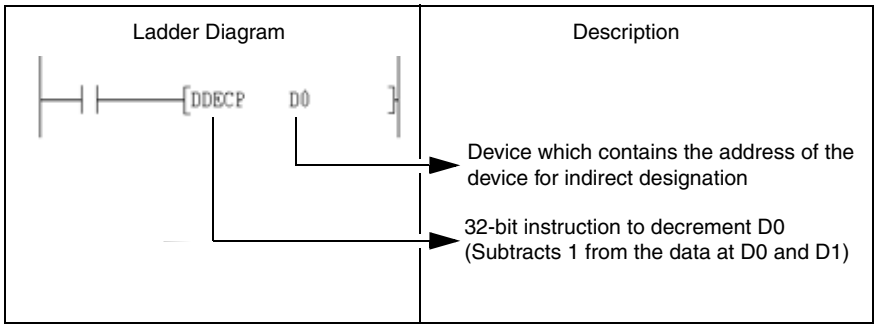
A list of devices which are capable of indirect designation is shown below.

Device Type		Indirect designation	Example of indirect designation
Internal devices (System, user)	Bit devices	Incapable	—
	Word devices	Capable	• @D100 • @D100Z2 (Index qualification)
Link direct device	Bit devices	Incapable	—
	Word devices	Capable (The ADRSET instruction cannot be used to write the indirect adress)	• @J1\W10 • @J1Z1\W10Z2 (Index qualification)
Special function module			• @U10\G0 • @U10Z1\G0Z2 (Index qualification)
Index register Zn		Incapable	—
File register		Capable	• @R0, @ZR20000 • @R0Z1, @ZR20000Z1 (Index qualification)
Extended data register (D)		Capable	• @D1000
Extended link register (W)			• @W1000
Nesting		Incapable	—
Pointer			—
Constants			—
Other			—

To store an address for indirect designation, two words are used. Therefore, to decrease or increase a stored adress for indirect designation by arithmetic instructions, the addition or subtraction of 32-Bit data is required.

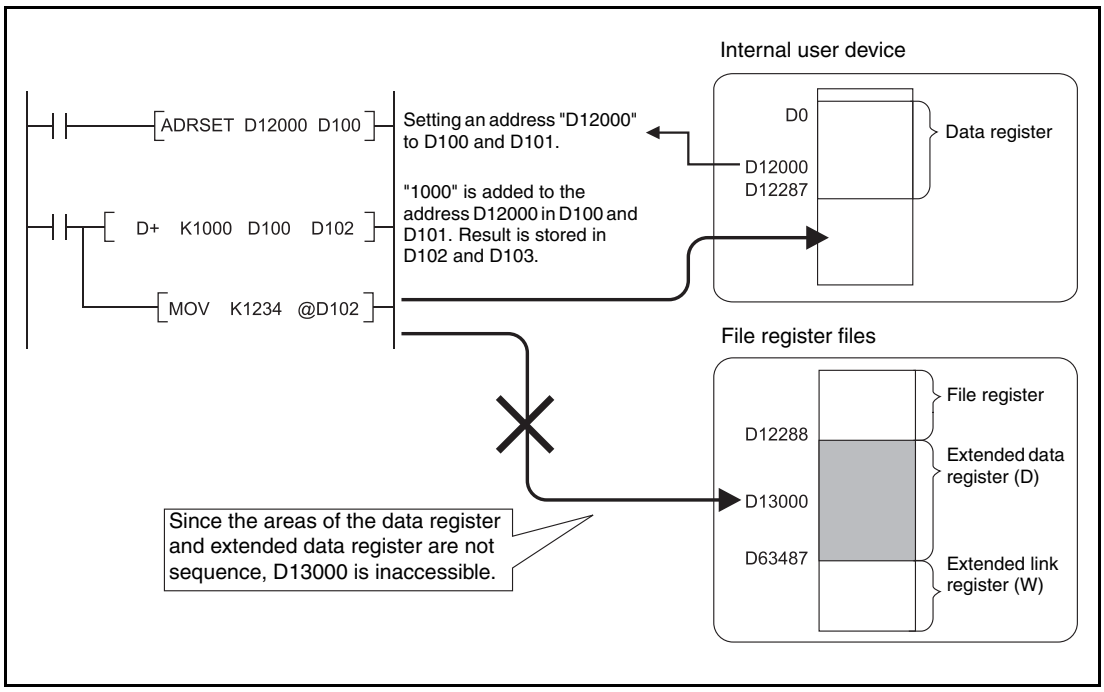
In the following program examples the device which stores the device for indirect designation is incremented and decremented by 32-Bit instructions. By doing so, the address of the device for indirect designation is increased resp. decreased by 1.





Indirect designation of extended data register (D) and extended link register (W)

Indirect designation can be performed in the extended data register (D) and extended link register (W).
Note that when indirect designation is performed to the extended data register (D) and data register (D) in internal device or to the extended link register (W) and link register (W) in internal device, the areas of the internal user device and extended data register (D) or extended link register (W) are not treated as a sequence.



3.8 Reducing instruction processing time

3.8.1 Subset processing

Subset processing is used to place limits on bit devices used by basic instructions and application instructions in order to increase processing speed. However, the instruction symbol does not change.

To shorten scans, run instructions under the conditions indicated below.

Conditions which each device must meet for subset processing

- When using word data

Device	Condition
Bit device	• Drive number and file name of comment file to be switched to or first number of device storing such data. • Only K4 can be designated for digit designation. • Does not perform indexing.
Word device	• Internal user device. • File register (R, ZR [4]) • Multiple CPU shared device [1, 2] • Index register (Z) / Standard device register (Z) [3]
Constants	No limitations

- When using double word data

Device	Condition
Bit device	• Designates a bit device number in a factor of 16. • Only K8 can be designated for digit designation. • Does not perform indexing.
Word device	• Internal user device. • File register (R, ZR [4]) • Multiple CPU shared device [1, 2] • Index register (Z) / Standard device register (Z) [3]
Constants	No limitations

- When using bit data

Device	Condition
Bit device	• Internal user device (indexing possible)
Word device	• Bit specification of internal user device • Bit specification of file register (R, ZR [4]) • Bit specification of multiple CPU shared device [1, 2]

[1] Only for Universal model QCPU

[2] Valid only for the multiple CPU high speed transmission area (from U3En\G10000)
(Excluding the case that indexing is executed for the head I/O number of the CPU module (U3En\G10000))

[3] Applies only to Universal model QCPU and LCPU.

[4] Applies only to Universal model QCPU (excluding Q00UJCPU) and LCPU.

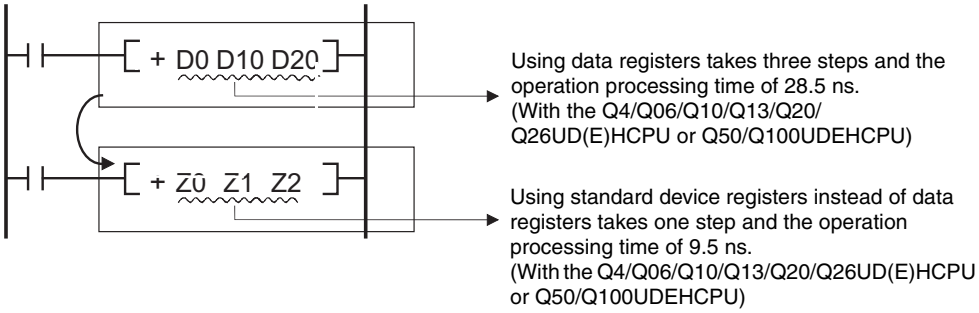
Instructions for which subset processing can be used

Types of Instructions	Instruction Symbols
Contact instructions	LD, LDI, AND, ANI, OR, ORI, LDP, LDF, ANDP, ANDF, ORP, ORF, LDPI, ANDPI, ANDFI, ORPI, ORFI
Output instructions	OUT, SET, RST
Comparison operation instruction	=, <>, <, <=, >, >=, D=, D<>, D<, D<=, D>, D>=
Arithmetic operation	+, –, x, /, INC, DEC, D+, D–, Dx, D/, DINC, DDEC B+, B–, Bx, B/, E+, E–, Ex, E/
Data conversion instructions	BCD, BIN, DBCD, DBIN, FLT, DFLT, INT, DINT
Data transfer instruction	MOV, DMOV, CML, DCML, XCH, DXCH FMOV, BMOV, EMOV
Program branch instruction	CJ, SCJ, JMP
Logic operations	WAND, DAND, WOR, DOR, WXOR, DXOR, WXNR, DXNR
Rotation instruction	RCL, DRCL, RCR, DRCR, ROL, DROL, ROR, DROR
Shift instruction	SFL, DSFL, SFR, DSFR
Data processing instructions	SUM, SEG
Structure creation instructions	FOR, CALL

3.8.2 Operation processing with standard device registers (Z) (Universal model QCPU and LCPU only)

Operation processing time can be reduced with standard device registers (Z).

The following figure shows an example program with standard device registers.



Operation processing time is reduced with the instructions that the subset processing is possible.

For the number of steps, refer to section 3.11.

For the operation time for each instruction, refer to Appendix A.

NOTE *Because standard device registers are the same devices as index registers, do not use device numbers of the standard device registers for the index registers.*

3.9 Operation errors

In the following cases operation errors occur:

- If the error conditions described under the topic "Operation Errors" for the individual instructions match, an error code is returned.
- When an intelligent function module device is used, no intelligent function module is installed at the specified I/O number position.
- When an intelligent function module device is used, the specified buffer memory address does not exist.
- If a link device is used, but the corresponding network does not exist.
- If a link device is used, but there is no network module connected to the specified I/O number.
- When a multiple CPU shared device is used, a CPU module is not installed at the head I/O number position of the specified CPU module.
- When a multiple CPU shared device is used, the specified shared memory address does not exist.
- The setting of the device number crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W).
  (Universal model QCPU (excluding Q00UJCPU) and LCPU)

NOTE *When file register is set but a memory card is not installed or when file register is not set, writing/reading to/from file register is as follows:*

- *For the High Performance model QCPU, Process CPU, and Redundant CPU*
  An error does not occur even when writing/reading to/from file register is performed. However, "0H" is stored when reading from file register is performed.
- *For the Universal model QCPU and LCPU*
  The OPERATION ERROR (error code 4101) occurs when writing/reading to/from file register is performed.

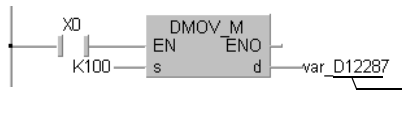
3.9.1 Verification of the device range

Instructions for specified each device, including MOV and DMOV

- For the Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

  If instructions use devices with fixed length (MOV, DMOV, etc.), the device range will not be verified. In those cases where the relevant address range is exceeded the data to be written is written to a vacant device.

  If for example, 12k addresses are designated, there will no error code be returned even if the register address D12287 is exceeded.

Ladder Diagram	Explanation
	D12287 and D12288 are designated in this example but D12288 does not exist. A vacant register will be overwritten with the contents of D12288.

  For an index qualification the device range is not verified either.

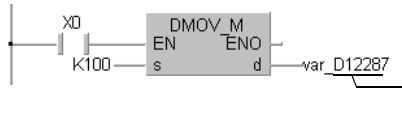
  In cases where the corresponding device range is exceeded as the result of performing indexing, data is written to other devices.

  For the assignment order of internal user devices, refer to page 3-48 ("Character string data") below.

- Universal model QCPU and LCPU

  The device range is checked. When the device number is outside the device range, an operation error occurs.

  For example, when12 k points are assigned to a data register, an error occurs if the device number of the data register exceeds D12287.

Ladder Diagram	Explanation
	When D12287 is specified with the DMOV instruction, the target devices are D12287 and D12288. However, an operation error occurs because D12288 does not exist.

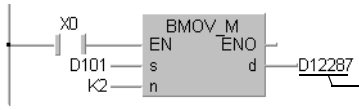
  The device range is checked even though indexing is executed. With changing the settings of the PLC parameter, the device range is not checked.

  For changing the settings of the PLC parameter of the programming tool, refer to the User's Manual of the corresponding programming tool.

Instructions for a block of devices, including BMOV and FMOV

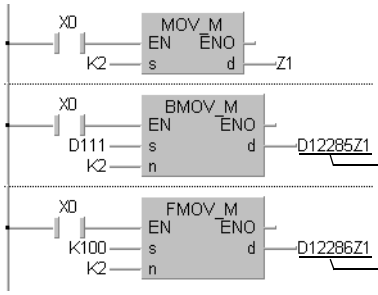
- For the Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

  If instructions use devices with variable length, the device range is verified (BMOV, FMOV, and other instructions that designate initial addresses). In those cases where the relevant address range is exceeded an error code is returned.
  If for example, 12k addresses are designated, the error code is returned after the register address D12287 is exceeded.

Ladder Diagram	Explanation
	D12287 and D12288 are designated in this example. However, D12288 does not exist and an error code is returned.

The device range is verified for an index qualification too.

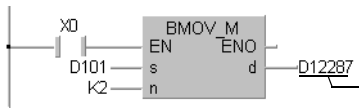
There is no error code returned, if the initial device number exceeds the address range.

Ladder Diagram	Explanation
	D12287 and D12288 are designated in this example. However, D12288 does not exist and an error code is returned. The initial device number D12289 exceeds the relevant range. The data are stored from the register W0 onwards without returning an error code.

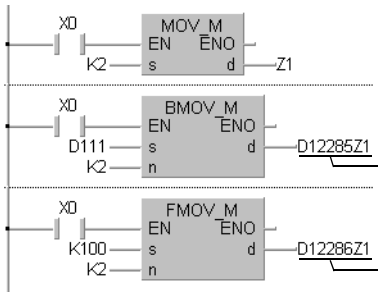
- Universal model QCPU and LCPU

  The device range is checked. In those cases where the relevant address range is exceeded an error code is returned.

  If for example, 12k addresses are designated, the error code is only returned after the register address D12287 is exceeded.

Ladder Diagram	Explanation
	D12287 and D12288 are designated in this example. However, D12288 does not exist and an error code is returned.

The device range is verified for an index qualification too. An error occurs when the head device number of the devices with indexing exceeds the device range.

Ladder Diagram	Explanation
	D12287 and D12288 are designated in this example. However, D12288 does not exist and an error code is returned. The initial device number D12289 exceeds the relevant range. An operation error occurs.

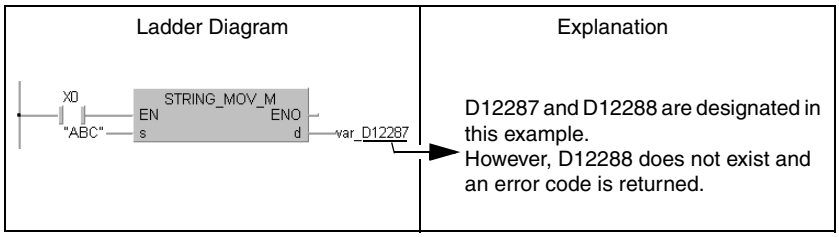
With changing the settings of the PLC parameter, the device range is not checked.

For changing the settings of the PLC parameter of the programming tool, refer to the User's Manual of the corresponding programming tool.

Character string data

Since character strings are of variable lengths the device range is verified. In cases where the corresponding device range is exceeded, an error code is returned.

If for example, 12k addresses are designated, there will no error code be returned until the register address D12287 is exceeded.

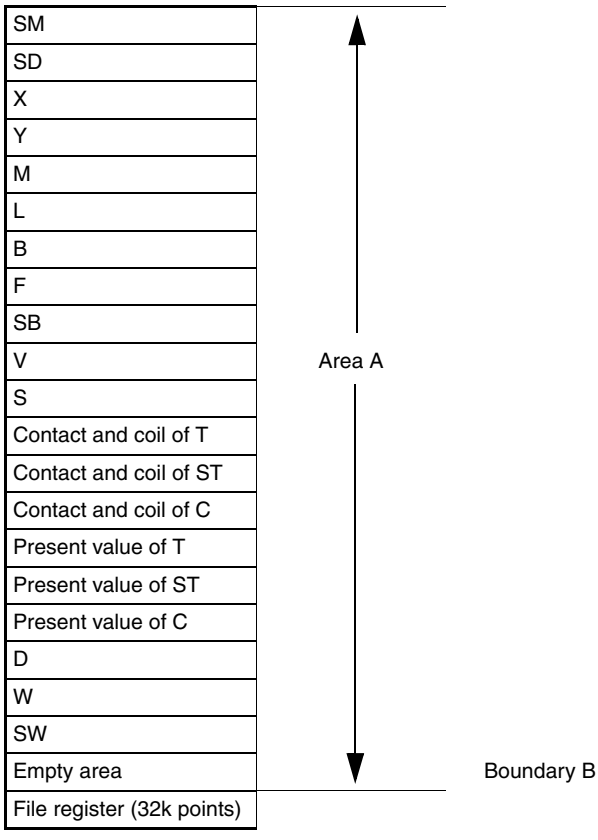


However, with the Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU, when indexing is executed and the head device number is outside the device range, no error occurs and the other devices are accessed.

When performing the following access in Universal model QCPU or LCPU, an error (error code 4101) occurs.

① Access crossing the boundary of devices caused by indexing (range of A area)

The allocation order of individual devices is shown below:



② Access crossing the boundary of file registers caused by indexing

③ Access to file registers (R, ZR) without setting file register files

④ Access to file registers (R, ZR) exceeded the range of file register files

Presetting PLC parameter not to check indexing device range enables the Universal model QCPU not to detect an error in the above accesses from ① to ④. Detecting an error in the above accesses however, depends on the serial No. of Universal model QCPU.

Setting device range in indexing	First 5 digits of serial No. for Universal model QCPU	
	"10021" or lower	"10022" or higher
Set	Detected errors in accesses ① to ④	
Not set	Detected errors in accesses ② to ④	Not detected

For changing the settings of the PLC parameter, refer to the User's Manual of the programming tool.

NOTE *When indexing is executed only with Universal model QCPU or LCPU, devices between internal user devices (SW) and file registers (R) cannot be skipped. (Error code 4101)*

Index qualification of the direct output (DY)

The device range is verified for an index qualification of the direct output (DY).

Precautions for using the extended data register (D) or extended link register (W) (for the Universal model QCPU (except Q00UJCPU), and LCPU)

With the following specification methods, data cannot be specified crossing over the boundary of the internal user device and extended data register (D) or extended link register (W). Doing so causes an "OPERATION ERROR" (Error code 4101).

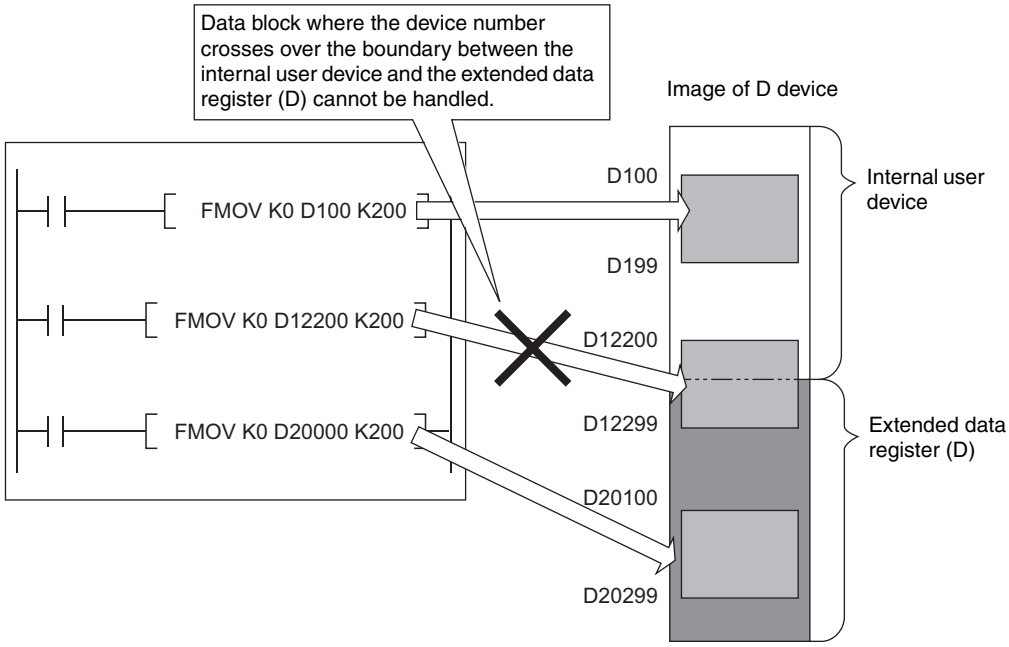
- Index modification
- Indirect designation
- Specification with the instructions that handle data blocks

  Data block indicates the following data:

  – Data used in the instructions, such as FMOV, BMOV, BK+, where multiple words are targeted for operation

  – Control data, composed of two or more words, specified in the instructions, such as SP.FWRITE, SP.FREAD

  – Data whose data type is 32-bit or more (BIN 32-bit, real number, indirect address of the device)



3.9.2 Verification of the device data

Verification of binary data

- If the operation result exceeds the value range, no error code is returned.
  The carry flag in this case is not set.

Verification of BCD data

- Each digit of the BCD values (0 to 9) is verified.
  If one individual digit exceeds the range of 0 to 9 (A to F), an error code is returned.
- If the operation result exceeds the value range, no error code is returned.
  The carry flag in this case is not set.

Verification of floating-point numbers (with single precision floating-point operation instruction)

Operation errors occur in the following cases:

- The absolute value of the floating-point number is 1.0×2^{-127} or lower.
- The absolute valuse of the floating-point number is 1.0×2^{128} or higher.

Verification of floating-point numbers (with double precision floating-point operation instruction)

Operation errors occur in the following cases:

- The absolute value of the floating-point number is 1.0×2^{-1023} or lower.
- The absolute value of the floating-point number is 1.0×2^{1024} or higher.

Verification of character strings

The device data are not verified.

3.9.3 Buffer memory access

For accessing buffer memories, using instructions with intelligent function module devices (from Un\G0) is recommended.

3.9.4 Multiple CPU shared memory access

For accessing multiple CPU shared memories, using instructions with multiple CPU shared devices (from U3En\G10000) is recommended.

3.10 Execution conditions of the instructions

3.10.1 Execution condition

There are 4 different types of execution conditions for the instructions:

- Non-conditional execution

  The instructions are executed regardless of the signal status of the devices.
  Example: LD X0, OUT Y10

- Execution at ON

  The instructions are executed as long as the execution instruction is set.
  Example: MOV, FROM

- Execution at leading edge

  The instructions are executed at leading edge (signal status changes from 0 to 1) from the execution condition.
  Example: PLS, MOVP

- Execution at trailing edge

  The instructions are executed at trailing edge (signal status changes from 1 to 0) from the execution condition.
  Example: PLF

The vast majority of instructions are of the following two types:

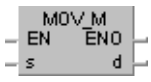
– Execution condition set ON

– Execution at leading edge from the execution condition

Execution condition set ON:

The instruction is executed as long as the execution instruction is set. Such instructions are not particularly indicated.

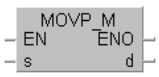
Example: MOV_M/ MOV

Ladder Diagram	MELSEC Instruction List
	

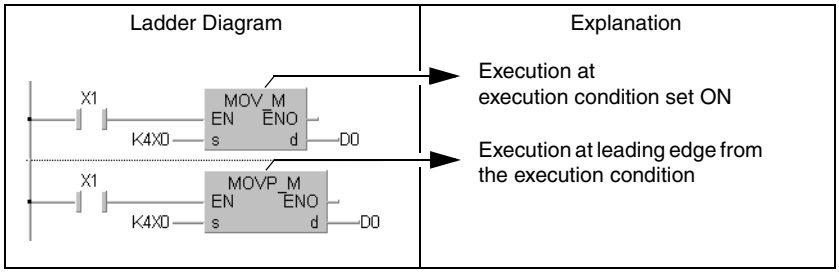
Execution at leading edge from the execution condition:

When judging the leading edge from the execution condition the instruction is executed only if the signal state changes from 0 to 1.

Example: MOVP_M/ MOVP

Ladder Diagram	MELSEC Instruction List
	

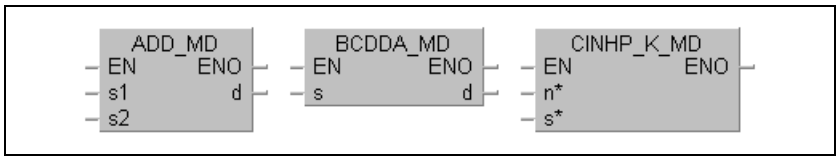
The following example shows the execution of the MOV instruction with the execution condition set ON and the execution at leading edge from the execution condition:



3.10.2 EN input and ENO output

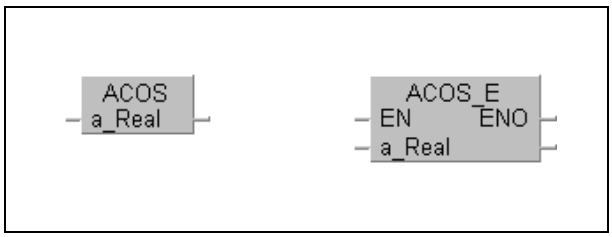
All instructions described in this manual are provided in the manufacturer library of the GX IEC Developer. These instructions in addition to the input and output variables provide an EN input and an ENO output.

The figure below shows several MELSEC instructions from the GX IEC Developer manufacturer library:



In the IEC standard library nearly all instructions appear twice. They just differ in the suffix "_E". These instructions provide an EN input and an ENO output.

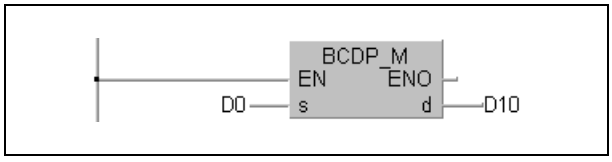
The figure below shows two IEC instructions from the standard library of the GX IEC Developer:



The following examples show the differing execution of the instruction with and without EN inputs and ENO outputs.

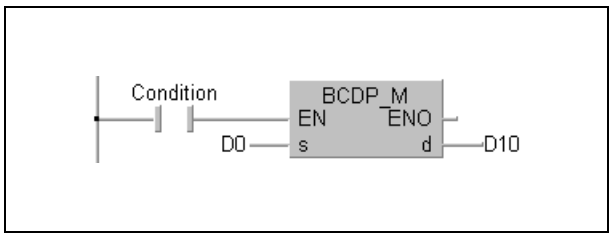
Example 1: Without additional connection

Without additional connection the execution condition of the instruction is permanently set.



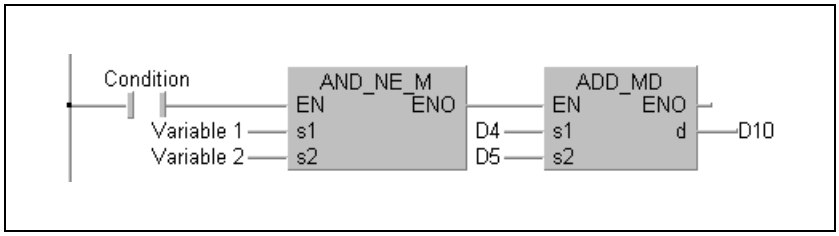
Example 2: Connection with a contact

If the EN input is connected with a contact, the instruction is executed if the condition is matched.



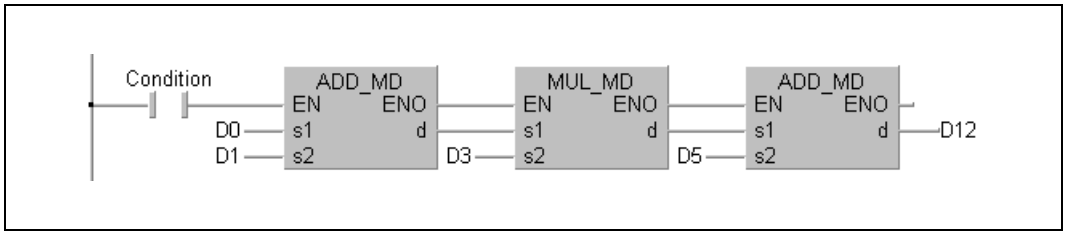
Example 3: Connection with an operation result

If the boolean result of an arithmetic operation is connected to the EN input, the instruction is only executed, if the result of the arithmetic operation is TRUE.



Example 4: Connection with the preceding instruction

If the EN input is connected to the ENO output of the preceding instruction, the instructions are only executed, if the condition is matched.



NOTE *The ENO output must not compulsorily be connected. The signal at the EN input is looped-through to the ENO output. If the EN input is "TRUE", the ENO output is "TRUE" as well.*

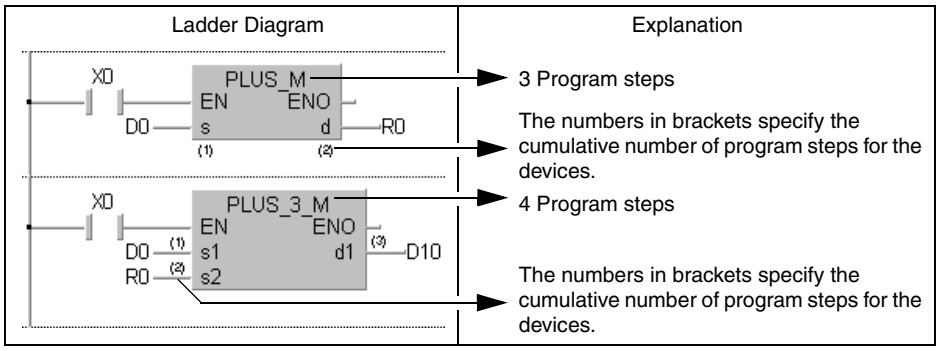
3.11 Number of program steps

In order not to exceed the required memory capacity in the internal memory and ROM or RAM memory of the memory cards a calculation of the total number of steps in a program is required. In the following sections the calculation of steps for the instructions is described.

Counting the number of basic steps

The number of steps for an instruction depends on the number of basic steps. Most of the instructions for their execution only require a number of basic steps. The number of basic steps depends on the number of used devices plus 1.

The example below shows the calculation of the number of basic steps for the PLUS instruction:



Conditions for increasing the number of steps

The number of steps is increased over the number of basic steps in cases where a device is used that is designated indirectly or for which the number of steps is increased.

- When device is designated indirectly

  In cases where indirect designation is done by @□, the number of steps is increased 1 step over the number of basic steps.

  For example, when a 3-step MOV instruction is designated indirectly (example: MOV K4X0 @D0), one step is added and the instruction becomes 4 steps.

- Devices with additional steps (Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU):

Devices increasing the Number of Steps	Added Steps	Example
Devices of special function modules	1	MOV U4\G10 D0
Multiple CPU shared device		MOV U3E1\G0 D0
Link devices		MOV J3\B20 D0
Index register		MOV Z0 D0
File registers addressed in series		MOV ZR123 D0
32-bit constants		DMOV K123 D0
Floating point number as constants		EMOV E0.1 D0
Character strings	For an odd number: (number of characters + 1)/2 For an even number: Number of characters/2	$MOV „123“ D0

- Devices with additional steps (Universal model QCPU(except Q00UJCPU) and LCPU)
  - Instructions applicable to subset processing

    The following table shows steps depending on the devices.

Instruction Symbols	Devices With Additional Steps	Added Steps (Number of Instruction Steps)	Basic Number of Steps
LD, LDI, AND, ANI, OR, ORI, LDP, LDF, ANDP, ANDF, ORP, ORF	Serial number access format file register, Extended data register (D), Extended link register (W)	1(2)	1
	Multiple CPU shared device 3)		
LDPI, LDFI	Serial number access format file register, Extended data register (D), Extended link register (W)	1(4)	3
	Multiple CPU shared device 3)		
ANDPI, ANDFI, ORPI, ORFI	Serial number access format file register, Extended data register (D), Extended link register (W)	1(5)	4
	Multiple CPU shared device 3)		
SET	Serial number access format file register Extended data register (D), Extended link register (W)	1(2)	1
	Multiple CPU shared device 3)		
OUT	Timer/Counter	3(4)	1
	Serial number access format file register Extended data register (D), Extended link register (W)	1(2)	
	Multiple CPU shared device 3)		
RST (bit device)	Serial number access format file register, Extended data register (D), Extended link register (W)	1(2)	1
	Multiple CPU shared device 3)		
RST (word device)	Timer/Counter (Bit/word device)	2(4)	2
	Serial number access format file register, Extended data register (D), Extended link register (W)	1(3)	
	Multiple CPU shared device 3)	1(3)	
LD=, LD<>, LD<, LD<=, LD>, LD>=, AND=, AND<>, AND<, AND<=, AND>, AND>=, OR=, OR<>, OR<, OR<=, OR>, OR>=	Standard device register 2)	-1	3
	Serial number access format file register, Extended data register (D), Extended link register (W)	1	
	Multiple CPU shared device 3)		

Instruction Symbols	Devices With Additional Steps	Added Steps (Number of Instruction Steps)	Basic Number of Steps
LDD=, LDD<>, LDD<, LDD<=, LDD>, LDD>=, ANDD=, ANDD<>, ANDD<, ANDD<=, ANDD>, AND>=, ORD=, ORD<>, ORD<, ORD<=, ORD>, ORD>=	Standard device register [2)]	-1	3
	Serial number access format file register, Extended data register (D), Extended link register (W)	1	
	Multiple CPU shared device [3)]		
	Decimal constant, hexadecimal constant, real constant		
+, -, +P, -P, WAND, WOR, WXOR, WXNR, WANDP, WORP, WXORP, WXNRP (2 devices)	Standard device register [2)]	d: -1	3
	Serial number access format file register Extended data register (D), Extended link register (W)	s1: 1, d: 3	
	Multiple CPU shared device [3)]		
D+, D-, D+P, D-P, DAND, DOR, DXOR, DXNR, DANDP, DORP, DXORP, DXNRP (2 devices)	Standard device register [2)]	d: -1	3
	Serial number access format file register, Extended data register (D), Extended link register (W)	s1: 1, d: 3	
	Multiple CPU shared device [3)]		
	Decimal constant, hexadecimal constant, real constant	s1: 1	
+, -, +P, -P, WAND, WOR, WXOR, WXNR, WANDP, WORP, WXORP, WXNRP (3 devices) [1)]	Serial number access format file register, Extended data register (D), Extended link register (W)	s1, s2: 1, d: 2	3
	Multiple CPU shared device [3)]		
D+, D-, D+P, D-P, DAND, DOR, DXOR, DXNR, DANDP, DORP, DXORP, DXNRP (3 devices)[1)]	Serial number access format file register, Extended data register (D), Extended link register (W)	s1, s2:1, d: 2	3
	Multiple CPU shared device [3)]		
	Decimal constant, hexadecimal constant, real constant	s1, s2: 1	
x, xP, /, /P	Serial number access format file register, Extended data register (D), Extended link register (W)	s1, s2:1, d: 2	3
	Multiple CPU shared device [3)]		
Dx, DxP, D/, D/P, Ex, ExP	Serial number access format file register, Extended data register (D), Extended link register (W)	s1, s2:1, d: 2	3
	Multiple CPU shared device [3)]		
	Decimal constant, hexadecimal constant, real constant	s1, s2: 1	
INC, INCP, DEC, DECP, DINC, DINCP, DDEC, DDECP	Index register/Standard device register [2)]	-1	2
	Serial number access format file register, Extended data register (D), Extended link register (W)	3	
	Multiple CPU shared device [3)]	1	

Instruction Symbols	Devices With Additional Steps	Added Steps (Number of Instruction Steps)	Basic Number of Steps
MOV, MOVP	Serial number access format file register, Extended data register (D), Extended link register (W)	1	2
	Multiple CPU shared device [3)]		
DMOV, DMOVP, EMOV, EMOVP	Serial number access format file register, Extended data register (D), Extended link register (W)	1	2
	Multiple CPU shared device [3)]		
	Decimal constant, hexadecimal constant, real constant		
BCD, BCDP, BIN, BINP, FLT, FLTP, CML, CMLP	Serial number access format file register, Extended data register (D), Extended link register (W)	s1: 1, s2: 2	2
	Multiple CPU shared device [3)]		
DBCD, DBCDP, DBIN, DBINP, INT, INTP, DINT, DINTP, DFLT, DFLTP, DCML, DCMLP	Serial number access format file register, Extended data register (D), Extended link register (W)	s1: 1, s2: 2	2
	Multiple CPU shared device [3)]		
	Decimal constant, hexadecimal constant, real constant	s1: 1	

[1] If the same device is used for s1 and s2, the number of basic steps increases by one.

[2] The number of steps decreases with a standard device register.

[3] Not available with LCPU.

When multiple standard device registers are used in an instruction applicable to subset processing, the number of steps decreases.

The following table shows the number of steps for each instruction.

Instruction Symbols	Locations Where Standard Device Register Is Used	Added Steps (Number of Instruction Steps)	Basic Number of Steps
LD=, LD<>, LD<, LD<=, LD>, LD>=, AND=, AND<>, AND<, AND<=, AND>, AND>=, OR=, OR<>, OR<, OR<=, OR>, OR>=, LDD=, LDD<>, LDD<, LDD<=, LDD>, LDD>=, ANDD=, ANDD<>, ANDD<, ANDD<=, ANDD>, AND>=, ORD=, ORD<>, ORD<, ORD<=, ORD>, ORD>=	s1 and s2	–2(1)	3
+, -, +P, -P, D+, D-, D+P, D-P, WAND, WOR, WXOR, WXNR, DAND, DOR, DXOR, DXNR, WANDP, WORP, WXORP, WXNRP, DANDP, DORP, DXORP, DXNRP (2 devices)	s1 and d	–2(1)	3
+, -, +P, -P, D+, D-, D+P, D-P, WAND, WOR, WXOR, WXNR, DAND, DOR, DXOR, DXNR, WANDP, WORP, WXORP, WXNRP, DANDP, DORP, DXORP, DXNRP (3 devices) [1)]	s1, s2 and d	–2(1)	3
	s1, or s2 and d	–1(2)	
	s1 and s2 (only when that device that the number of steps does not increase is specified for d)	±0(3)	
	s1 and s2 (only when a serial number access format file register is specified for d)	+2(5)	
x, xP, /, /P	s1, s2 and d	–2(1)	3
	s1, or s2 and d	–1(2)	
Dx, DxP, D/, D/P, Ex, ExP	s1, s2 and d	–2(1)	3
	s1, or s2 and d	–1(2)	
	s1 and s2 (only when that device that the number of steps does not increase is specified for d)	±0(3)	
	s1 and s2 (only when a serial number access format file register is specified for d)	+2(5)	
MOV, MOVP, DMOV, DMOVP, EMOV, EMOVP	s1 and d	–1(1)	2
BCD, BCDP, BIN, BINP, DBCD, DBCDP, DBIN, DBINP, FLT, FLTP, DFLT, DFLTP, INT, INTP, DINT, DINTP, CML, CMLP, DCML, DCMLP	s1 and d	–1(1)	2

[1] If the same device is used for s1 and d, the number of basic steps increases by one.

– Except Instructions applicable to subset processing

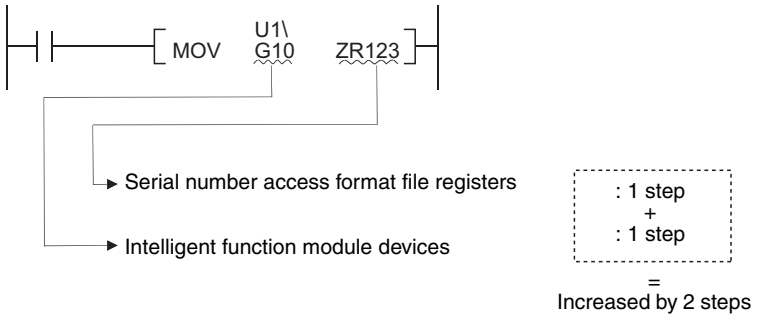
The following table shows steps depending on the devices.

Devices with Additional Steps	Added Steps	Example
Devices of special function modules	1	MOV U4\G10 D0
Multiple CPU shared device		MOV U3E1\G10000 D0
Link direct device		MOV J3\B20 D0
Index register / standard device register		MOV Z0 D0
File registers addressed in series		MOV ZR123 D0
Extended data register (D)		MOV D123
Extended link register (W)		MOV W123
32-bit constants		DMOV K123 D0
Floating point number as constants		EMOV E0.1 D0
Character strings	For an odd number: (number of characters + 1/2) For an even number: Number of characters/2	$MOV „123“ D0

In cases where several of these factors apply the number of steps sums up.

If for example, MOV U1\G10 ZR123 is programmed, 1 step is added for the buffer memory and 1 step for the file register addressed in series, resulting in a total of 2 steps (see the following figure):

Example: MOV
If U1\G10 ZR123 has been designated, a total of 2 steps is added.



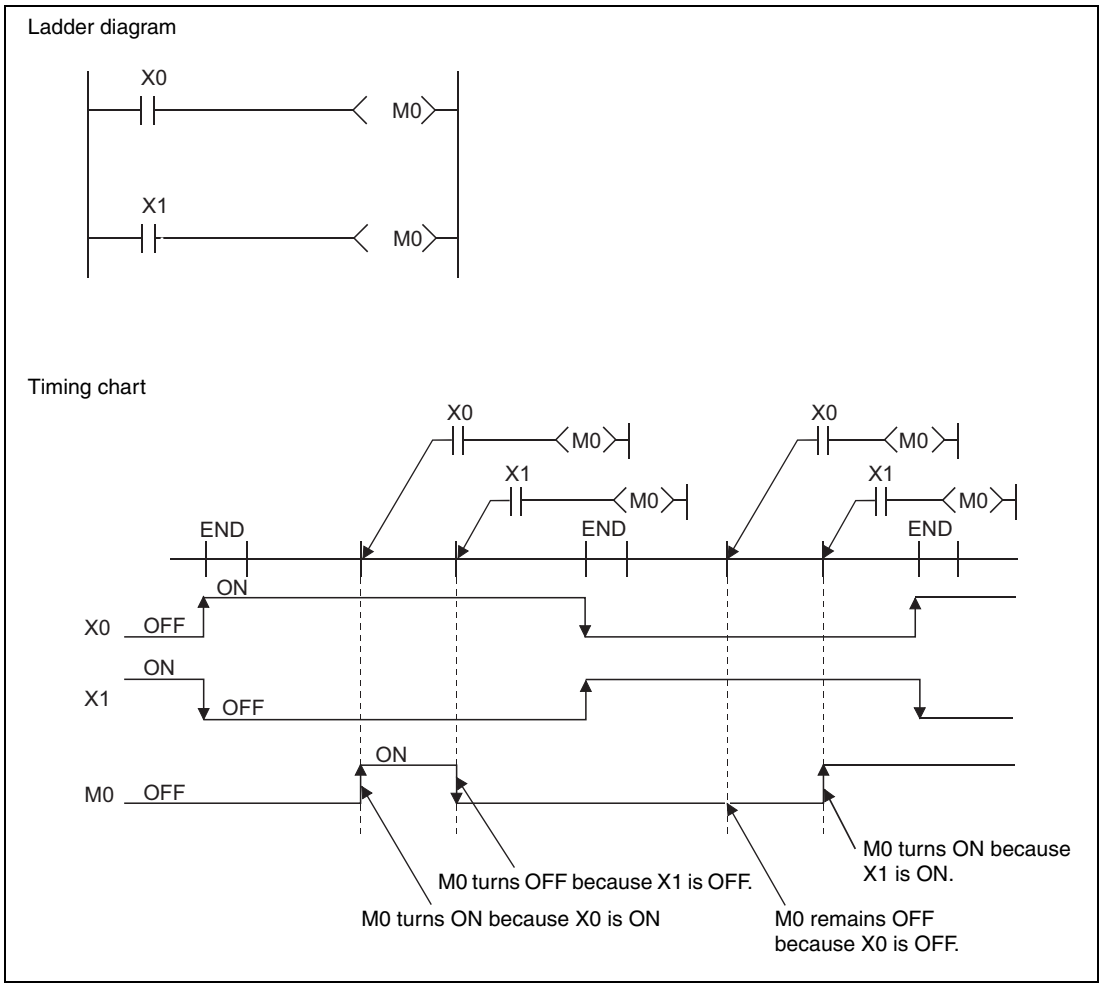
3.12 Multiple Instructions using the same device

This section describes the operation for executing multiple instructions of the OUT, SET/RST, or PLS/PLF that use the same device in one scan.

3.12.1 OUT instructions using the same device

Do not program more than one OUT instruction using the same device in one scan. If the OUT instructions using the same device are programmed in one scan, the specified device will turn ON or OFF every time the OUT instruction is executed, depending on the operation result of the program up to the relevant OUT instruction. Since turning ON or OFF of the device is determined when each OUT instruction is executed, the device may turn ON and OFF repeatedly during one scan.

The following diagrams show an example of a ladder that turns the same internal relay (M0) with inputs X0 and X1 ON and OFF.



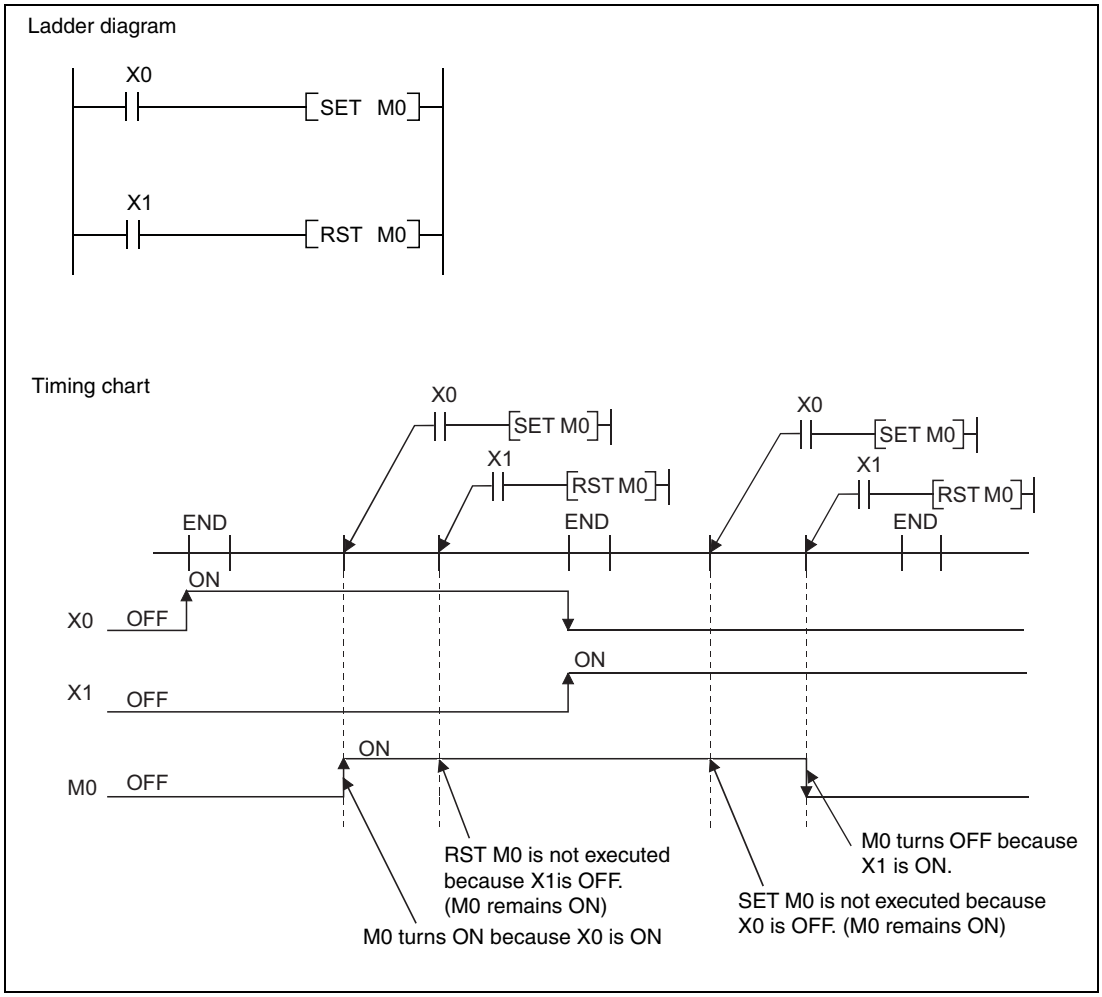
With the refresh type CPU module, when the output (Y) is specified by the OUT instruction, the ON/OFF status of the last OUT instruction of the scan will be output.

3.12.2 SET/RST instructions using the same device

The SET instruction turns ON the specified device when the execution command is ON and performs nothing when the execution command is OFF. For this reason, when the SET instructions using the same device are executed two or more times in one scan, the specified device will be ON if any one of the execution commands is ON.

The RST instruction turns OFF the specified device when the execution command is ON and performs nothing when the execution command is OFF. For this reason, when the RST instructions using the same device are executed two or more times in one scan, the specified device will be OFF if any one of the execution commands is ON.

When the SET instruction and RST instruction using the same device are programmed in one scan, the SET instruction turns ON the specified device when the SET execution command is ON and the RST instruction turns OFF the specified device when the RST execution command is ON. When both the SET and RST execution commands are OFF, the ON/OFF status of the specified device will not be changed.



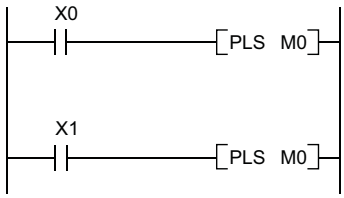
When using a refresh type CPU module and specifying output (Y) in the SET/RST instruction, the ON/OFF status of the device at the execution of the last instruction in the scan is returned as the output (Y).

3.12.3 PLS instructions using the same device

The PLS instruction turns ON the specified device when the execution command is turned ON from OFF. It turns OFF the device at any other time (OFF to OFF, ON to ON, or ON to OFF).

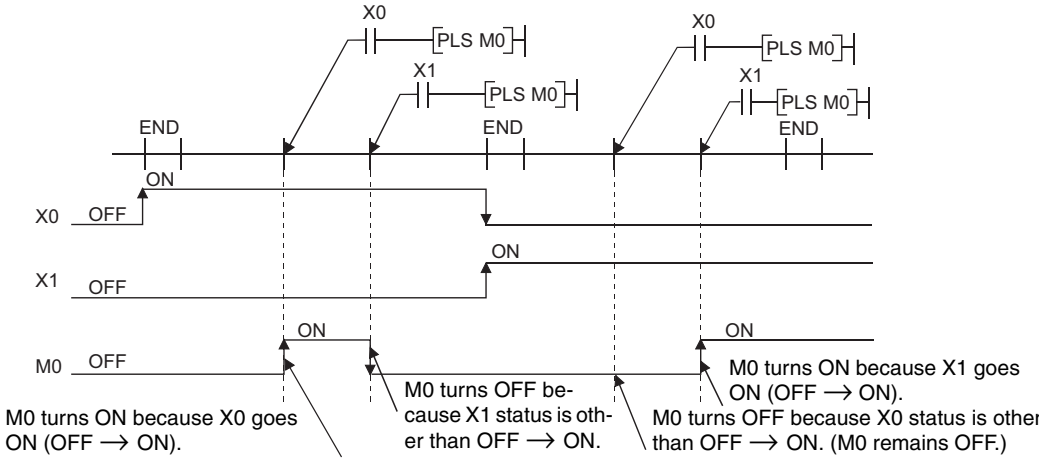
If two or more PLS instructions using the same device are executed in one scan, each instruction turns ON the device when the corresponding execution command is turned ON from OFF and turns OFF the device in other cases. For this reason, if multiple PLS instructions using the same device are executed in a single scan, a device that has been turned ON by the PLS instruction may not be turned ON during one scan.

Ladder diagram

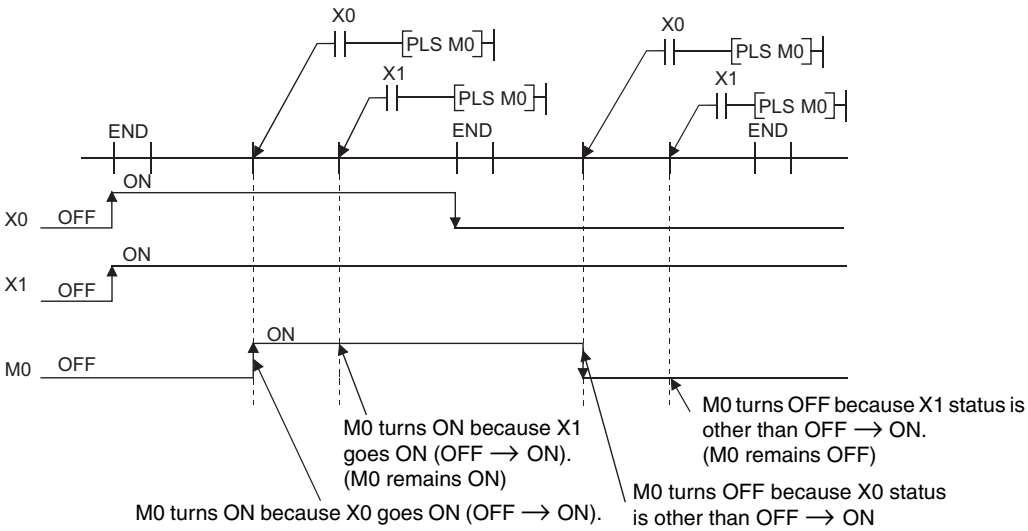


Timing chart

The ON/OFF timing of the X0 and X1 is different.
(The specified device does not turn ON throughout the scan.)



X0 and X1 turn ON from OFF at the same time.



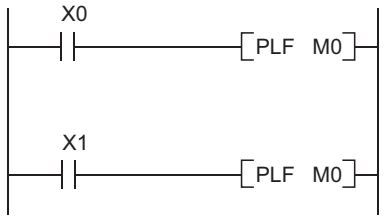
When using a refresh type CPU module and specifying output (Y) in the PLS instructions, the ON/OFF status of the device at the execution of the last PLS instruction in the scan is returned as the output (Y).

3.12.4 PLF instructions using the same device

The PLF instruction turns ON the specified device when the execution command is turned OFF from ON. It turns OFF the device at any other time (OFF to OFF, OFF to ON, or ON to ON).

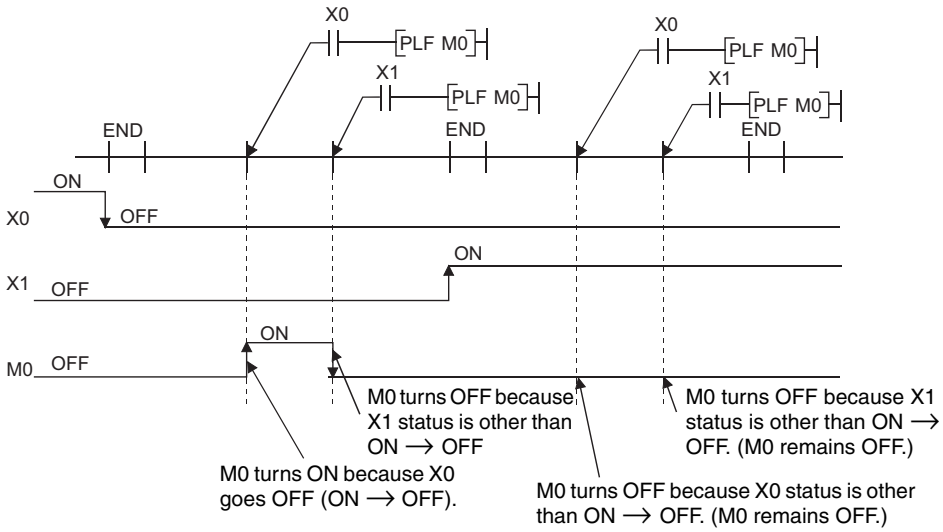
If two or more PLF instructions using the same device are executed in one scan, each instruction turns ON the device when the corresponding execution command is turned OFF from ON and turns OFF the device in other cases. For this reason, if multiple PLF instructions using the same device are executed in a single scan, a device that has been turned ON by the PLF instruction may not be turn ON during one scan.

Ladder diagram

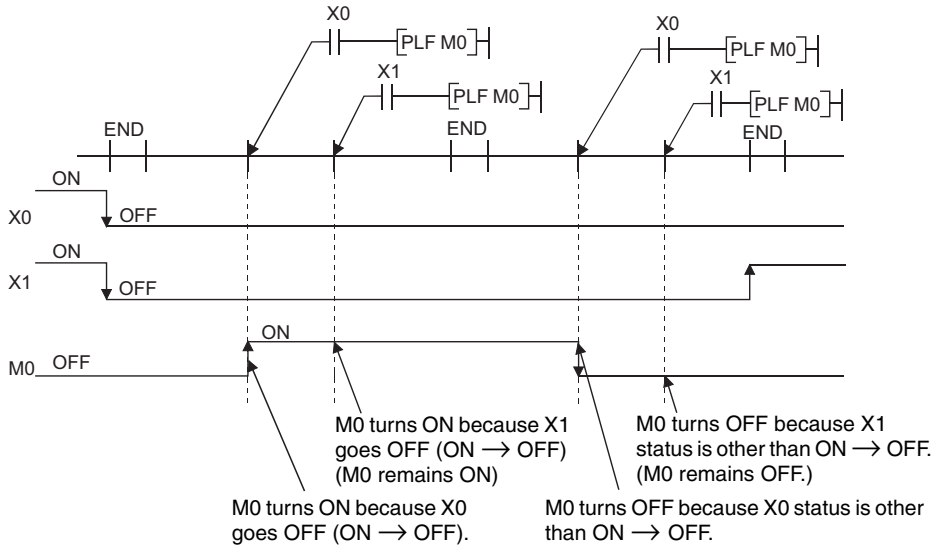


Timing chart

The ON/OFF timing of the X0 and X1 is different.
(The specified device does not turn ON throughout the scan.)



X0 and X1 turn OFF from ON at the same time.



When using a refresh type CPU module and specifying output (Y) in the PLF instructions, the ON/OFF status of the device at the execution of the last PLF instruction in the scan is returned as the output (Y).

3.13 Precautions for use of file registers

This section explains the precautions for use of the file registers in the QCPU and LCPU.

CPU modules that cannot use file registers

The Q00JCPU and Q00UJCPU cannot use the file registers. When using the file registers, use the CPU module of other than the Q00JCPU and Q00UJCPU.

Setting of file registers to be used

When using the file registers, the file registers to be used must be set with the PLC parameter or QDRSET instruction. (The PLC parameters of the Q00CPU, Q01CPU and LCPU need not be set since they are preset to "Use file register". QDRSET instructions are not available with LCPU.)

If the file registers to be used have not been set, normal operation cannot be performed with the instructions that use the file registers.

NOTE *Even when file registers to be used are not set in the PLC parameter, a program that uses file registers can be created.*
For the CPU module other than the Universal model QCPU and LCPU, an error does not occur when that program is written to the CPU module.
However, note that the correct data cannot be written/read to/from the file register.
For the Universal model QCPU and LCPU, an error occurs if the program where file registers are used is executed.

Securing of file register area

- High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU

  When using file registers, register the file registers to the standard RAM/memory card to secure the file register area.

- Basic Model QCPU (except Q00JCPU)

  The file register area has been secured in the standard RAM beforehand. The user need not secure the file register area.

- LCPU

  To use the file register, secure a file register area by registering the file register in standard RAM.

The following table indicates the memories that can use the file registers in each CPU module.

Memory	High Performance model QCPU / Process CPU / Redundant CPU / Universal model QCPU (except Q00UJCPU)	Basic Model QCPU (except Q00JCPU) / LCPU
Standard RAM	●	●
Memory card [1)]	● [2)]	○

● Can be registered
○ Cannot be registered

[1] When the flash memory is used, only read from the file registers can be performed. (Write to the flash ROM cannot be performed.)

[2] Unusable for the Q00UCPU and Q01UCPU.

NOTE *For the file register setting method and file register area securing method, refer to User's Manual (Functions Explanation, Program Fundamentals) for the CPU module used.*

Designation of file register number in excess of the registered number of points

- Basic Model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

  An error will not occur if data are written or read to or from the file registers that have numbers greater than the registered number of points. However, note that the read/write of correct data to/from the file registers cannot be performed.

- Universal model QCPU and LCPU

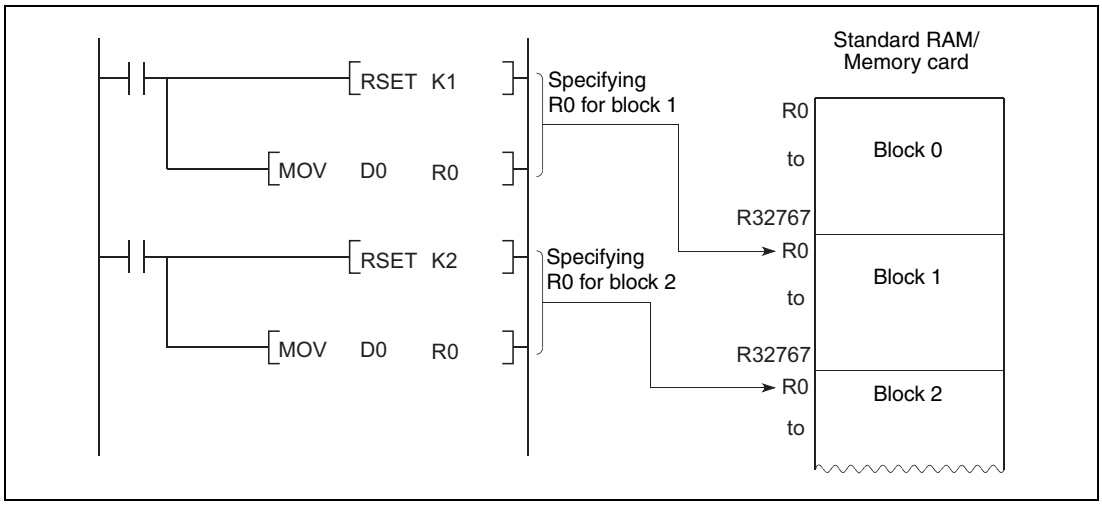
  When data are written to or read from the file registers that are not registered, an error occurs. (Error code: 4101)

File register specifying method

There are the block switching method and serial number access method to specify the file registers.

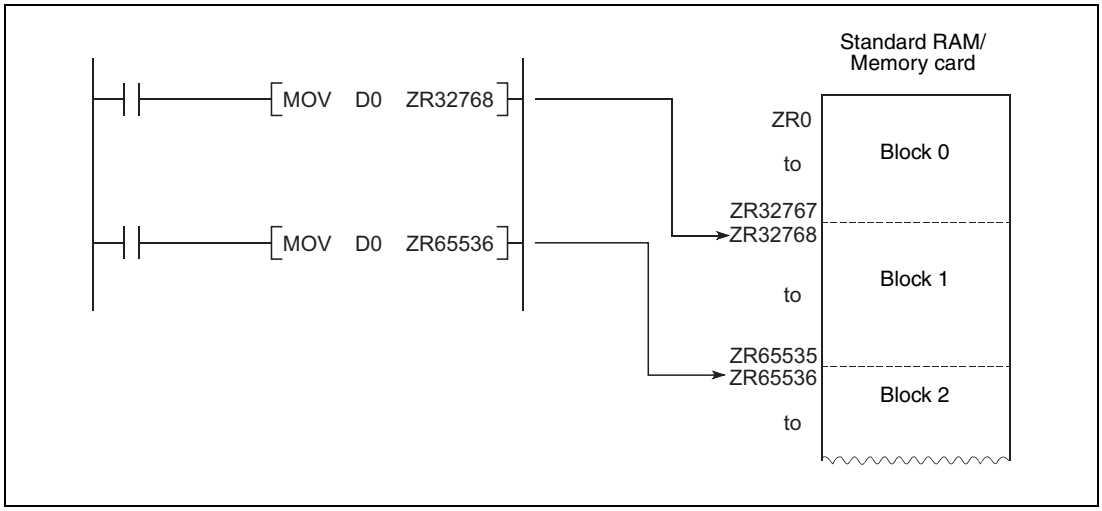
- Block switching method

  In the block switching method, specify the number of used file register points in units of 32k points (one block). For file registers of 32k points or more, specify the file registers by switching the block No. to be used with the RSET instruction. Specify each block as R0 to R32767.



● Serial number access method

In the serial number access method, specify the file registers beyond 32k points with consecutive device numbers. The file registers of multiple blocks can be used as consecutive file registers. Use "ZR" as the device name.



Settings and restrictions when refreshing file registers

● Settings

The settings of refresh devices are as follows.

– Refresh settings for CC-Link IE controller network (Cannot be set on LCPU.)

– Refresh settings for MELSECNET/H (Cannot be set on LCPU.)

– Refresh settings for CC-Link

– Auto refresh settings for the intelligent function module

– Auto refresh settings for the multiple CPU system (Cannot be set on LCPU.)

● Restrictions

The restrictions when specifying file registers to refresh devices are as follows.

– On QCPU, Refresh cannot be performed correctly if the use of file register which has the same name as the program is specified by the PLC parameter. When the file register which has the same name as the program is used, refresh is performed to the data of the file register having the same name as the program that is set at the last number in the [Program] tab page of PLC parameter.

To read/write the refresh data, specify the file register to the refresh device after switching the file register to the corresponding one with the QDRSET instruction.

– Refresh cannot be performed correctly if the file name of file register or the drive number is changed by the QDRSET instruction. (QDRSET instructions are not available with LCPU.)

If the file name of file register or the drive number is changed by the QDRSET instruction, link refresh is performed to the data of the setting file at the time of the END instruction execution.

To read/write the refresh data, specify the file register of the setting file at the time of the END instruction execution.

If the drive number is changed by the QDRSET instruction when "ZR" is specified for the device in the CPU modules other than the Universal model QCPU, an error (LINK PARA ERROR (3101)) occurs. (Note that an error does not occur when "R" is specified for the device.)

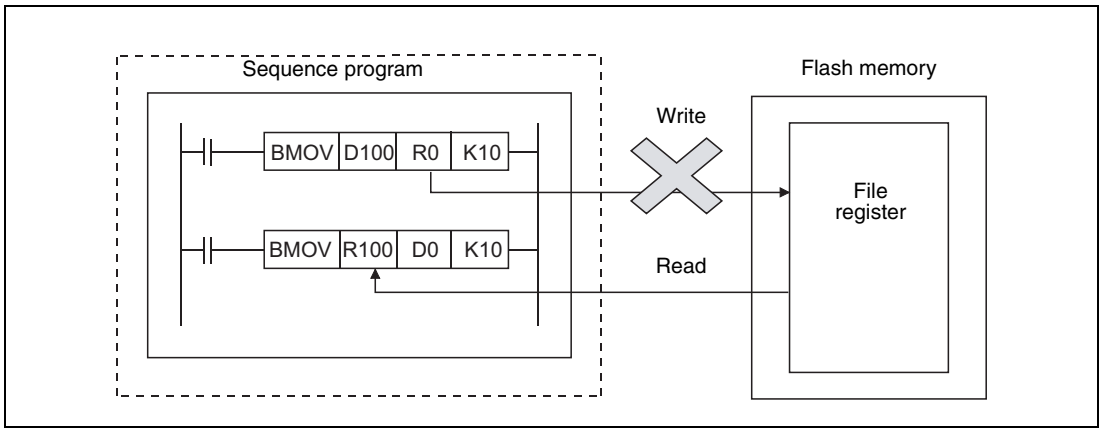
– When a block number is switched by the RSET instruction, refresh is performed to the data of the file register (R) in the switched block number.

When a block number is switched by the RSET instruction, refresh is performed to the data of the file register (R) in the block number at the time of the END instruction execution.

To read/write the refresh data, specify the file register of the block number at the time of the END instruction execution.

Precautions when file registers in the flash memory are used

File registers in the flash memory can be only read in a sequence program.
(Write to the flash memory cannot be performed in a sequence program.)



When using the flash memory for the file registers, write data in advance. Using GX Works2, write data to the flash card.

4 Layout and Structure of the Chapters

This chapter gives an introduction to the chapters 5 through 12 and describes the layout and structure of the explanations to the instructions for the MELSEC Sstem Q and L series.

The figure below shows that each of the these chapters starts with a table that lists and comments the structure and subdivision of the instructions described in that chapter.

6 Application Instructions, Part 1

The application instructions, part 1 comprise instructions that process numerical 16-bit and 32-bit data, floating point data, and character string data. Commonly, these basic instructions perform comparison and arithmetic operations.

Instruction	Meaning
Comparison operation instruction	Compares data to data (e.g. =, >, ≥)
Arithmetic operation instruction	Adds, subtracts, multiplies, divides, increments, and decrements BIN and BCD data, floating point data, and BIN block data Links character strings
Data conversion instruction	Converts data types (e.g. BCD -> BIN, BIN -> BCD)
Data transfer instruction	Transmits designated data
Program branch instruction	Program jump commands
Program execution control instruction	Enables and disables program interrupts
Refresh instruction	Refreshes bit devices, links, and I/O interfaces
Other convenient instructions	Count 1- or 2-phase input up or down, teaching timer, special function timer, rotary table near path rotation control, ramp signal, pulse density measurement, fixed cycle pulse output, pulse width modulation, matrix input

Each subdivided topic is described in the following according chapter and illustrated by program examples.

4.1 Overview of the instructions

Each subdivided topic starts with a table that lists all individual instructions described in this section. As the figure below shows, all variations of the instructions are represented in MELSEC and IEC editor notation.

6.1 Comparison Operation Instructions

Comparison operation instructions compare data values (e.g. equal to =, greater than >, less than <). Programming the comparison operation instructions is similar to the corresponding basic instructions:

LD, LDI ⇒ LD=, LDD=

AND, ANI ⇒ AND=, ANDD=

OR, ORI ⇒ OR=, ORD=

Function	MELSEC-Instruction in MELSEC Editor	MELSEC-Instruction in IEC Editor	Function	MELSEC-Instruction in MELSEC Editor	MELSEC-Instruction in IEC Editor
= equal	LD=	LD_EQ_M	≤ less equal	LD<=	LD_LE_M
	AND=	AND_EQ_M		AND<=	AND_LE_M
	OR=	OR_EQ_M		OR<=	OR_LE_M
	LDD=	LDD_EQ_M		LDD<=	LDD_LE_M
	ANDD=	ANDD_EQ_M		ANDD<=	ANDD_LE_M
	ORD=	ORD_EQ_M		ORD<=	ORD_LE_M
	LDE=	LD_EEQ_M		LDE<=	LD_ELE_M
	ANDE=	AND_EEQ_M		ANDE<=	AND_ELE_M
	ORE=	OR_EEQ_M		ORE<=	OR_ELE_M

When using the GX IEC Developer, always choose the IEC instruction when different notations are offered.

4.2 The CPU table

The sections describing the instructions start with a table that indicates each CPU (Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU, LCPU) capable of processing the respective instruction. The capable CPUs are indicated by a black spot.

Data conversion instructions **ENEG, ENEGP**

6.3.12 ENEG, ENEGP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

[1] Basic model QCPU: The serial number (upper five digits) is "04122" or higher.

Any particular processing details of a certain CPU are commented in a footnote (e.g. extended instructions, refer to section 3.3 "Programming of dedicated instructions".

4.3 Devices

The table "Devices" lists all usable devices that can be used for the internal variables (e.g. s1, s2, d).

The devices are not listed separately; only a distinction is drawn whether the instruction is capable of designating bit and/or word devices.

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	—	—	—
n	●	●	●	●	●	●	●	●	—

Whether the instruction supports file register access is indicated in the column "File Register".

The column "MELSECNET/H Direct **J□\□**" specifies whether the instruction supports read/ write operations of bit and/or word data from/to stations connected to the MELSECNET/H. "**J□**" specifies the station number and "**□**" the device number.

The column "Special Function Module **U□\G□**" specifies whether the instruction supports read/ write operations of data from/to the buffer memory of an installed special function module. "**U□**" specifies the head address of the special function module and "**G□**" the buffer memory address.

Whether the instruction can apply an index qualification is indicated in the column "Index Register Zn".

Whether decimal (K) or hexadecimal (H, 16#) constants can be processed by the instruction is indicated in the column "Constant K, H (16#)".

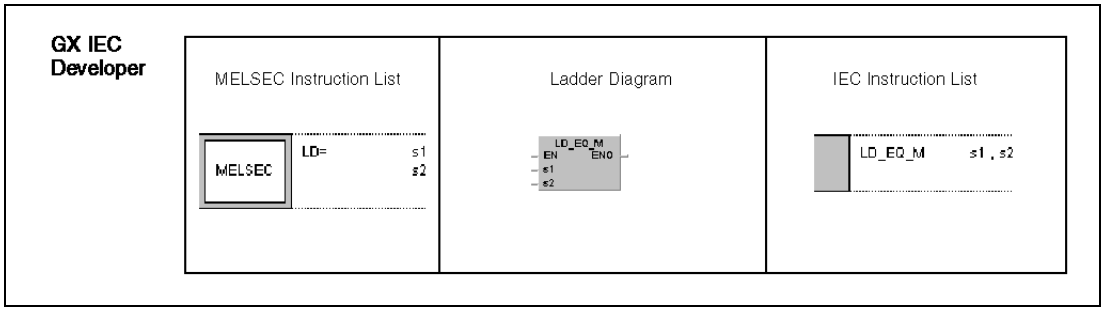
The column "Other" specifies whether the instruction uses any other devices and constants. Any particular details are commented in footnotes below the table.

4.4 Representation format of the instruction

4.4.1 Representation in the GX IEC Developer

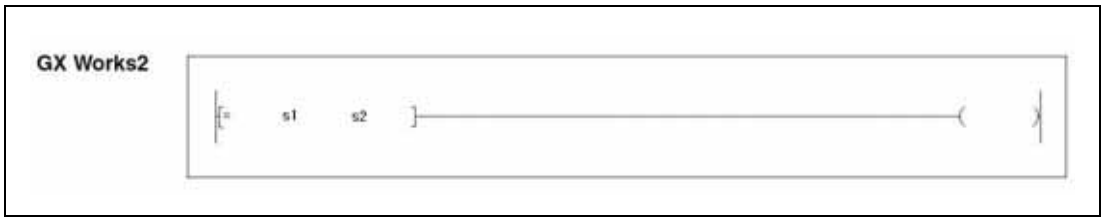
The device tables are followed by the representation format of the instruction in the GX IEC Developer.

The figure below from the left to the right shows the representation of the instruction LD_EQ_M in the MELSEC editor (MELSEC instruction list) and in the IEC editor (ladder diagram and IEC instruction list).



4.4.2 Representation in GX Works2

The representation format for the instruction in the GX IEC Developer is followed by the representation format of the instruction in GX Works2.



4.5 Variables

The table of variables lists all internal variables of the instruction.

Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	s+0: Measurement of table rpm (internal use only).	BIN 16-bit	Array [1..3] of ANY16
	s+1: Number of position.		
	s+2: Number of sector.		
n1	Number of sectors (divisions) on table (2 to 32767).		ANY16
n2	Number of low speed sectors (0 to n1).		ANY16
d	d+0: A-phase input signal.	Bit	Array [1..8] of Bool
	d+1: B-phase input signal.		
	d+2: Zero position detection input signal.		
	d+3: High speed forward output signal (internal use only).		
	d+4: Low speed forward output signal (internal use only).		
	d+5: Stop output signal (internal use only).		
	d+6: High speed reverse output signal (internal use only).		
	d+7: Low speed reverse output signal (internal use only).		

The column "Meaning" describes the functions of the devices and device elements.

The column "Data Type" lists the data types of the devices. Provided that there are differences between the data types of the MELSEC and the IEC editor, these are listed as well. Refer to the sections 3.4 "Programming of variables" and 3.5 "Data types" for further details on variables.

4.6 Functions

The section "Functions" describes the functions of the instruction in detail.

The figure below shows the description of the functions of the LDF/LDP instruction.

Functions

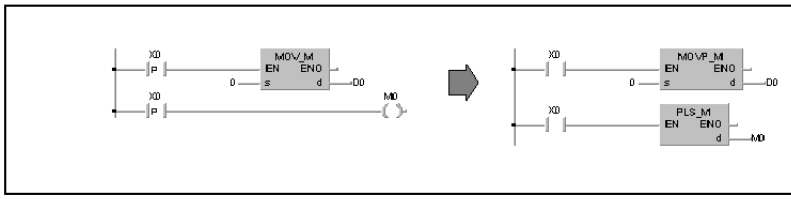
Pulse operation start

LDP **leading edge**

LDF **trailing edge**

Similar to the LD and LDI instructions, these instructions designate contacts specified by bit or word devices. The result of the LDP instruction is 1, if the addressed bit of the device changes from 0 to 1 (leading edge). The result of the LDF instruction is 1, if the addressed bit of the device changes from 1 to 0 (trailing edge). As single instruction the LDP instruction executes the same function as a PLS instruction and with the input condition at leading edge generates a pulse output.

The program example on the left shows a ladder diagram applying an LDP instruction. The example on the right does not apply an LDP instruction.



4.7 Notes

The section "NOTE" points out particular details, errors, and sources of malfunction in the programming of the instruction.

NOTE	*The MEP and MEF instructions will occasionally not function when pulse conversion is applied to contacts that are indexed by a subroutine or by a FOR/NEXT instruction. In this case, the EPG/EGF instruction has to be applied.* *The MEP/MEF instruction operates with the ooperation results immediately prior to the MEP and MEF instructions. For this reason, an AND instruction should be used at the same position. The MEP and MEF instructions cannot be used at the LD or OR position.*

4.8 Operation errors

The description of the operation errors mainly refer to the error code list, see section 13.1 "Error code list".

The figure below shows the operation errors of the DELTA-/DELTAP instruction.

Operation Errors	In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0. • The number of output designated by d exceeds the output range. (Error code: 4101)

4.9 Program examples

The program examples given at the end of each section primarily contain programs for the MELSEC System Q.

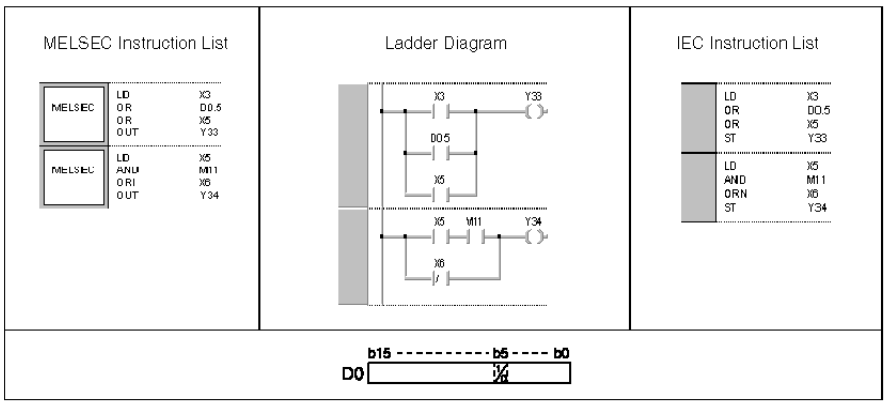
The program examples are programmed in the representation format of the ladder diagram. For a clearer description in many cases graphical illustrations were added.

The figure below shows a program example of the instructions LD, AND, OR, and ORI.

Program Example 1

LD, AND, OR, ORI

The following program shows series and parallel connections of contacts. Bit 5 (b5) in D0 is also read as contact.



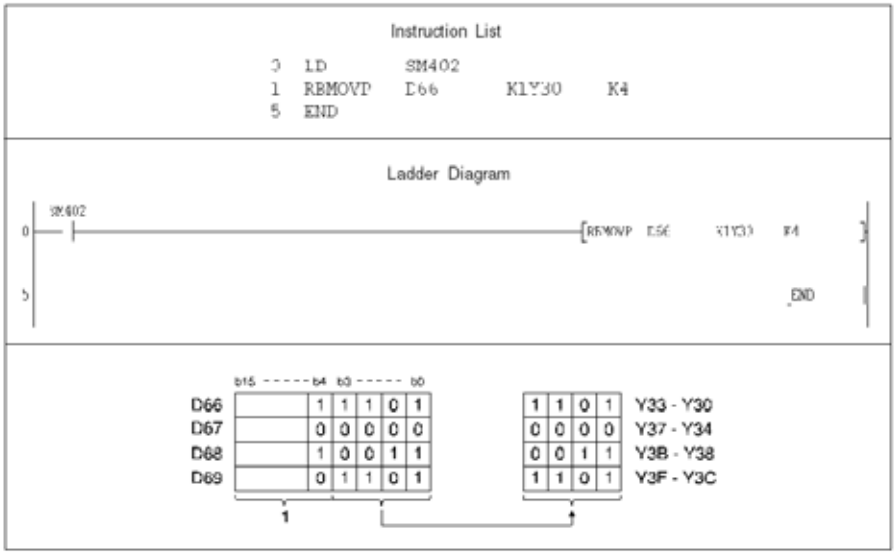
In the following figure a program example for the RBMOVP instruction is shown. The representation of the instructions is that of the GX Works2.

Program Example 1

RBMOVP

The following program transfers the lower four bits (b0 through b3) of data in D66 through D69 to the outputs Y30 through Y3F with the rising edge of SM402. The number of data (4 blocks) is specified by n.

The bit patterns show the structure of bits before and after the transfer.



[1] These bits are ignored.

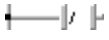
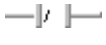
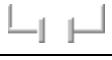
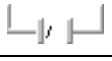
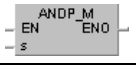
5 Sequence Instructions

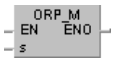
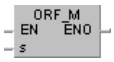
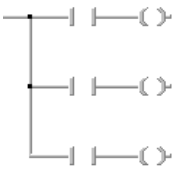
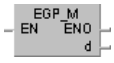
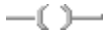
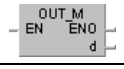
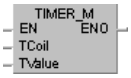
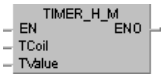
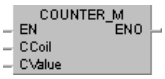
Sequence instructions, besides conventional instructions to program input and output contacts, also include program jump commands, block connection instructions and bit shift instructions, master control, program termination and other instructions. These are the fundamental instructions for programming the MELSEC series.

The following table shows the division of the fundamental instruction set:

Instruction	Meaning
Input instruction	Operation start, series and parallel connection of contacts.
Connection instruction	Series and parallel block connection, storage and processing of operation results, inversion of operation results, conversion of operation results into pulses, setting of edge relays.
Output instruction	Bit devices, counter and timer contacts, output, setting, and resetting of annunciators, setting and resetting of devices, leading edge and trailing edge output, bit device output inversion, generating pulses.
Shift instruction	Shifting bit devices.
Master control instruction	Setting and resetting single parts of a program.
Termination instruction	End of a part of program, end of sequence and routine programs.
Miscellaneous instructions	Sequence program stop, no operation.

NOTE *The following table, besides the MELSEC instructions in the different editors, also contains the according IEC instructions:*

MELSEC Instruction				IEC Instruction in IEC Editor
in MELSEC Editor	in IEC Editor			
	Instruction List	Ladder Diagram		
LD	—		—	LD
LDI	—		—	LDN
AND	—			AND
ANI	—			ANDN
OR	—		—	OR
ORI	—		—	ORN
LDP	LDP_M	—	—	—
LDF	LDF_M	—	—	—
ANDP	ANDP_M	—		—
ANDF	ANDF_M	—		—

MELSEC Instruction				IEC Instruction in IEC Editor
in MELSEC Editor	in IEC Editor			
	Instruction List	Ladder Diagram		
ORP	ORP_M	—		—
ORF	ORF_M	—		—
LDPI				
LDFI				
ANDPI				
ANDFI				
ORPI				
ORFI				
ANB	—		—	AND (...)
ORB	—		—	OR (...)
MPS	MPS_M			—
MRD	MRD_M			—
MPP	MPP_M			—
INV	INV_M			NOT
MEP	MEP_M	—		—
MEF	MEF_M	—		—
EGP	EGP_M	—		—
EGF	EGF_M	—		—
OUT	OUT_M			ST
OUT T	TIMER_M	—		—
OUTH T	TIMER_H_M	—		—
OUT C	COUNTER_M	—		—

MELSEC Instruction				IEC Instruction in IEC Editor
in MELSEC Editor	in IEC Editor			
	Instruction List	Ladder Diagram		
OUT F				
SET	SET_M	—(S)—	SET_M EN ENO d	S
RST	RST_M	—(R)—	RST_M EN ENO d	R
SET F				
RST F				
PLS	PLS_M	—	PLS_M EN ENO d	R_TRIG [1)]
PLF	PLF_M	—	PLF_M EN ENO d	R_TRIG [1)]
FF	FF_M	—	FF_MD EN ENO s	—
DELTA	DELTA_M	—	DELTA_M EN ENO d	—
DELTAP				
SFT	SFT_M	—	SFT_M EN ENO d	SHL/SHR
SFTP				
MC	MC_M	—	MC_M EN ENO n* d	—
MCR	MCR_M	—	MCR_M EN ENO n*	—
FEND	FEND_M	—	FEND_M EN ENO	[2)]
END	END_M	—	END_M EN ENO	[2)]
STOP	STOP_M	—	STOP_M EN ENO	—
NOP	—		—	—
NOPLF				
PAGE				

[1] These are IEC function blocks.

[2] FEND and END are set automatically by GX Works2 and the GX IEC Developer.

5.1 Input instructions

5.1.1 LD, LDI, AND, ANI, OR, ORI

CPU

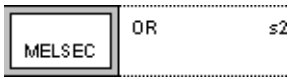
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

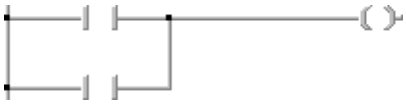
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DX, BL
s	●	●	●	●	●	●	—	—	●

GX IEC Developer

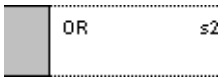
MELSEC Instruction List



Ladder Diagram



IEC Instruction List (IEC Instruction)



GX Works2

Variables

Set Data	Meaning	Data Type
s	Devices used as connections	bit

Functions

Operation start

LD **Load (normally open contact)**

LDI **Load inverse (normally closed contact)**

Every operation starts with an LD (**L**oa**D**) or an LDI ((**L**oa**D I**nverse) instruction.

The LD instruction specifies an NO contact (normally open) and the LDI instruction specifies an NC contact (normally closed). The device designated by the instruction is the input condition (operation result) for the following instruction.

Series connection

AND **of NO contacts**

ANI **of NC contacts**

Contacts are connected in series via an AND instruction as NO contact or via an ANI instruction as NC contact.

The device designated by the instruction sets the operation condition for the following instruction.

Both commands are logical connections and must not be programmed at the beginning of an operation.

Parallel connection

OR **of NO contacts**

ORI **of NC contacts**

Parallel connection of contacts is established via an OR instruction as NO contact or via an ORI instruction as NC contact.

The device designated by the instruction sets the operation condition for the following instruction.

Both commands are logical connections and must not be programmed at the beginning of an operation.

NOTE

The devices designated by the instructions can also be word devices. In this case, the condition of a specified bit is read as contact.

Word devices are designated in hexadecimal code. Bit b11 in D0 for example is designated as D0.0B.

For further information on addressing bits in word devices refer to chapter 3 "Configuration of Instructions".

Program Example 1

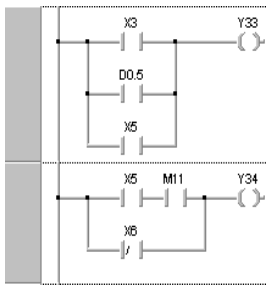
LD, AND, OR, ORI

The following program shows series and parallel connections of contacts. Bit 5 (b5) in D0 is also read as contact.

MELSEC Instruction List

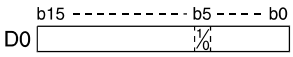
```
MELSEC  LD   X3
        OR   D0.5
        OR   X5
        OUT  Y33
MELSEC  LD   X5
        AND  M11
        ORI  X6
        OUT  Y34
```

Ladder Diagram



IEC Instruction List

```
LD   X3
OR   D0.5
OR   X5
ST   Y33
LD   X5
AND  M11
ORN  X6
ST   Y34
```



Program Example 2

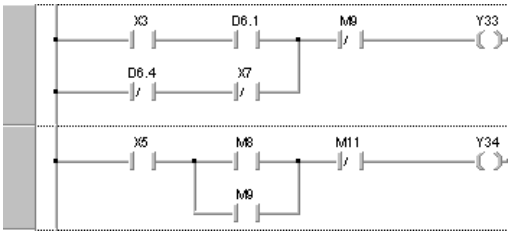
LD, LDI, AND, ANI, OR

The following program shows combined connections. Some contact points are connected via ORB and ANB instructions. Bits (b1 and b4) in D6 are read as contacts.

MELSEC Instruction List

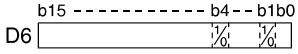
```
MELSEC  LD   X3
        AND  D6.1
        LDI  D6.4
        ANI  X7
        ORB
        ANI  M9
        OUT  Y33
MELSEC  LD   X5
        LD   M8
        OR   M9
        ANB
        ANI  M11
        OUT  Y34
```

Ladder Diagram



IEC Instruction List

```
LD   X3
AND  D6.1
OR(
     D6.4
NOT
ANDN X7
)
ANDN M9
ST   Y33
LD   X5
AND(
     M8
OR   M9
)
ANDN M11
ST   Y34
```



Program Example 3

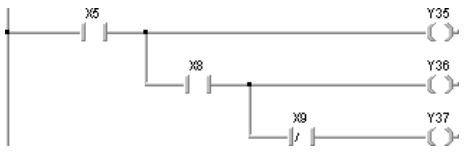
LD, AND, ANI

The following program outputs operation results of devices at Y35 through Y37.

MELSEC Instruction List

```
MELSEC  LD   X5
        OUT  Y35
        AND  X8
        OUT  Y36
        ANI  X9
        OUT  Y37
```

Ladder Diagram



IEC Instruction List

```
LD   X5
ST   Y35
AND  X8
ST   Y36
ANDN X9
ST   Y37
```

5.1.2 LDP, LDF, ANDP, ANDF, ORP, ORF

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

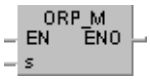
Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DX
s	●	●	●	●	●	●	—	—	●

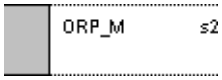
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
s	Devices used as connections	bit

Functions

Pulse operation start

LDP leading edge

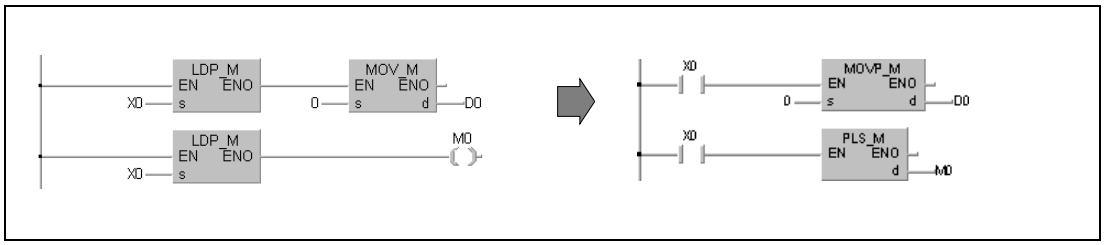
LDF trailing edge

Similar to the LD and LDI instructions, these instructions designate contacts specified by bit or word devices.

The result of the LDP instruction is 1, if the addressed bit of the device changes from 0 to 1 (leading edge).

The result of the LDF instruction is 1, if the addressed bit of the device changes from 1 to 0 (trailing edge). As single instruction the LDP instruction executes the same function as a PLS instruction and with the input condition at leading edge generates a pulse output.

The program example on the left shows a ladder diagram applying an LDP instruction. The example on the right does not apply an LDP instruction.



Pulse series connection

ANDP leading edge

ANDF trailing edge

The ANDP instruction connects a contact in series with a contact specified by a bit or word device. This contact has the condition 1, if the addressed bit of a device changes from 0 to 1.

Using an ANDF instruction the specified contact has the condition 1, if the addressed bit of a device changes from 1 to 0.

Pulse parallel connection

ORP leading edge

ORF trailing edge

The ORP instruction connects a contact in parallel to a contact specified by a bit or word device. This contact has the condition 1, if the addressed bit of a device changes from 0 to 1.

Using an ORF instruction the specified contact has the condition 1, if the addressed bit of a device changes from 1 to 0.

The following table shows the results of an LDP, LDF, ANDP, ORP, ANDF and ORF instruction:

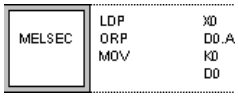
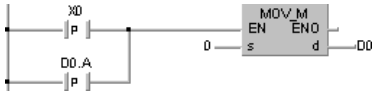
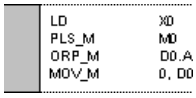
Device specified by LDP/LDF/ANDP/ORP/ANDF/ORF Instruction	Result of LDP/ANDP/ORP Instruction	Result of LDF/ANDF/ORF Instruction
Bit Device/Word Device Bit Designation		
0 → 1	1	0
0	0	
1		
1 → 0		1

NOTE *Word devices are designated in hexadecimal code. Bit b11 in D0 for example is designated as D0.0B.*

Program Example

ORP

With leading edge from X0 or by setting (leading edge) bit 10 (b10) in data register D0, the following program executes a MOV instruction.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

5.1.3 LDPI, LDFI, ANDPI, ANDFI, ORPI, ORFI

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] Availability depending on serial number:
- QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
- QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

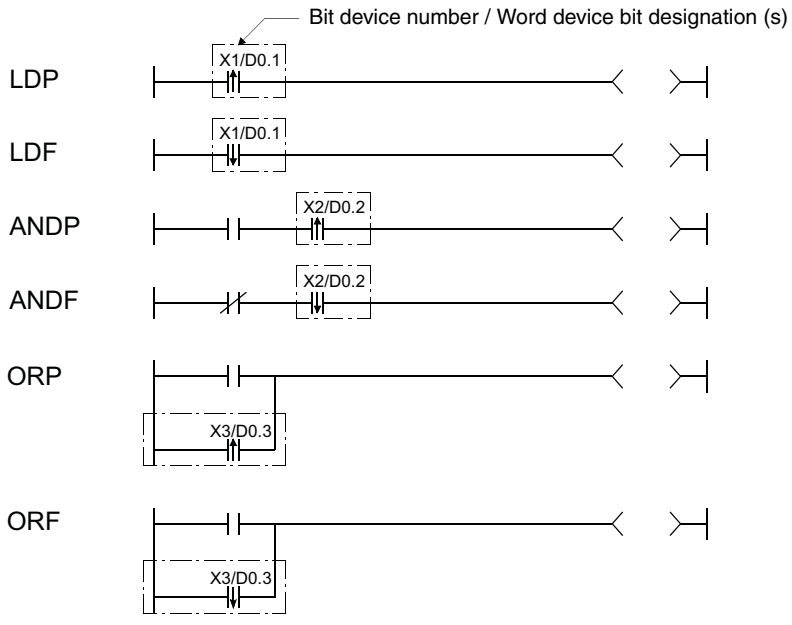
Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DX
s	●	●	●	●	●	●	—	—	●

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s	Devices used as connections	bit

Functions

Pulse NOT operation start

LDPI leading edge

LDFI trailing edge

LDPI is the leading edge pulse NOT operation start instruction that is on only at the leading edge of the specified bit device (when the bit device goes from on to off) or when the bit device is on or off. If a word device has been specified, LDPI is on only when the specified bit is 0, 1, or changes from 1 to 0.

LDFI is the trailing edge pulse NOT operation start instruction that is on only at the trailing edge of the specified bit device (when the bit device goes from off to on) or when the bit device is on or off. If a word device has been specified, LDFI is on only when the specified bit is 0, 1, or changes from 0 to 1.

Pulse NOT series connection

ANDPI leading edge

ANDFI trailing edge

ANDPI is a leading edge pulse NOT series connection, and ANDFI is a trailing edge pulse NOT series connection.

ANDPI and ANDFI execute an AND operation with the previous operation result, and take the resulting value as the operation result.

Pulse NOT parallel connection

ORPI leading edge

ORFI trailing edge

ORPI is a leading edge pulse NOT parallel connection, and ORFI is a trailing edge pulse NOT parallel connection. ORPI and ORFI execute an OR operation with the previous operation result, and take the resulting value as the operation result.

The on or off data used by LDPI/LDFI/ANDPI/ANDFI/ORPI/ORFI are indicated in the table below.

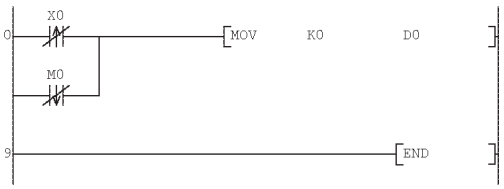
Device specified by LDPI/LDFI/ANDPI/ANDFI/ORPI/ORFI Instruction	Result of LDPI/ANDPI/ORPI Instruction	Result of LDFI/ANDFI/ORFI Instruction
Bit Device/Word Device		
$0 \rightarrow 1$	1	0
0	0	
1		
$1 \rightarrow 0$		1

Program Example 1

LDPI, ORFI

The following program stores 0 into D0 when X0 is on, off, or turns from on to off, or M0 is on, off, or turns from off to on.

Ladder Diagram

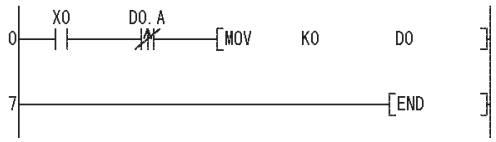


Program Example 2

ANDPI

The following program stores 0 into D0 when X0 is on and b10 (bit 11) of D0 is on, off, or turns from on to off.

Ladder Diagram



5.2 Connection instructions

5.2.1 ANB, ORB

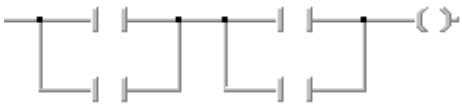
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List (IEC Instruction)
		

GX Works2

Variables

Set Data	Meaning	Data Type
—	—	—

Functions

Ladder block series connection

ANB Block series connection

The ANB instruction (AND block) connects two or more parallel connection blocks in series and supplies an operation result for the following operations.

If more than two blocks are connected in series, after each parallel block an ANB instruction has to be programmed.

The ANB connection is an independent instruction and does not require any device.

Within one program the ANB instruction can be applied any number of times.

If more than two blocks are connected consecutively, the number of ANB instructions is limited to 15 (= 16 blocks). Exceeding this limit results in malfunction.

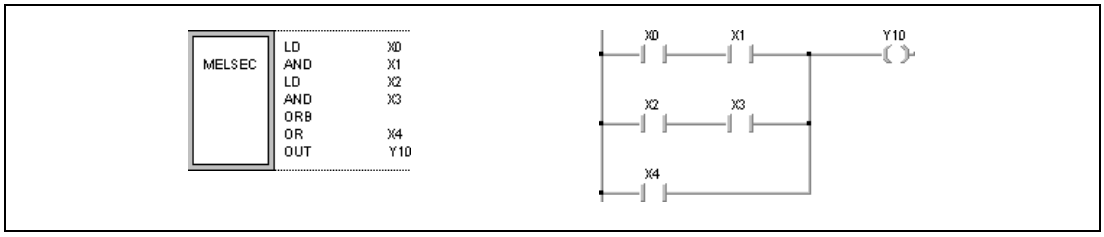
Ladder block parallel connection

ORB Block parallel connection

The ORB instruction (OR block) connects two or more series connection blocks in parallel and supplies an operation result for the following operations.

If more than two blocks are connected in parallel, after each series block an ORB instruction has to be programmed.

For block parallel connections designating one contact only an OR or ORI instruction has to be set.



The ORB connection is an independent instruction and does not require any device.

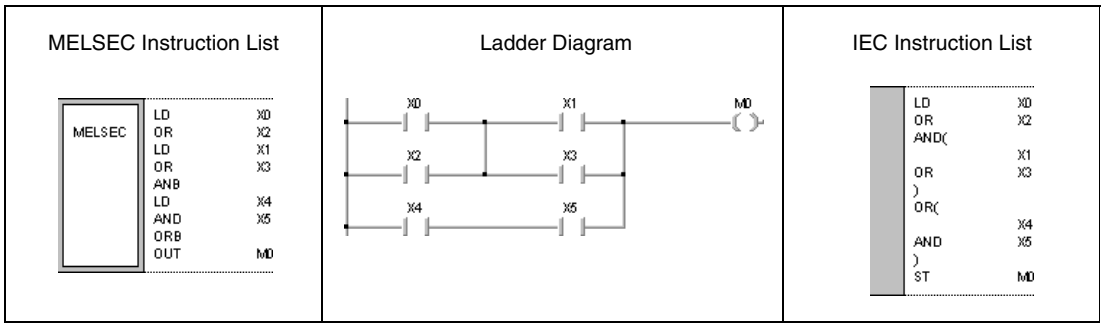
Within one program the ORB instruction can be applied any number of times.

If more than two blocks are connected consecutively, the number of ORB instructions is limited to 15 (= 16 blocks). Exceeding these limits results in malfunction.

Program Example

ANB, ORB

The following program connects the parallel connection block of X0 and X2 in series with the parallel connection block of X1 and X3. The result is connected in parallel with the series connection of X4 an X5.



5.2.2 MPS, MRD, MPP

NOTE *These instructions should not be used within the IEC editors.*

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

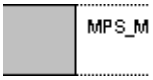
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

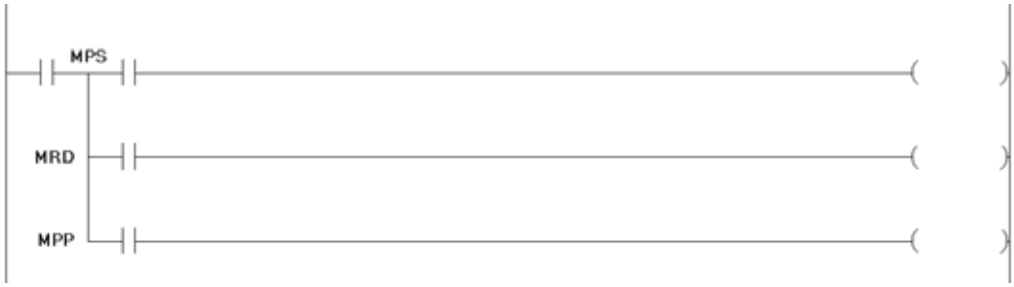
MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
—	—	—

Functions **Operation result processing**

MPS Store operation result (memory push)

The MPS instruction stores the operation result preceding the MPS instruction.

Up to 16 consecutive MPS instructions per network can be programmed.
If the MPP instruction is used during this process, the number of uses calculated for the MPS instruction will be decremented by one.

MRD Read operation result (memory read)

The MRD instruction reads stored operation results via an MPS instruction. The following operation executed depends on the reading result.

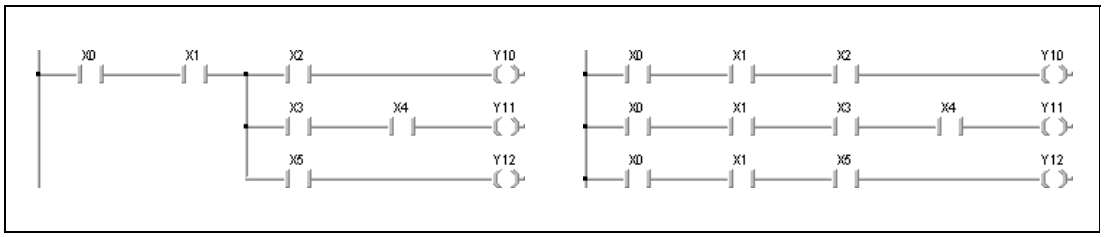
MPP Read and clear operation result (memory pop)

The MRD instruction reads stored operation results via an MPS instruction. The following operation executed depends on the reading result. Then the result is cleared.
Subtracts 1 from the number of MPS instruction times of use.

The MPS, MRP and MPP instructions are independent instructions and do not require any device.

In ladder programming mode the MPS, MRD and MPP instructions are not displayed explicitly. Whether connections are of the MPS, MRD or MPP type depends on the structure of the ladder diagram.

The example on the left shows a ladder diagram applying MPS, MRD or MPP instructions. The example on the right shows a ladder diagram without MPS, MRD or MPP instructions.



The number of MPS instructions in a program must equal the number of MPP instructions. Failure to observe this will not correctly display the ladder in the ladder mode of the peripheral device.

Program Example 1

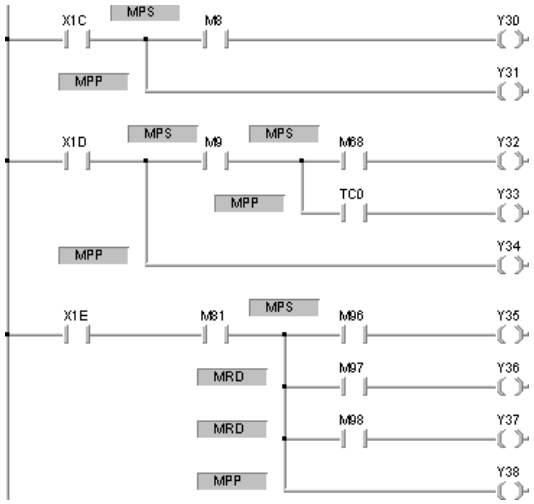
MPS, MRD, MPP

The following program illustrates the use of instructions for programming combined connections.

MELSEC Instruction List

```
MELSEC
LD    X1C
MPS
AND   M8
OUT   Y30
MPP
OUT   Y31
LD    X1D
MPS
AND   M9
MPS
AND   M68
OUT   Y32
MPP
AND   T0
OUT   Y33
MPP
OUT   Y34
LD    X1E
AND   M81
MPS
AND   M96
OUT   Y35
MRD
AND   M97
OUT   Y36
MRD
AND   M98
OUT   Y37
MPP
OUT   Y38
```

Ladder Diagram



Program Example 2

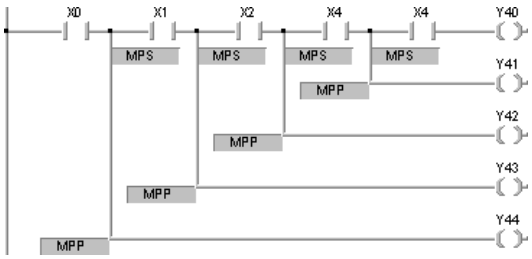
MPS, MRD, MPP

The following program illustrates the programming of instructions that output interim results in a series connection.

MELSEC Instruction List

```
MELSEC
LD    X0
MPS
AND   X1
MPS
AND   X2
MPS
AND   X3
MPS
AND   X4
OUT   Y40
MPP
OUT   Y41
MPP
OUT   Y42
MPP
OUT   Y43
MPP
OUT   Y44
```

Ladder Diagram



5.2.3 INV

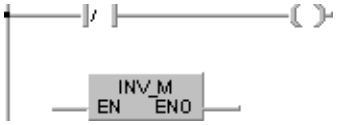
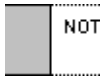
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

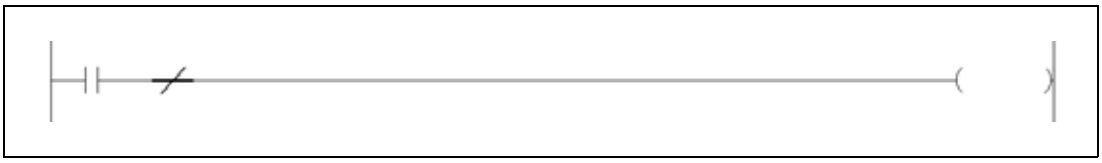
Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List (IEC Instruction)
		

GX Works2



Variables

Set Data	Meaning	Data Type
—	—	—

Functions

Operation result inversion

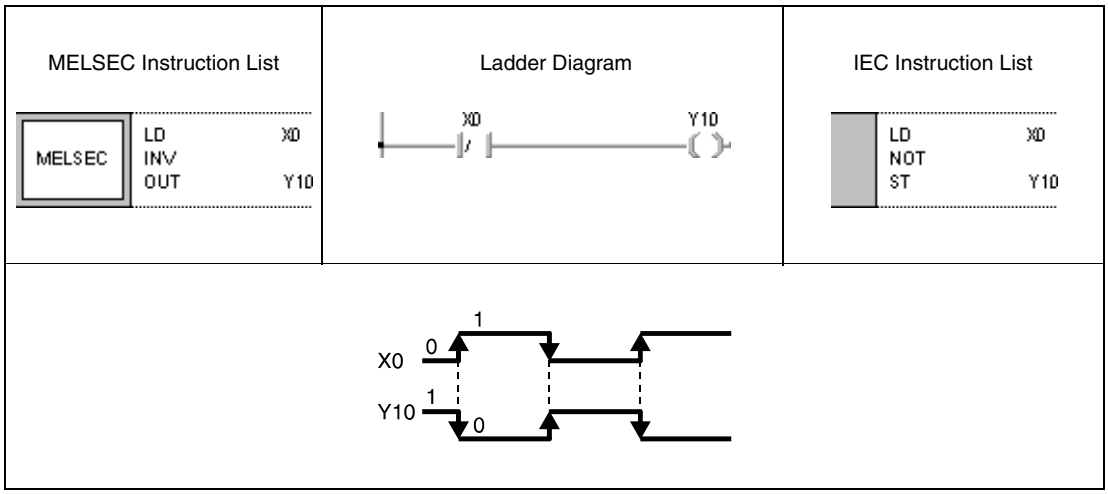
INV Inversion instruction

The INV instruction inverts the operation result preceding the INV instruction.

- If the result is 1 before the operation it will be 0 afterwards.
- If the result is 0 before the operation it will be 1 afterwards.

Program Example

The following program inverts the status of X0 and outputs the inverted signal at Y10.



5.2.4 MEP, MEF

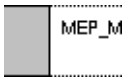
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

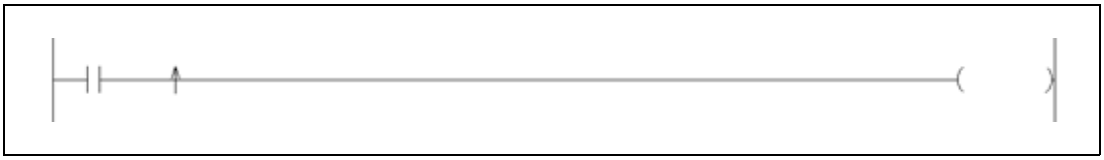
Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

GX Works2



Variables

Set Data	Meaning	Data Type
—	—	—

Functions

Operation result into pulse conversion

MEP Pulse generation at leading edge of operation result

The MEP instruction is used in cases where the applied instructions cannot output operation results as specified pulse output.

The MEP instruction is set after the according instruction and generates one output pulse, when the input signal changes from 0 to 1 (at leading edge). The next pulse is generated when the input is at leading edge once again.

MEF Pulse generation at trailing edge of operation result

The MEF instruction is used in cases where the applied instructions cannot output operation results as specified pulse output. The MEF instruction is set after the according instruction and generates one output pulse, when the input signal changes from 1 to 0 (at trailing edge). The next pulse is generated when the input is at trailing edge once again.

These two instructions are especially suitable for multiple contacts connections. For example, multiple NO contacts (normally open contacts) connected in series would maintain the operation result 1 if they were all closed. If a relay was set by this operation result, it could not be reset. With a MEP instruction connected in series with these NO contacts the relay could be reset because the instruction outputs one pulse only, if the series connection result of all contacts changes from 0 to 1.

NOTE

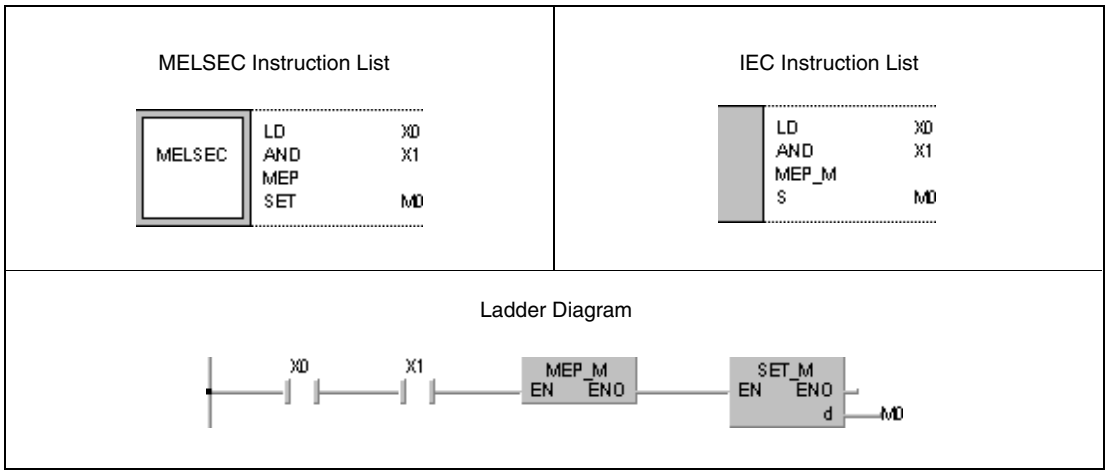
The MEP and MEF instructions will occasionally not function properly when pulse conversion is applied to contacts that are indexed by a subroutine or by a FOR/NEXT instruction. In this case, the EGP/EGF instruction has to be applied.

The MEP/MEF instruction operates with the operation results immediately prior to the MEP and MEF instructions. For this reason, an AND instruction should be used at the same position. The MEP and MEF instructions cannot be used at the LD or OR position.

Program Example

MEP

With leading edge from the series connection result at X0 and X1, the following program sets the relay M0.



5.2.5 EGP, EGF

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

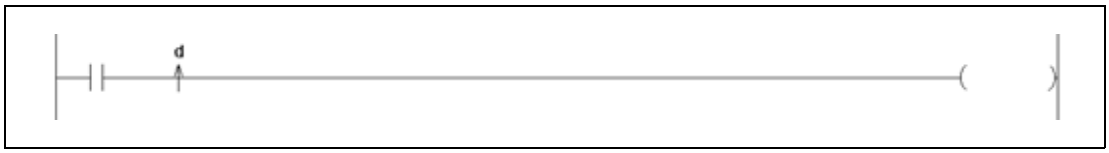
Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				V
d	—	—	—	—	—	—	—	—	●

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
MELSEC EGP_M d	EGP_M EN ENO d	EGP_M d

GX Works2



Variables

Set Data	Meaning	Data Type
d	Edge relay, storing the operation result.	bit (V only)

Functions

Pulse conversions of edge relay operation results

EGP Switching an edge relay with leading edge of an operation result

Operation results up to the EGP instruction are stored in memory by the edge relay (V). Goes ON (continuity status) at the leading edge (OFF to ON) of the operation result up to the EGP instruction. If the operation result up to the EGP instruction is other than a leading edge (i.e., from ON to ON, ON to OFF, or OFF to OFF), it goes OFF (non-continuity status).

EGF Switching an edge relay with trailing edge of an operation result

Operation results up to the EGF instruction are stored in memory by the edge relay (V). Goes ON at the trailing edge (from ON to OFF) of the operation result up to the EGF instruction. If the operation result up to the EGF instruction is other than a trailing edge (i.e., from OFF to ON, ON to ON, or OFF to OFF), it goes OFF (non-continuity status).

The EGP and EGF instructions are used for subroutine programs, and for conducting pulse operations for programs designated by indexing between the FOR and NEXT instructions.

The EGP and EGF instructions can be used like an AND instruction.

Program Example

EGP

The following program first resets the index register Z0 to 0 and then calls the subroutine UP1 (1). With leading edge X0Z0 is set to X0 and V0Z0 is set to V0. Further, D0Z0 is set to D0 and incremented by 1.

After returning, the index register Z0 stores 1, and the subroutine is called again (2). With leading edge from X1, V1 is set and D1 is incremented.

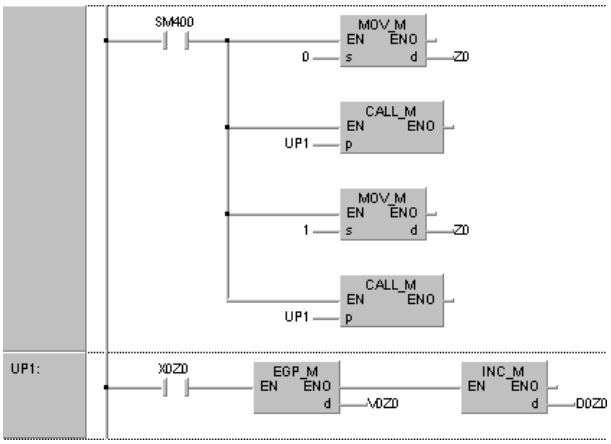
MELSEC Instruction List

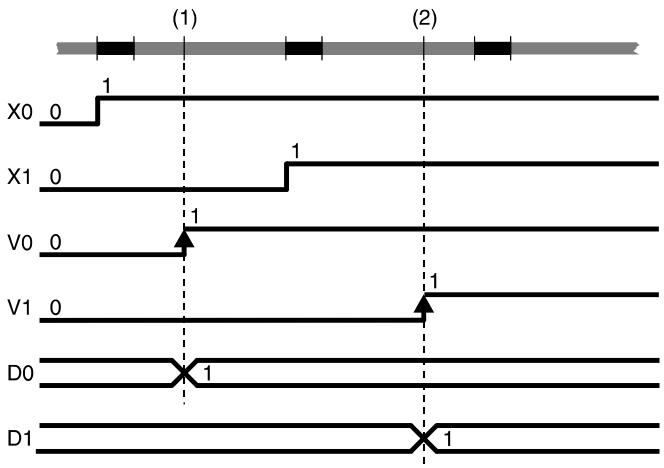
```
MELSEC   LD     SM400
         MOV    K0
                Z0
         CALL   UP1
         MOV    K1
                Z0
         CALL   UP1
         FEND
UP1:     LD     X0Z0
MELSEC   EGP    V0Z0
         INC    D0Z0
         RET
```

IEC Instruction List

```
         LD     SM400
         MOVE_E 0, Z0
         JMP    UP1
         LD     SM400
         MOVE_E 1, Z0
         JMP    UP1
UP1:     LD     X0Z0
         EGP_M  V0Z0
         INC_M  D0Z0
```

Ladder Diagram





5.3 Output instructions

5.3.1 OUT

CPU

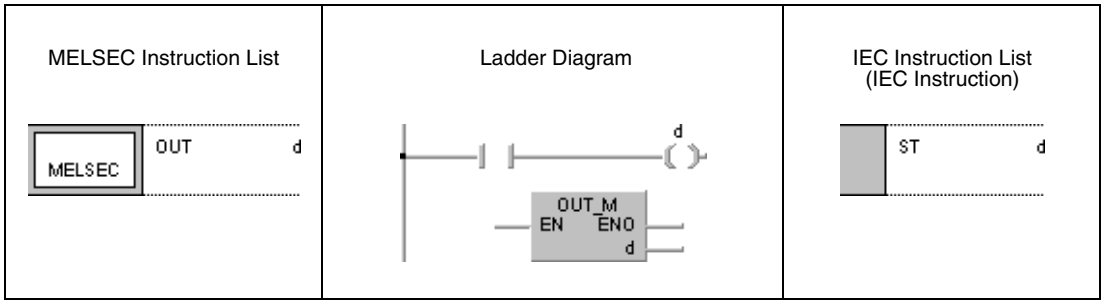
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

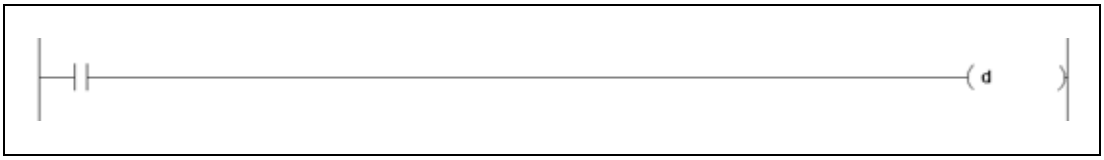
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DY
d	●[1)]	●	●	●	●	●	—	—	●

[1] Except T, C, F

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d	Number of device to be set (1) or reset (0).	bit

Functions **Output instruction**

OUT Out instruction (excluding timers, counters, and annunciators)

An output is set depending on the preceding input condition.

Several OUT instructions can be programmed in parallel following an input condition.

The operation result of an OUT contact can be used as input condition for the following program steps as NO contact (normally open) or NC contact (normally closed).

OUT Instruction				
Input Condition	Output Contact	Bit Device or Word Device Bit Designation	Contact Type	
			NO Contact	NC Contact
0	OFF	0	Non-continuity	Continuity
1	ON	1	Continuity	Non-continuity

NOTE *See section 3.12.1 for the operation to be performed when the OUT instruction for the same device is executed more than once during one scan.*

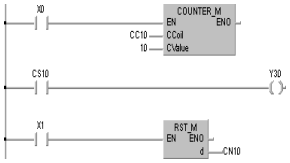
Program Example 1 OUT

The following program shows the programming of an OUT instruction using bit devices as outputs (Y33 through Y35).

MELSEC Instruction List

```
MELSEC
LD   X5
OUT  Y33
LD   X6
OUT  Y34
OUT  Y35
```

Ladder Diagram



IEC Instruction List

```
LD   X5
ST   Y33
LD   X6
ST   Y34
ST   Y35
```

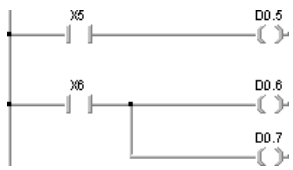
Program Example 2 OUT

The following program shows the programming of an OUT instruction using bits of the word device D0 (bits b5 through b7).

MELSEC Instruction List

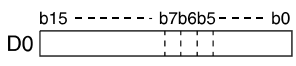
```
MELSEC
LD   X5
OUT  D0.5
LD   X6
OUT  D0.6
OUT  D0.7
```

Ladder Diagram



IEC Instruction List

```
LD   X5
ST   D0.5
LD   X6
ST   D0.6
ST   D0.7
```



5.3.2 OUT T, OUTH T

CPU

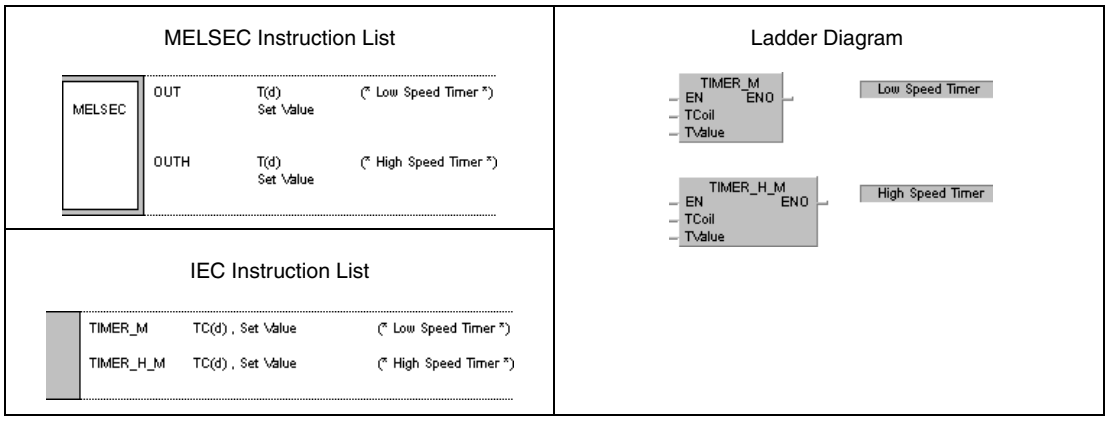
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

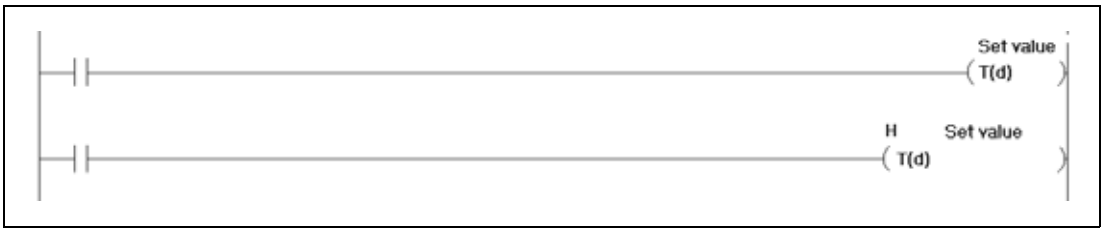
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K	Other
	Bit	Word		Bit	Word				
d	●[1)]	—	—	—	—	—	—	—	—
2)	—	●[3)]	●	—	●	●	—	●[4)]	—

[1] T only
[2] Time setting
[3] Except T, C
[4] Specification of time settings by decimal constants (K) only. Hexadecimal constants cannot be read.

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d	Number of timer	bit
Set value	Timer setting value	BIN 16-bit

Functions **Setting timers**

OUT T Low speed timer (100 ms)

OUTH T High speed timer (10 ms)

When the operation results up to the OUT(H) T instruction are ON, the timer coil goes ON and the timer counts up to the value that has been set. This time is designated directly by a constant or variably by the value in a data register.

After the specified time has passed (setting value ≤ actual value) the succeeding input contact is set.

The operation result of the OUT(H) T contact is programmed as input condition in one (or several) following program step(s) like a common NO (normally open) or NC (normally closed) contact.

Several OUT(H) T instructions can be programmed succeeding one single input condition.

The contact responds as follows when the operation result up to the OUT instruction is a change from ON to OFF:

Timer			Timer as Input Condition			
Type	Timer Coil	Actual Value	Prior to Time Up		After Time Up	
			NO contact	NC contact	NO contact	NC contact
Low speed (100 ms)	OFF	0	Non-continuity	Continuity	Non-continuity	Continuity
High speed (10 ms)						
Low speed (100 ms), retentive	OFF	Actual value maintained	Non-continuity	Continuity	Continuity	Non-continuity
High speed (10 ms), retentive						

To clear the present value of a retentive timer and turn the contact OFF after time up, use the RST instruction.

A negative number (–32768 to –1) cannot be set as the setting value for the timer. If the setting value is 0, the timer will time out when the OUT(H) T instruction is executed. Please note: When specifying a setting value for the timer using a word device the value is not checked whether it is in the setting range. Check the value in the user program so that a negative number is not set.

The execution of the OUT(H) T instruction performs as follows:

– The timer coil designated by d is set or reset.

– The according timer contact is set or reset.

– The time settings are refreshed.

If a program jumps to an OUT(H) T instruction while it is executed, the contact conditions and timer settings are maintained.

If one instruction is executed repeatedly within one cycle, the value of the repetitions is refreshed.

Indexing for timer coils or contacts can be conducted only by Z0 or Z1. Timer setting value has no limitation for indexing.

NOTE

Timer's time limit
Time limit of the timer is set in the PLC system of the PLC parameter dialog box.

Type of Timer	Basic Model QCPU, High Performance model QCPU, Process CPU, Redundant CPU		Universal model QCPU, LCPU	
	Setting Range	Setting Unit	Setting Range	Setting Unit
Low speed timer Low speed retentive timer	1 ms to 1000 ms (Default: 100 ms)	1 ms	1 ms to 1000 ms (Default: 100 ms)	1 ms
High speed timer High speed retentive timer	0.1 ms to 100.0 ms (Default: 10.0 ms)	0.1 ms	0.01 ms to 100.0 ms (Default: 10.0 ms)	0.01 ms

Please refer to section A.5.4 for more information about timers.

Program Example 1

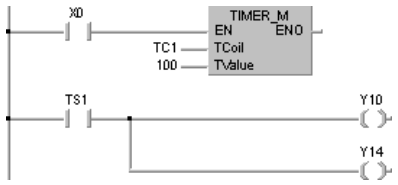
OUT T

10 seconds after setting X0, the following program sets the outputs Y10 and Y14. A low speed timer (T1, 100 ms) is used.

MELSEC Instruction List

```
MELSEC
LD    X0
OUT   T1
      K100
LD    T1
OUT   Y10
OUT   Y14
```

Ladder Diagram



IEC Instruction List

```
LD      X0
TIMER_M TC1 , 100
LD      TS1
ST      Y10
ST      Y14
```

Program Example 2

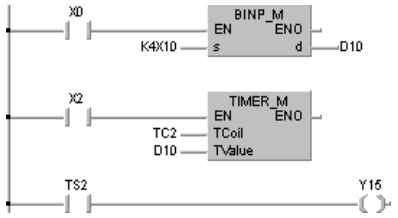
OUT T

The following program reads the time setting via the inputs X10 to X1F in BCD data format. With leading edge from X0 BCD data is converted into BIN data first and stored in D10. After setting X2 the time setting is read. After the set time has passed Y15 is set. A low speed timer (T2, 100 ms) is used.

MELSEC Instruction List

```
MELSEC
LD    X0
BINP  K4X10
      D10
LD    X2
OUT   T2
      D10
LD    T2
OUT   Y15
```

Ladder Diagram



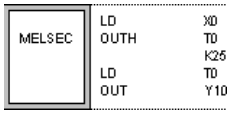
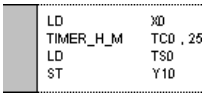
IEC Instruction List

```
LD      X0
BINP_M  K4X10,D10
LD      X2
TIMER_M TC2,D10
LD      TS2
ST      Y15
```

Program Example 3

OUTH T

250 ms after setting X10 the following program sets the output Y10. A high speed timer (10 ms) is used.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

5.3.3 OUT C

CPU

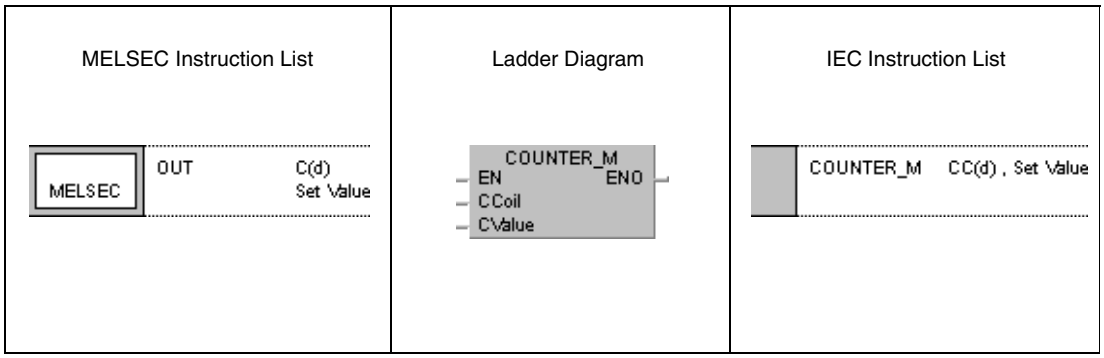
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

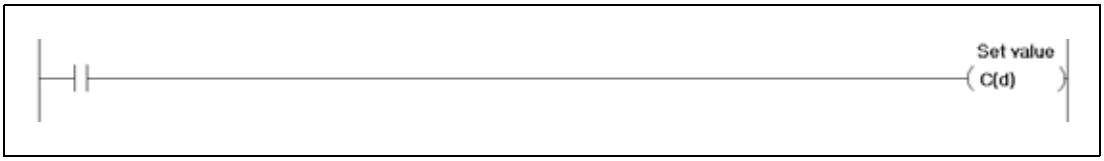
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K	Other
	Bit	Word		Bit	Word				
d	●[1)]	—	—	—	—	—	—	—	—
2)	—	●[3)]	●	—	●	●	—	●[4)]	—

[1] C only
[2] Count setting
[3] Except T, C
[4] Specification of count settings by decimal constants (K) only. Hexadecimal constants cannot be read.

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d	Number of counter	bit
Set value	Counter setting value	BIN 16-bit

Functions **Setting counters**

OUT C Counter

When the operation results up to the OUT instruction change from OFF to ON, 1 is added to the present value (count value).

The count up status (present value set value), and the contacts respond as follows:

No count is conducted with the operation results at ON. There is no need to perform pulse conversion on count input.

After the count up status is reached, there is no change in the count value or the contacts until the RST instruction is executed.

A negative number (–32768 to –1) cannot be set as the setting value for the timer. If the set value is 0, the processing is identical to that which takes place for 1.

Indexing for the counter coil and contact can use only Z0 and Z1. Counter setting value has no limitation for indexing.

NOTE *Please refer to section A.5.5 of this manual for more information about counters.*

Program Example 1 OUT C

After X0 has been set for 10 times, the following program sets Y30 and if X1 is set resets Y30.

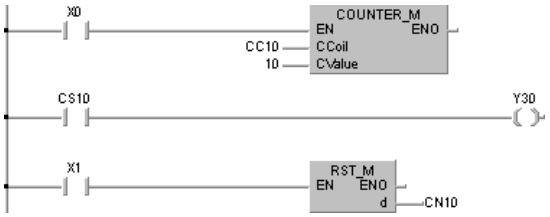
MELSEC Instruction List

```
MELSEC   LD    X0
         OUT   C10
               K10
         LD    C10
         OUT   Y30
         LD    X1
         RST   C10
```

IEC Instruction List

```
         LD          X0
         COUNTER_M   CC10 , 10
         LD          CS10
         ST          Y30
         LD          X1
         RST_M       CN10
```

Ladder Diagram



Program Example 2

OUT C

The following program sets the setting value in C10 to 10 (D0 =10) with leading edge from X0, and to 20 (D0 =20) with leading edge from X1.

If X3 is set, the counter starts counting and sets Y30 when it reaches the setting value in D0.

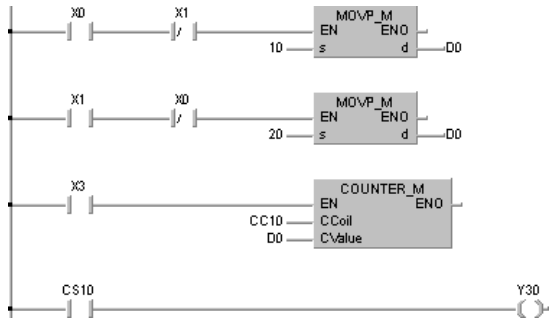
MELSEC Instruction List

```
MELSEC
        LD      X0
        ANI     X1
        MOVP    K10
                D0
        LD      X1
        ANI     X0
        MOVP    K20
                D0
        LD      X3
        OUT     C10
                D0
        LD      C10
        OUT     Y30
```

IEC Instruction List

```
        LD      X0
        ANDN    X1
        MOVP_M  10 , D0
        LD      X1
        ANDN    X0
        MOVP_M  20 , D0
        LD      X3
        COUNTER_M  CC10 , D0
        LD      CS10
        ST      Y30
```

Ladder Diagram



5.3.4 OUT F

CPU

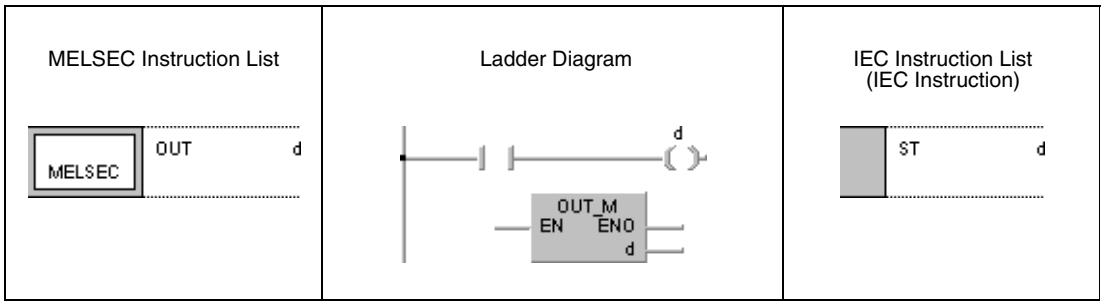
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

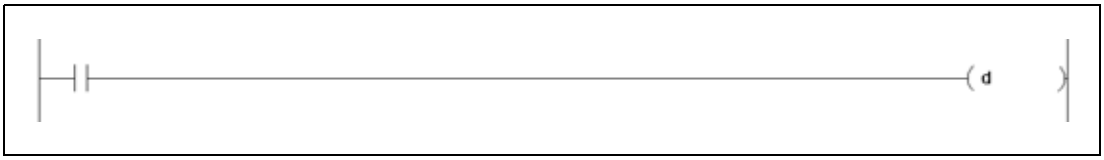
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	●[1)]	—	—	—	—	—	—	—	—

[1] F only

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d	Number of annunciator to be set	bit (F only)

Functions

Output of annunciators

OUT F Annunciator

If the input condition of an OUT F instruction is set, the annunciator is set and the following operations are performed:

- The "USER" LED ("ERR." LED for Basic model QCPU) lights up.
- The numbers of set annunciators are stored in the special registers SD64 through SD79.
- The value in SD63 is incremented by 1.

If special register SD63 stores the value 16, i.e. 16 annunciators are already ON, no further numbers are stored in the range of SD64 through SD79.

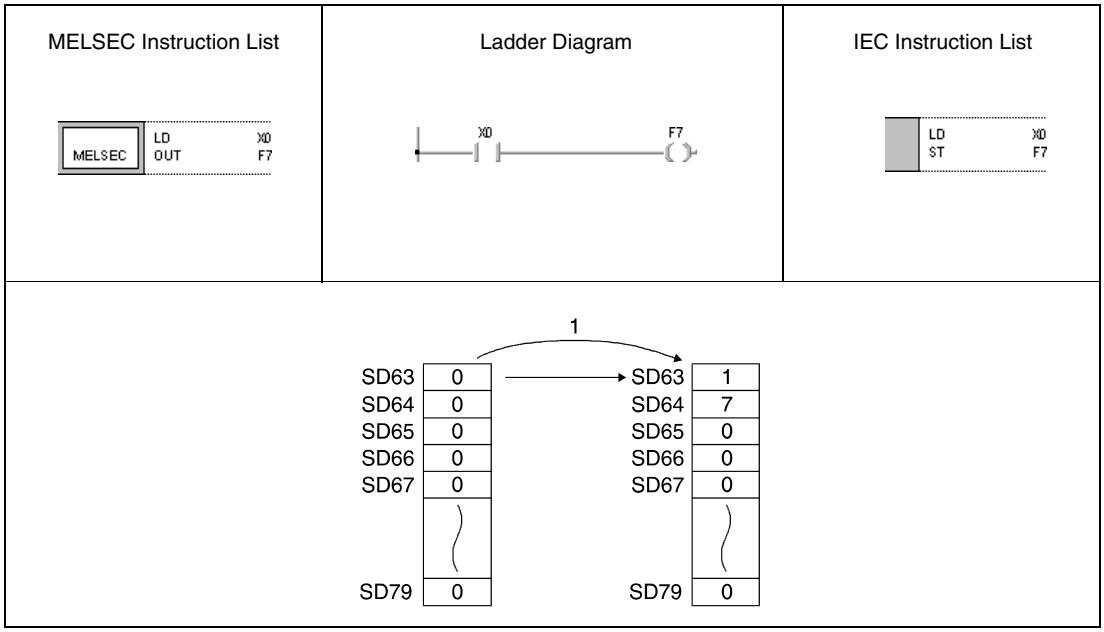
If an annunciator is reset via an OUT instruction, the condition of the "USER" LED ("ERR." LED for Basic model QCPU), and the content of the special registers SD63 through SD79 are maintained.

Annunciators, registers, and displays are cleared via the RST F instruction.

Program Example

OUT F

If X0 is set, the following program sets the annunciator F7. The number 7 is stored in the registers SD64 through SD79. The value in register SD63 is incremented by 1 (i.e. 1 number of annunciator stored).



[1] X0 is set

5.3.5 SET

CPU

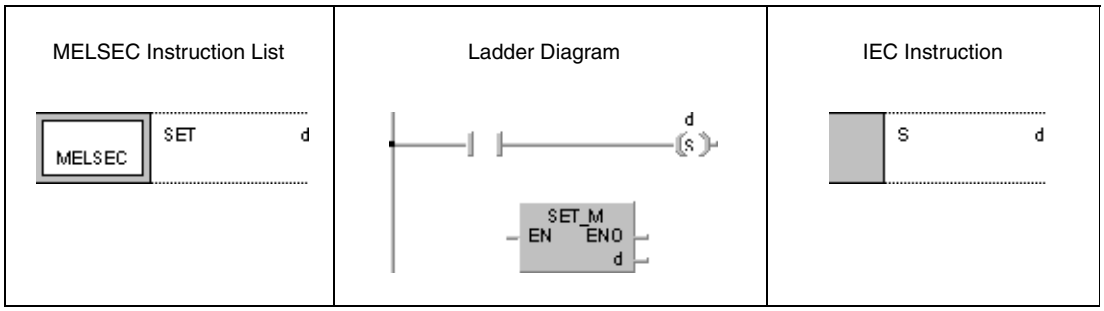
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

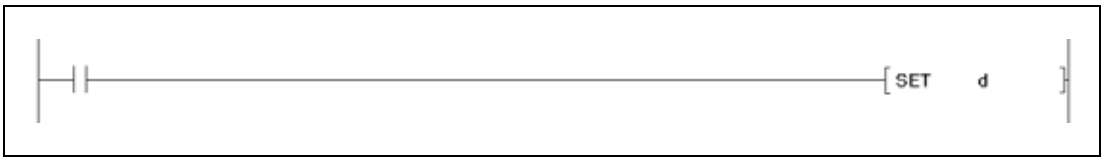
	Usable Devices									
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other	
	Bit	Word		Bit	Word				BL	DY
d	●	●[1)]	●[1)]	●	●	●	—	—	●	●

[1] Except T, C

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d	Number of bit device (output contact) to be set / word device bit designation.	bit

Functions

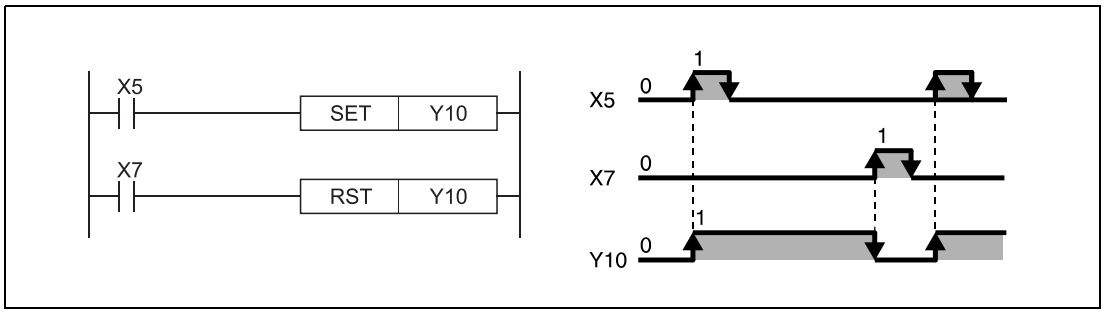
Setting of devices

SET Set instruction

The SET instruction consists of a SET command followed by a number (address) of device d to be set.

When the execution command is turned ON, the status of the designated device (bit device or designated bit of word device) is set to 1.

If the input condition is reset once again, the set device remains set. A device can be reset via the RST instruction.



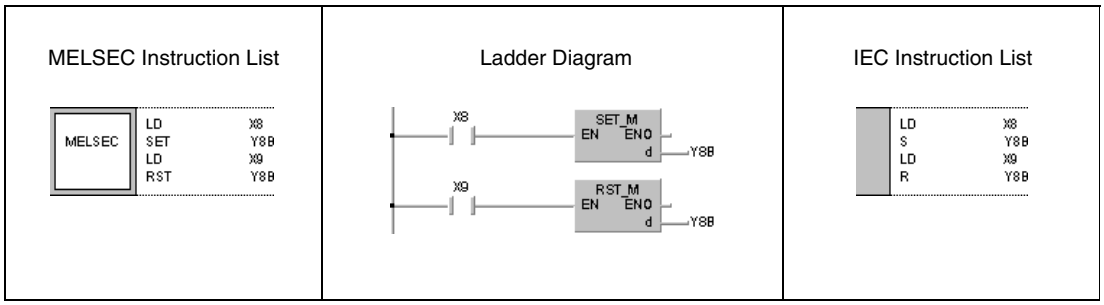
NOTE

See section 3.12.2 for the operation to be performed when the SET instruction for the same device is executed more than once during one scan.

Program Example 1

SET

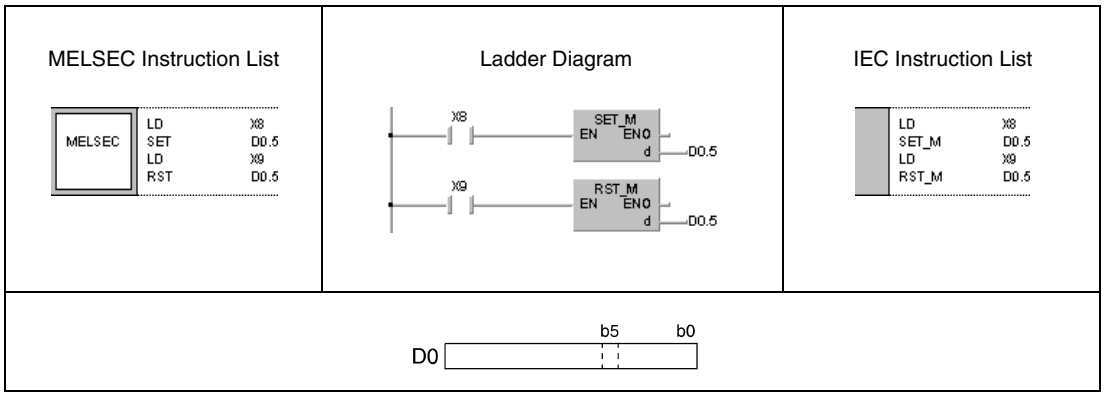
If X8 is set, the following program sets the output Y8B. If X9 is set, Y8B is reset.



Program Example 2

SET

If X8 is set, the following program sets bit 5 (b5) in D0 from 0 to 1. If X9 is set, this bit is reset.



5.3.6 RST

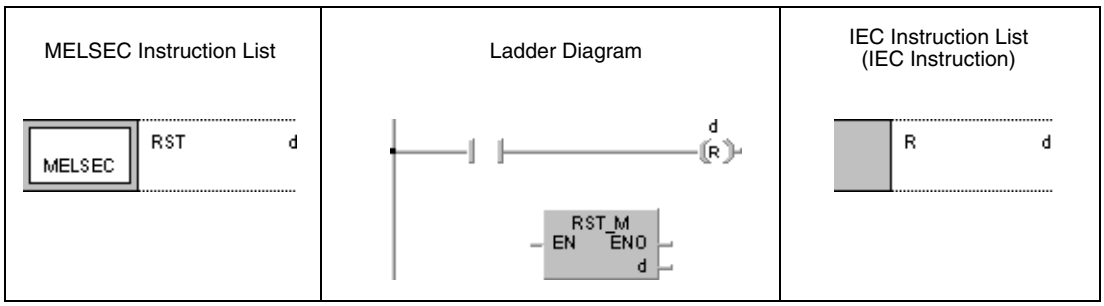
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

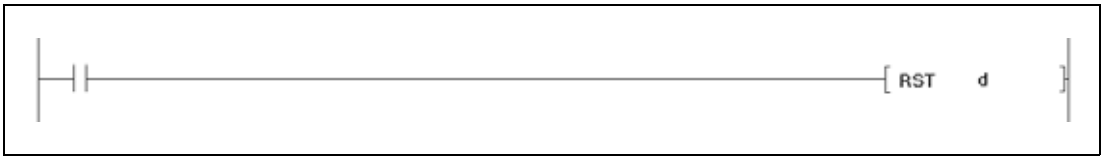
Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DY
d	●	●	●	●	●	●	●	—	●

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d	Device to be reset	bit, BIN 16-bit

Functions

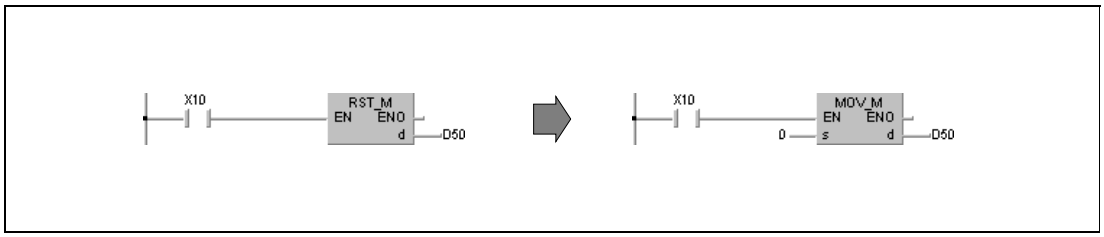
Resetting devices

RST Reset instruction

The RST instruction consists of an RST command followed by a number (address) of device d to be reset.

After execution of the RST instruction input and output contacts of bit devices are switched off (0), actual values of timers and counters (T, C) are reset to 0 and the according contacts are switched off, the designated bit of a word device is reset to 0, and the content of word devices is reset to 0.

The functions of the word devices designated by the RST instruction are identical to that of the MOV instruction in the following diagram on the right.



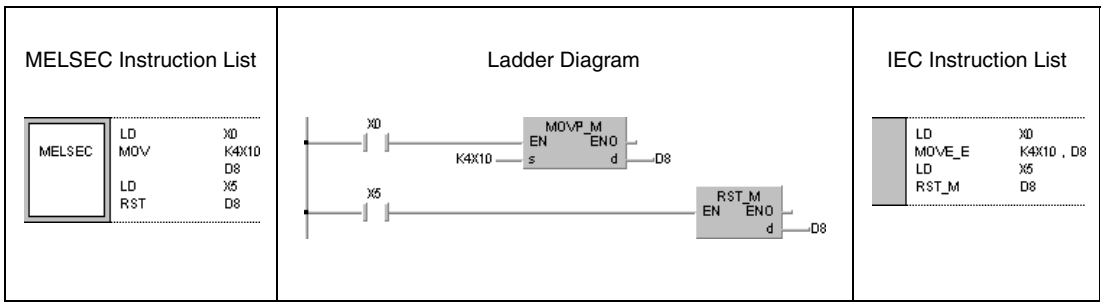
NOTE

See section 3.12.2 for the operation to be performed when the RST instruction for the same device is executed more than once during one scan.

Program Example 1

RST

With leading edge from X0, the following program stores the content at X10 through X1F in the data register D8. If X5 is set, the content of D8 is reset to 0.



Program Example 2

RST T, C

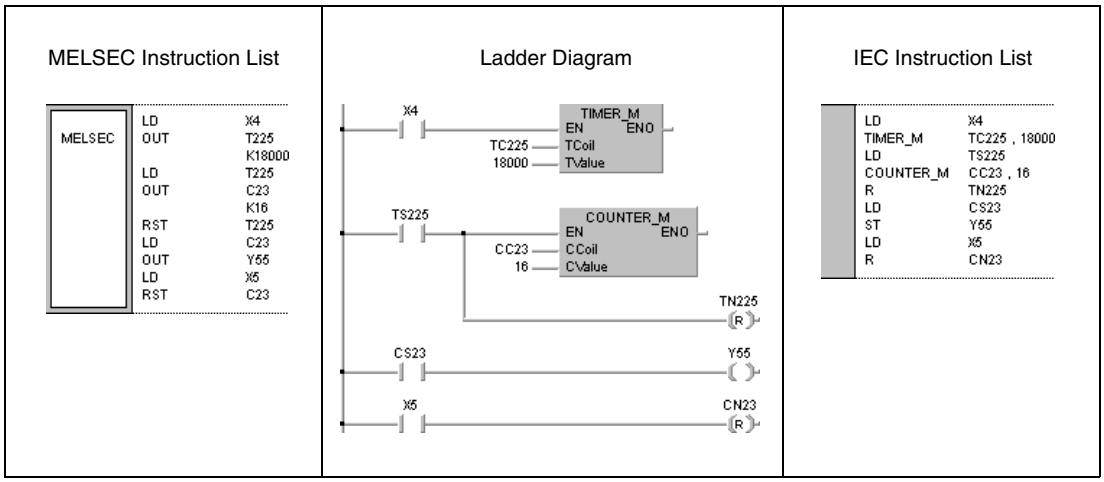
The following program illustrates resetting of retentive timers and counters.

In the first program step T225 is set, if X4 has been set for 30 minutes (18000 seconds).

In the second program step C23 counts the number of times T225 is set.

If this timer is set for 16 times (setting value of C23 = 16) the output Y55 is set.

If X5 is set, the counter will be reset to 0.



5.3.7 SET F, RST F

CPU

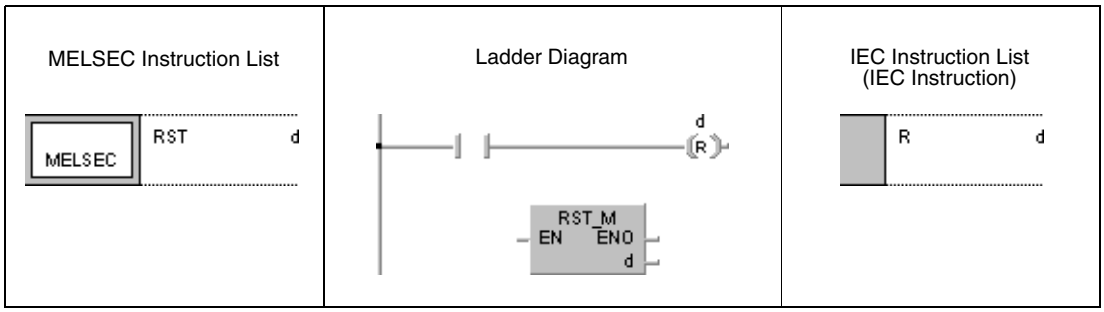
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

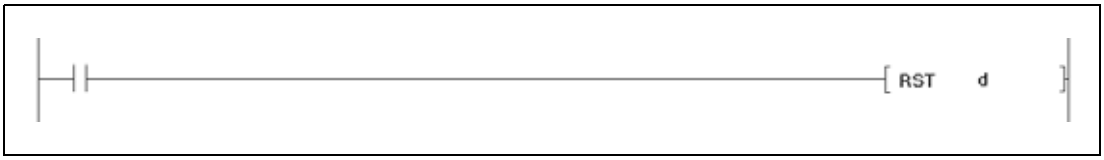
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	●[1)]	—	—	—	—	—	—	—	—

[1] F only

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d (SET)	Number of annunciator to be set	bit (F only)
d (RST)	Number of annunciator to be reset	bit (F only)

Functions **Setting and resetting of annunciators**

SET F Set instruction

The SET F instruction consists of a SET command followed by an annunciator number designated by d to be set.

After execution of the input condition, the designated device number d is set. The SET instruction outputs a pulse to set an annunciator.

The following procedures are executed:

- The "USER" LED lights up. ("ERR." LED for Basic model QCPUs)
- The numbers (addresses) of set annunciators are stored in the registers SD64 through SD79.
- The value in SD63 is incremented by 1.

If special register SD63 stores the value 16, i.e. 16 annunciators are already ON, no further numbers are stored in the range of SD64 through SD79.

RST F Reset instruction

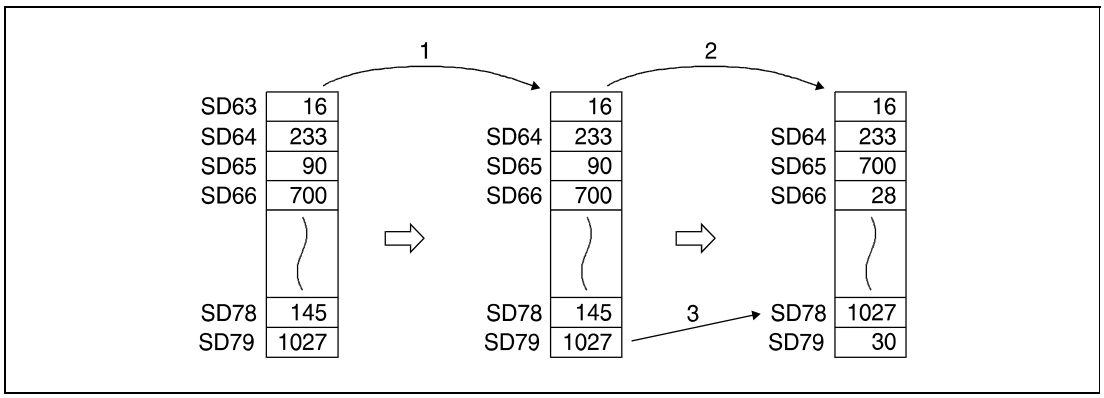
The RST F instruction consists of an RST command followed by an annunciator number designated by d to be reset.

After execution of the input condition the designated device number is reset. The output signal resetting an annunciator is a pulse.

The number of a reset annunciator is cleared from the registers SD64 through SD79 and the value in register SD63 is decremented by 1. If the value in the register SD63 was 16 and annunciators are cleared from this register via an RST F instruction then those annunciator numbers are stored that could not be stored before. These annunciator numbers are stored in the cleared registers within SD64 through SD79.

If the value in special register SD63 is decremented to 0 and all annunciators are reset, the "USER" LED turns off ("ERR." LED for Basic model QCPUs).

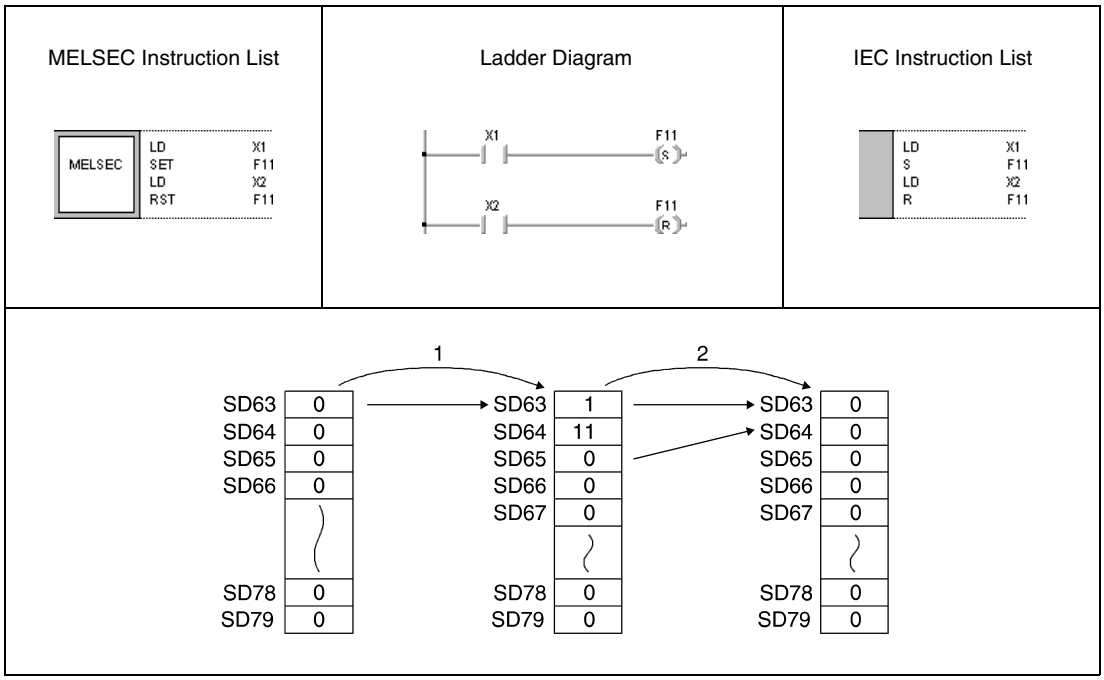
In the diagram below F30 is set in a first step (1) but cannot be registered because there are 16 numbers already stored. In a second step (2) F90 is reset. Thus, in a third step (3) F30 can be stored in SD79 because the other stored annunciators are shifted up by one cleared register (SD65).



Program Example

SET F/ RST F

If X1 is set, the following program sets the annunciator F11. The number 11 is stored in the registers SD64 through SD79 and the value in SD63 is incremented by 1 (1). Then, if X2 is set, the annunciator F11 is reset. The number 11 is cleared from the special registers SD64 through SD79 and the value in SD63 is decremented by 1 (2).



5.3.8 PLS, PLF

CPU

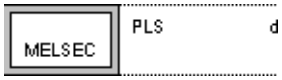
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

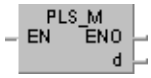
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DY
d	●	●	●	●	●	●	—	—	●

GX IEC Developer

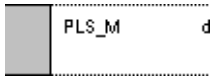
MELSEC Instruction List



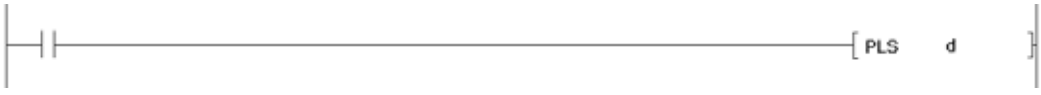
Ladder Diagram



IEC Instruction List



GX Works2



Variables

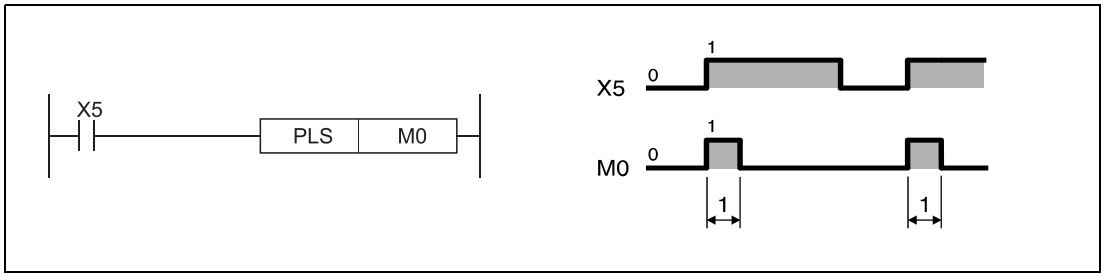
Set Data	Meaning	Data Type
d	Device of which the output signal is converted into a pulse	bit

Functions **Leading edge and trailing edge output**

PLS Output at leading edge

The PLS instruction consists of the PLS command followed by the number of device d to be set.

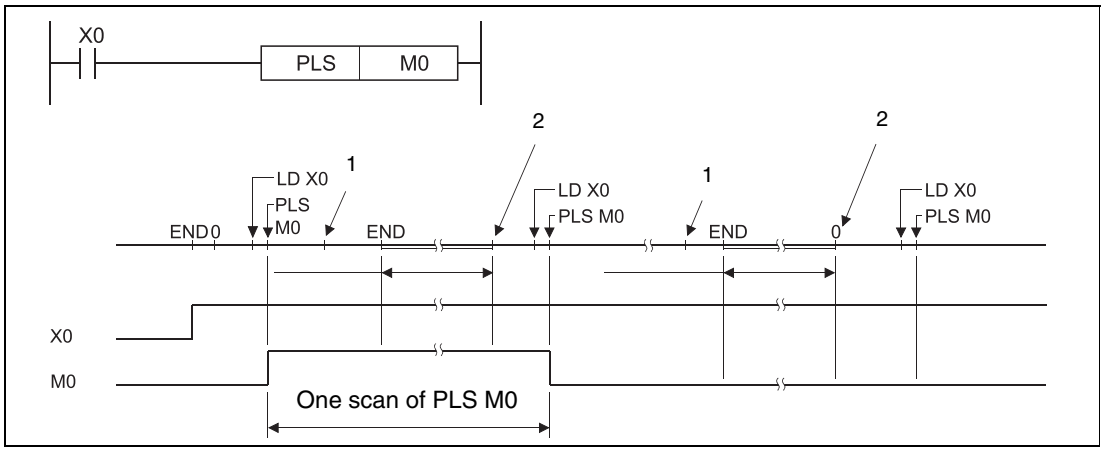
The PLS instruction (pulse) with leading edge from the input condition sets a device for one program scan.



[1] One scan

NOTE *See section 3.12.3 for the operation to be performed when the PLS instruction for the same device is executed more than once during one scan.*

If the RUN/STOP key switch on the CPU unit is set to STOP while a PLS instruction is executed, the PLS instruction will not be executed further on after the switch is set back to RUN even if the input condition is still set.



[2] RUN/STOP switch of the CPU switched from RUN to STOP

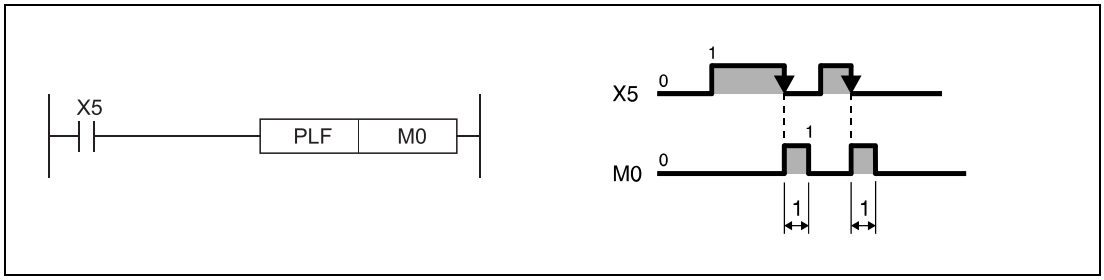
[3] RUN/STOP switch of the CPU switched from STOP to RUN

If a latch relay is designated by a PLS instruction, and the power is turned OFF while a latch relay is set, after turning ON the power again the designated latch relay is set for one scan.

PLF Output at trailing edge

The PLF instruction consists of the PLF command followed by the number of device d to be set.

The PLF instruction with trailing edge from the input condition sets a device for one program scan.



[1] One scan

NOTE *See section 3.12.4 for the operation to be performed when the PLF instruction for the same device is executed more than once during one scan.*

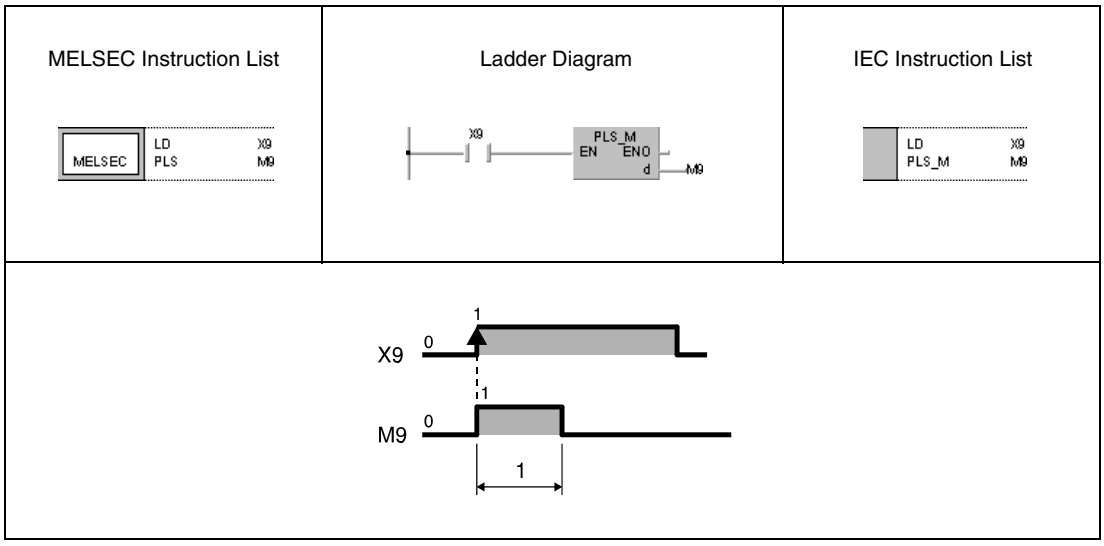
If the RUN/STOP switch of the CPU unit is set to STOP while a PLS instruction is executed, the PLS instruction will not be executed further on after the switch is set back to RUN even if the input condition is still set.

NOTE *The device d designated by a PLS or PLF instruction remains set for more than one program scan if a CJ or similar instruction was applied to jump to the PLS or PLF instruction and the part of program was not executed.*

Program Example 1

PLS

With leading edge from X9, the following program sets the internal relay M9 for one program scan.

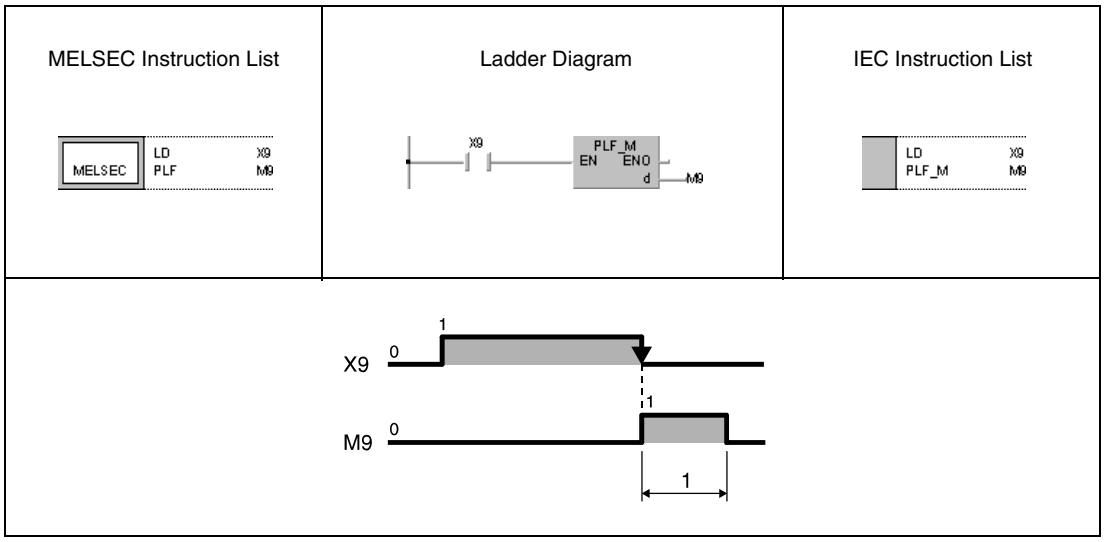


[1] One scan

Program Example2

PLF

With trailing edge from X9, the following program sets the internal relay M9 for one program scan.



[1] One scan

5.3.9 FF

CPU

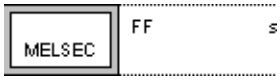
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

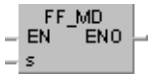
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DY
d	●	●	●	●	●	●	—	—	●

GX IEC Developer

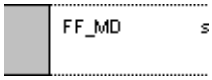
MELSEC Instruction List



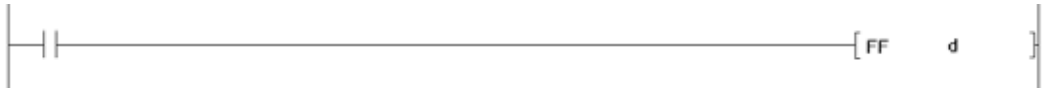
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
d	Number of bit device or designated bit of word device to be inverted.	bit

Functions

Bit device output inversion

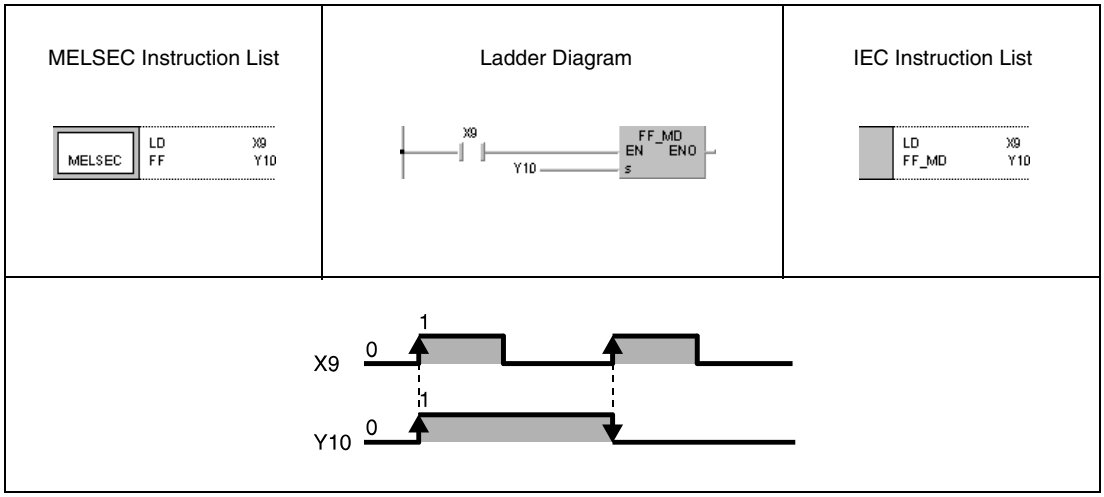
FF Inversion of bit output device

The FF instruction inverts the operation condition of the device designated by d with leading edge at the input of the FF instruction. The device can be a bit device or a specified bit of a word device. If the condition of the output device is set (1) it will be reset (0) after inversion. If the condition of the output device is reset (0), it will be set (1) after inversion.

Program Example 1

FF

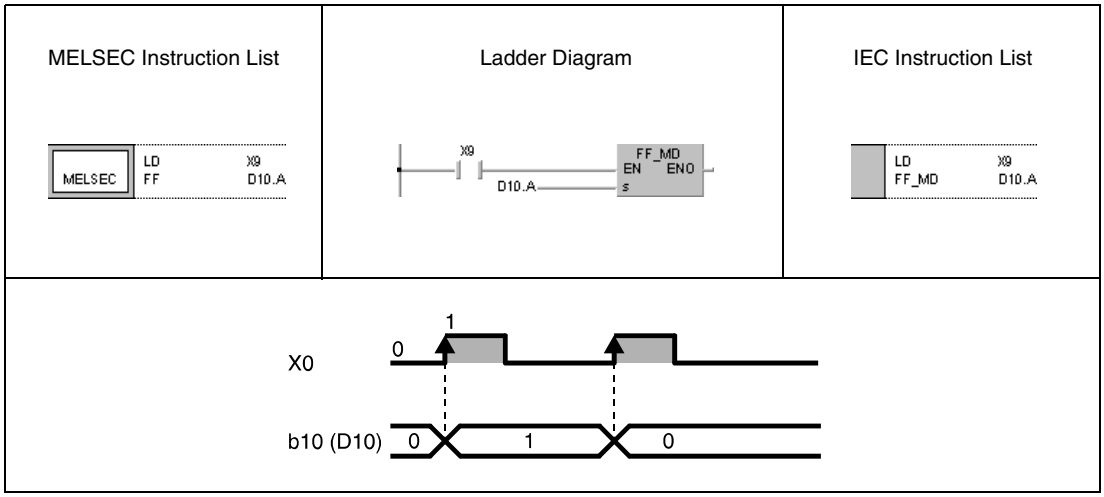
With leading edge from X9, the following program inverts the output condition of Y10.



Program Example 2

FF

With leading edge from X9, the following program inverts bit 10 (b10) of D10.



5.3.10 DELTA, DELTAP

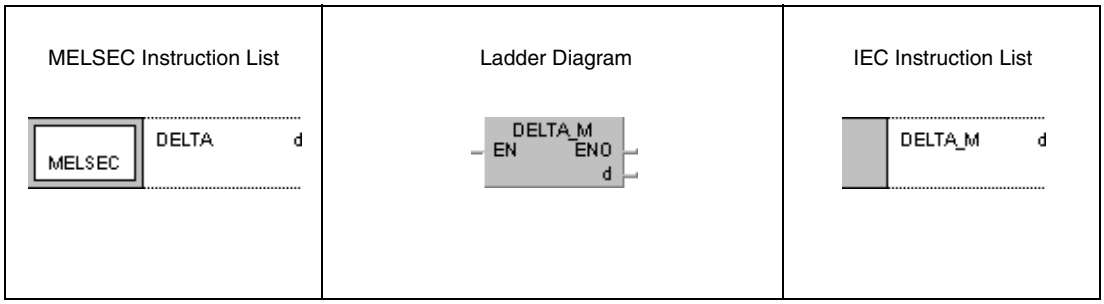
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

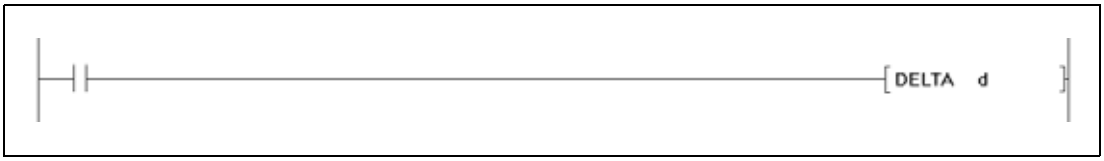
Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DY
d	—	—	—	—	—	—	—	—	●

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d	Number of direct access output to generate pulse at.	bit [1)]

[1] direct access outputs only

Functions

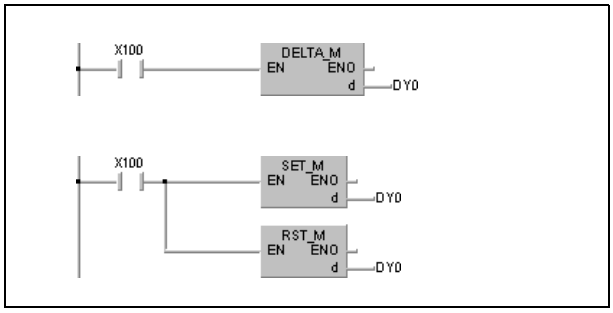
Generating pulses at direct access outputs

DELTA Pulse conversion of contacts

The DELTA instruction generates a pulse at a direct access output (DY) designated by d, i.e. the output is set for a certain time only.

If the output designated by the DELTA instruction is DY0, the executed function is identical to that of the SET/RST instruction (see diagram).

The DELTA(P) instruction is used by commands for leading edge execution in special function units.



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of output designated by d exceeds the output range. (Error code 4101)

Program Example

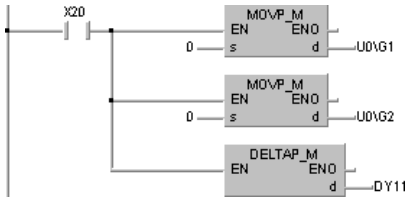
DELTAP

With leading edge from X20, the following program presets CH1 of the AD61 output unit mounted at slot 0 of the main base unit. The preset value 0 is stored at addresses 1 and 2 of the AD61 buffer memory. The DELTAP instruction outputs the preset instruction at DY11.

MELSEC Instruction List

```
MELSEC   LD      X20
         DMOVP   K0
                 U0\G1
         DELTAP  DY11
```

Ladder Diagram



IEC Instruction List

```
LD      X20
MOVP_M  0 , U0\G1
MOVP_M  0 , U0\G2
DELTAP_M DY11
```

5.4 Shift instructions

5.4.1 SFT, SFTP

CPU

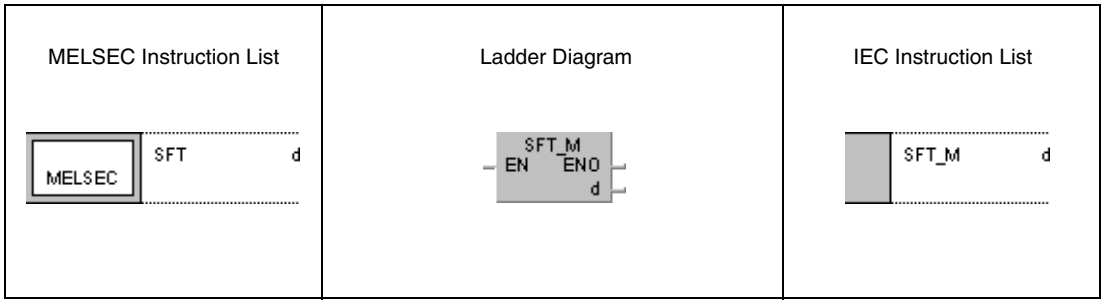
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

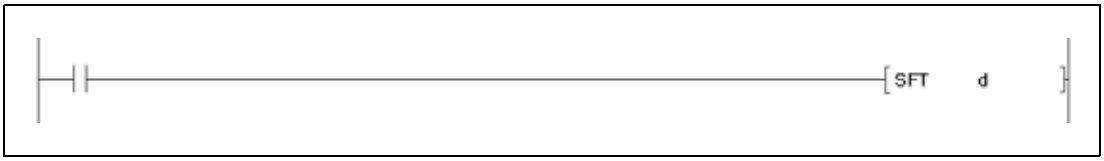
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				DY
d	●[1)]	●[1)]	●[1)]	●[1)]	●[1)]	●[1)]	—	—	●

[1] Except T and C

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d	Number of device to be shifted.	bit

Functions

Shift instruction

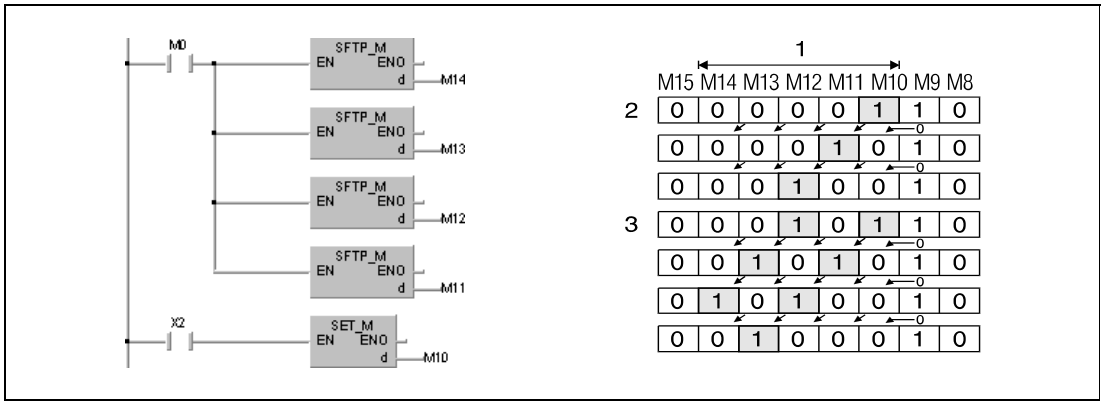
SFT Shifting bit devices

The SFT instruction shifts devices by one bit. Devices are only shifted via the SFTP instruction, if the input condition is set (leading edge).

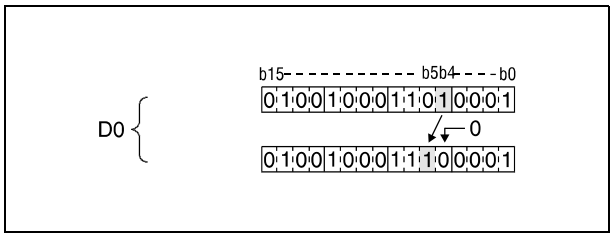
The instruction shifts the condition of a device (specified by d-1) to the destination address d. The condition of the device with the lower address d-1 is reset. Turn the first device to be shifted ON with the SET instruction.

If several SFT instructions are applied consecutively, the program starts from the device with the higher number.

The program below sets the internal relay M10 if X2 is set (2,3). The condition of M10 (1) is shifted via the SFT P instruction within the shift range (1).



If bits in word devices are shifted, the condition (0/1) of the bit d-1 is shifted to d. The bit d-1 is reset after the SFT instruction. In the following illustration bit 5 (b5) in D0 is shifted. Bit 4 (b4) is reset after execution of the instruction.



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU and LCPU only.)
  (Error code 4101)

Program Example

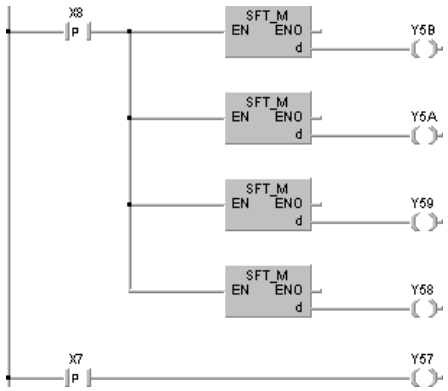
SFT

With leading edge from X8, the following program shifts the condition of Y57 to Y5B. With leading edge from X7, Y57 is set.

MELSEC Instruction List

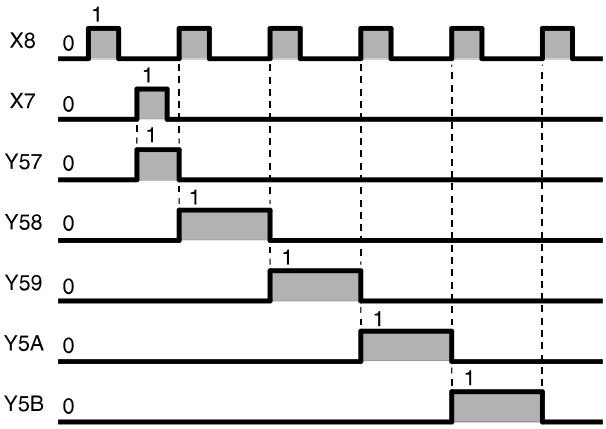
```
MELSEC
        LDP   X8
        SFT   Y5B
        SFT   Y5A
        SFT   Y59
        SFT   Y58
        LDP   X7
        SET   Y57
```

Ladder Diagram



IEC Instruction List

```
        LD    X8
        PLS_M M8
        LD    M8
        SFT_M Y5B
        SFT_M Y5A
        SFT_M Y59
        SFT_M Y58
        LD    X7
        PLS_M M7
        LD    M7
        S     Y57
```



5.5 Master control instructions

5.5.1 MC, MCR

NOTE *These instructions should not be used within the IEC editors.*

CPU

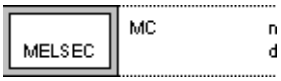
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices									
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other	
	Bit	Word		Bit	Word				N	DY
n	—	—	—	—	—	—	—	—	●	
d	●	●	●	●	●	●	—	—		●

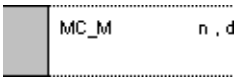
GX IEC Developer

MELSEC Instruction List

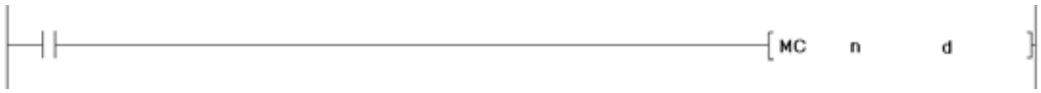


Ladder Diagram

IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
n	Level of nesting (N0 – N14).	Nesting
d	Number of device to set nesting.	bit

Functions Setting and resetting master control

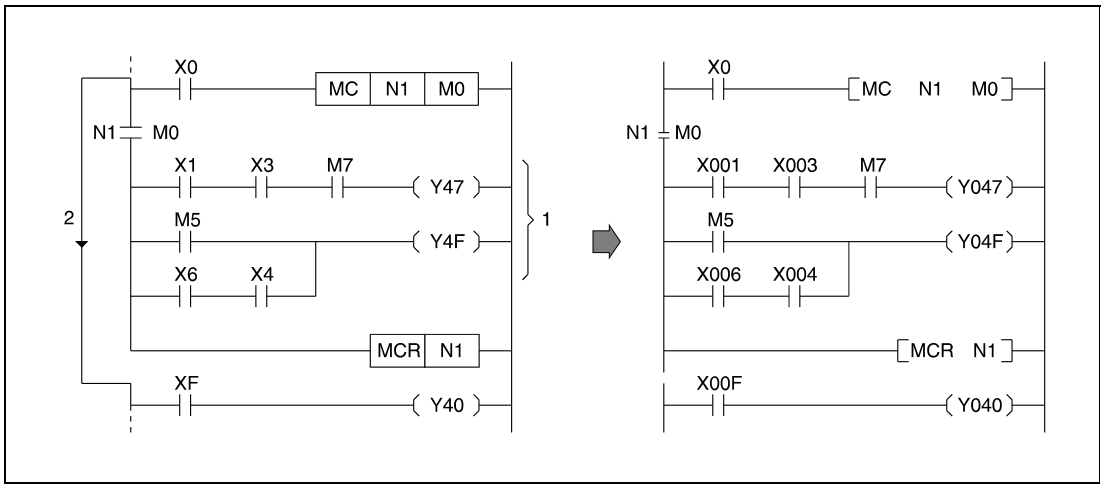
General notes

The MC instruction is applied to create highly efficient ladder switching sequence programs. After setting the input condition, the program part between the destination d and the MCR instruction is executed. The master control regions are distinguished by nesting (N). Nesting can be performed from N0 through N14.

Since the GX IEC Developer Software does not allow a vivid programming of the MC/MCR instruction, here the ladder diagrams of the GX Works2 Software are shown as an illustration.

The ladder diagram illustrates the function of the MC instruction. If the input X0 is reset, the program part in level 1 (designated by N1) is skipped (1). If X0 is set, the program part from N1 to the MCR instruction is executed (2).

When programming in the ladder mode, it is not necessary to input MC contacts on the vertical bus. These are displayed automatically.



MC Activating indicated program parts

The MC instruction is the start instruction for master control to process a specified program part. If the input condition of the MC instruction is set, the devices between the MC and the MCR instruction are processed regularly.

The devices between the MC and the MCR instruction are even processed after the input condition of the MC instruction is reset. Therefore, the program scan time in this case is not decreased. When the input condition is reset, the devices between the MC and the MCR instruction are processed as follows:

Devices	Processing
10 ms timer 100 ms timer	Count value setting is reset to 0. Input and output contacts are reset (0).
Retentive 10 ms timer Retentive 100 ms timer Counter	Count value setting and condition of input contacts remained. Output contact is reset (0).
Devices in the OUT instruction	All outputs are reset.
Devices in the SET, RST, and SFT instruction	Actual status remained.

NOTE *If an instruction that does not require any input condition (e.g. FOR/NEXT, EI, DI) is placed between the MC and MCR instructions, this instruction is executed by the PLC without regard to the input condition of the MC instruction.*

For one MC instruction, identical nesting levels n are allowed, provided that different numbers (addresses) of devices are set.

After setting the MC instruction the device designated by d is set. If this device is designated as input condition elsewhere in the program, the contacts are processed as double contacts and set or reset in parallel. Therefore, the device designated by d should not be used within other instructions.

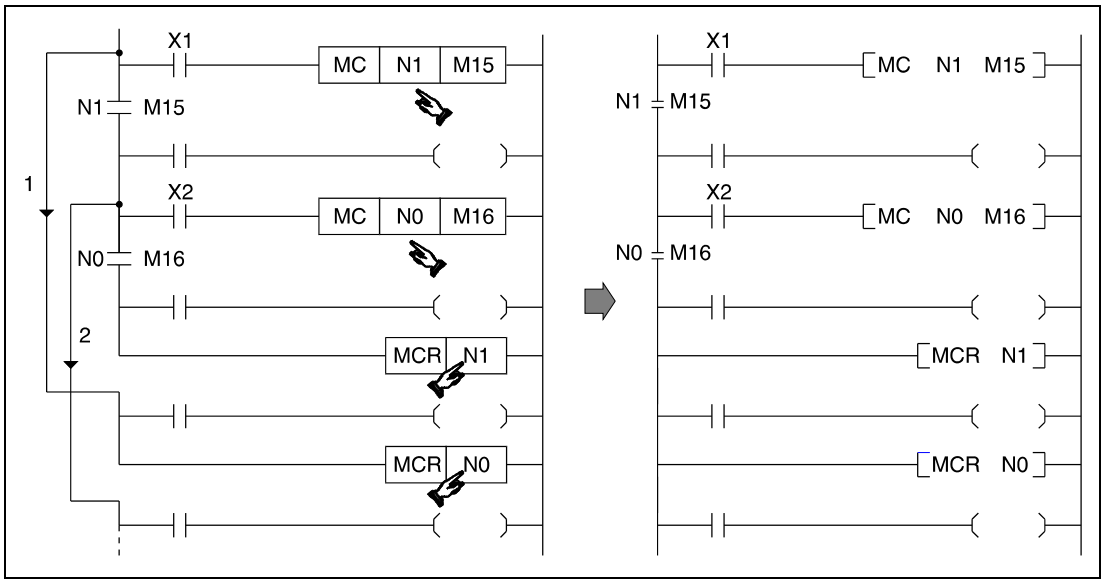
MCR Deactivating indicated program parts

The MCR instruction resets the MC instruction and indicates the end of the program part for master control.

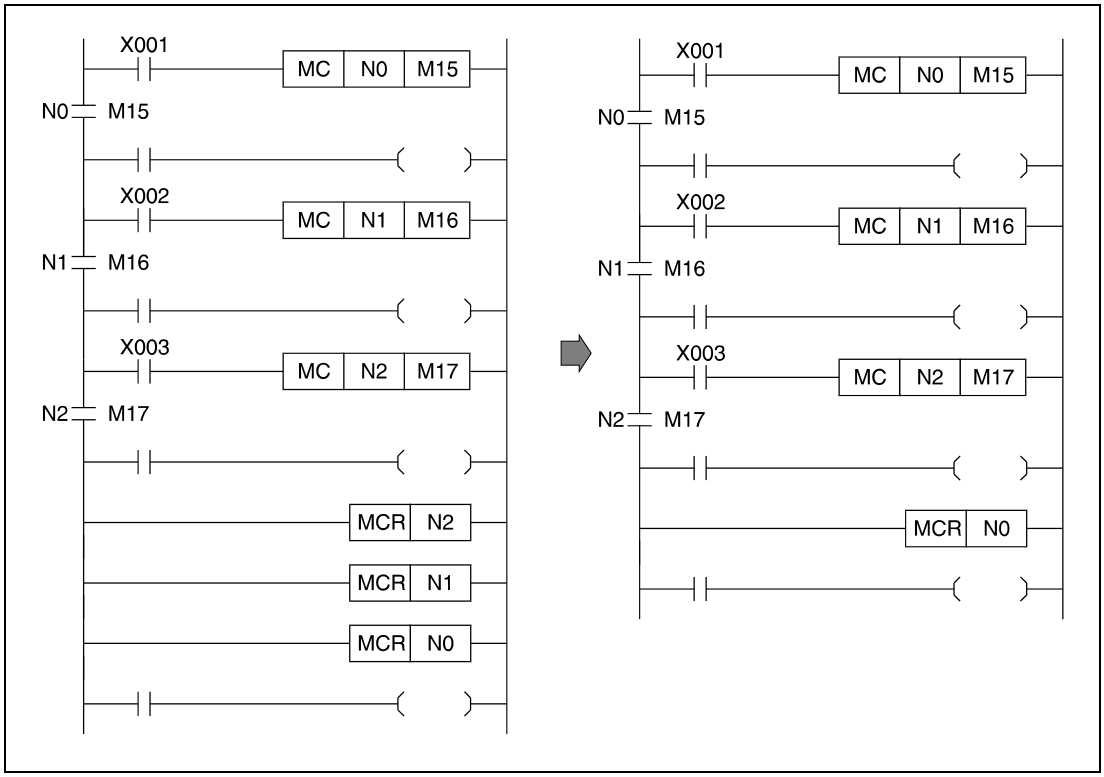
The MCR instruction must not be set via an input contact.

Notes on programming nesting numbers (addresses):

Nesting can be performed from N0 to N14. The first master control region designated by the MC instruction has to start with the lowest nesting address and the first MCR instruction has to start with the highest nesting address. If nesting addresses are designated in a different order, the nesting levels (1, 2) are not processed accurately by the PLC. The following diagram illustrates this case.



If several MCR instructions are progammed consecutively, the program can be shortened by placing one MCR instruction only with the lowest nesting address to finish all MC program parts.

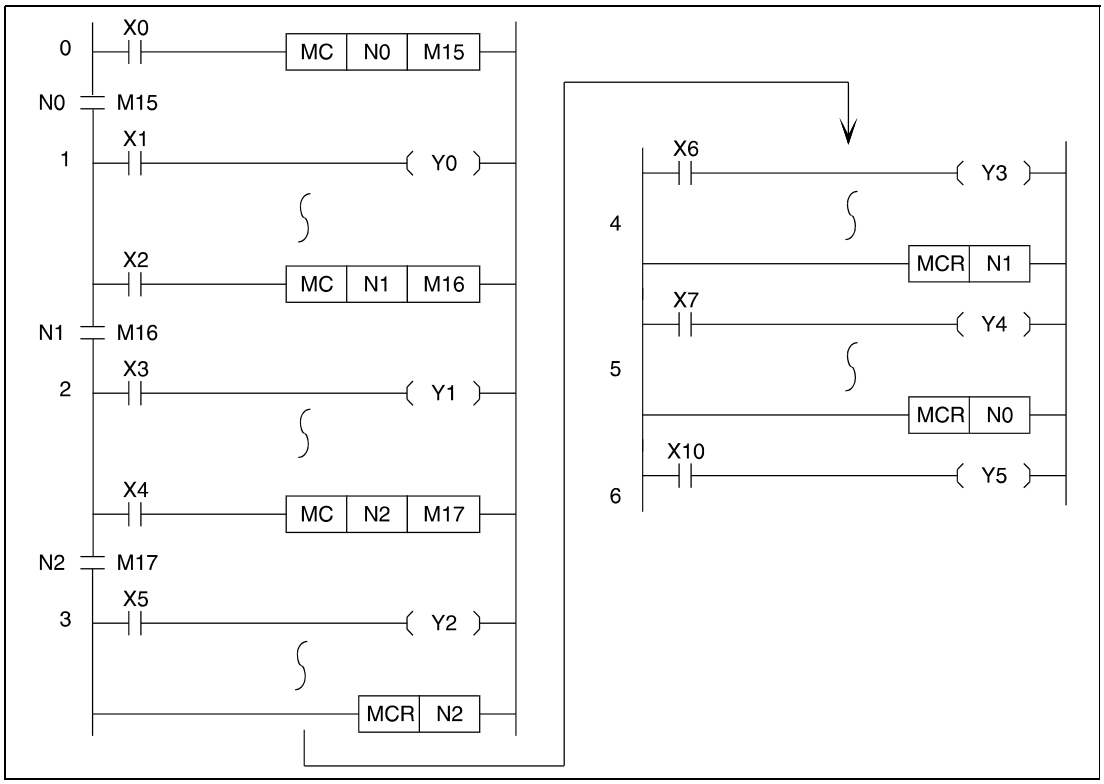


Program Example

MC, MCR

The MC instruction designates a nesting address N to specify the nesting level. Nesting addresses can be designated within N0 to N14.

The nesting addresses determine the execution sequence of MC program parts. The following program illustrates designation of different execution levels by nesting addresses. For better comprehensibility the GX Works2 ladder diagram is shown:

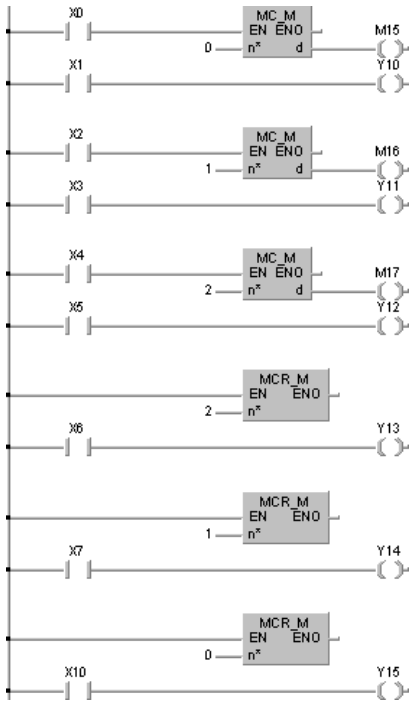


In addition the GX IEC Developer ladder diagram is shown:

MELSEC Instruction List

```
MELSEC
LD    X0
MC    N0
      M15
LD    X1
OUT   Y10

LD    X2
MC    N1
      M16
LD    X3
OUT   Y11

LD    X4
MC    N2
      M17
LD    X5
OUT   Y12

MCR   N2
LD    X6
OUT   Y13

MCR   N1
LD    X7
OUT   Y14

MCR   N0
LD    X10
OUT   Y15
```

Ladder Diagram



IEC Instruction List

```
LD    X0
MC_M  N0 , M15
LD    X1
ST    Y10

LD    X2
MC_M  N1 , M16
LD    X3
ST    Y11

LD    X4
MC_M  N2 , M17
LD    X5
ST    Y12

MCR_M N2
LD    X6
ST    Y13

MCR_M N1
LD    X7
ST    Y14

MCR_M N0
LD    X10
ST    Y15
```

5.6 Termination instructions

5.6.1 FEND

NOTE *This instruction should not be used within the IEC editors.*

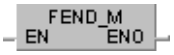
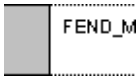
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

GX Works2

Variables

Set Data	Meaning	Data Type
—	—	—

Functions **End of main routine program**

FEND End of program branches

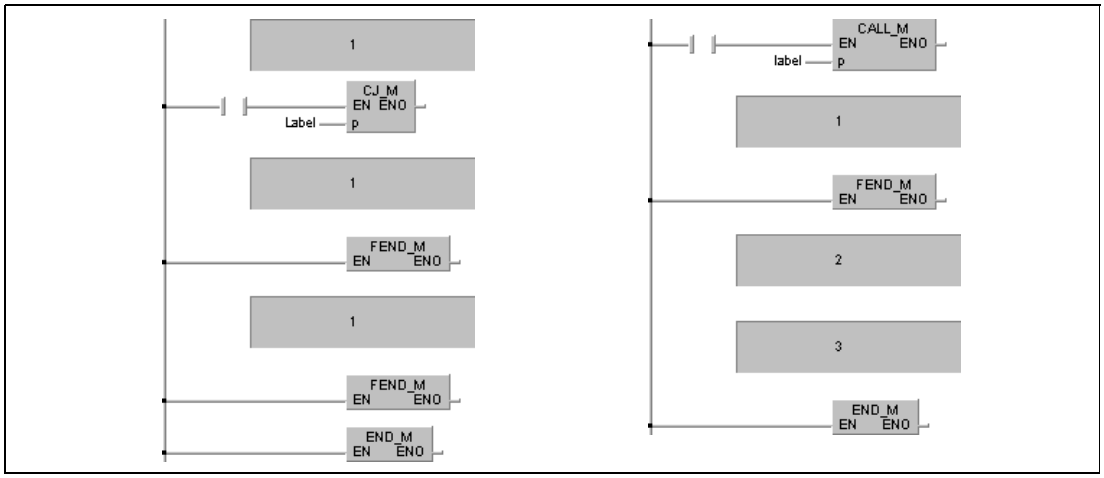
The FEND instruction specifies the end of a program branch. This branch can either be a main routine program or a subroutine program.

After execution of the FEND instruction the program jumps to the END instruction. The execution of internal processes like timer/counter processing or CPU self-diagnostics check begin at program step 1 again.

The program example on the left shows the termination of program branches invoked via the CJ (conditional jump) instruction.

After execution of the CJ instruction the invoked program part is executed up to the next FEND instruction. Without execution of the CJ instruction the program jumps back to program step 0 after the next FEND instruction.

The program example on the right shows the execution of the FEND instruction in order to split a main routine program from a sub-routine or interrupt program.



[1] Main routine program

[2] Subroutine program

[3] Interrupt program

NOTE *In the instruction list of the GX Works2 the FEND instruction has to be programmed by the user. After this program organization unit has been processed no further one will be executed because it would follow the FEND instruction.*

Alternatively to this programming the IEC editor can be used. In that case the FEND instruction would be set by the GX IEC Developer *compiler automatically.*

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The FEND instruction is executed after a CALL, FCALL, ECALL, or EFCALL instruction and before a RET instruction. (Error code 4211)
- The FEND instruction is executed after a FOR instruction and before a NEXT instruction. (Error code 4200)
- The FEND instruction is executed during an interrupt program and before an IRET instruction. (Error code 4221)
- The FEND instruction is executed after a CHKCIR instruction and before a CHKEND instruction. (Error code 4230)
- The FEND instruction is executed after an IX instruction and before an IXEND instruction. (Error code 4231)

5.6.2 END

NOTE *This instruction should not be used within the IEC editors.*

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

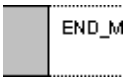
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				U
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

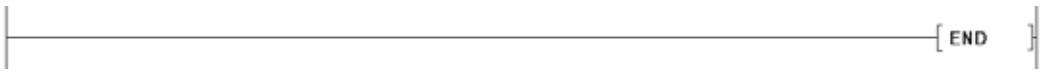
MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2



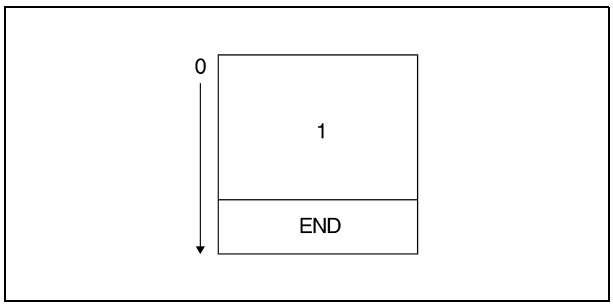
Variables

Set Data	Meaning	Data Type
—	—	—

Functions **End of sequence program**

END End of sequence program

The END instruction specifies the end of a program. Executing the END instruction the program jumps back to program step 0.

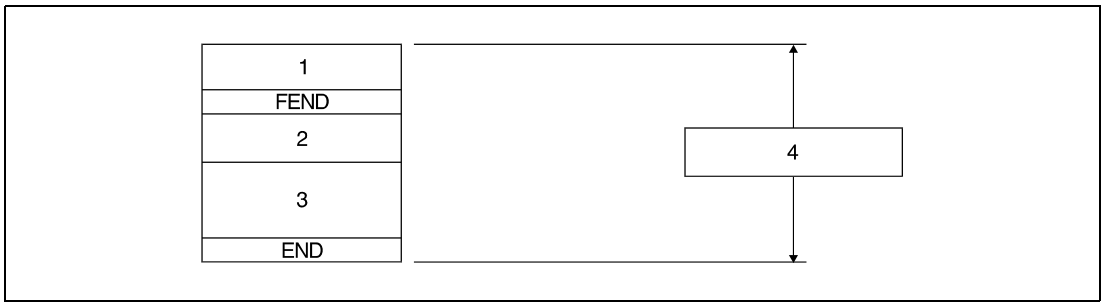


[1]Sequence program

The END instruction cannot be applied in a program routine. A program routine is terminated by the FEND instruction.

If the END instruction is missing in a program an error message is returned when starting the program, and the program execution is terminated by the PLC. Without the END instruction operation errors even occur, if the capacity of a subprogram is set by parameters.

The following diagram illustrates appropriate programming of the END and FEND instruction:



[1] Main routine program
[2] Subroutine program
[3] Interrupt program
[4] Sequence program

NOTE *The FEND instruction will be set by both the GX IEC Developer and GX Works2 automatically.*

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The jump destination of a CJ, SCJ, or JMP instruction is allocated after the END instruction.
- A subprogram or interrupt routine allocated after the END instruction is called.
- The END instruction is executed after a CALL, FCALL, ECALL, or EFCALL instruction and before a RET instruction. (Error code 4211)
- The END instruction is executed after a FOR instruction and before a NEXT instruction. (Error code 4200)
- The END instruction is executed during an interrupt program and before an IRET instruction. (Error code 4221)
- The END instruction is executed after a CHKCIR instruction and before a CHKEND instruction. (Error code 4230)
- The END instruction is executed after an IX instruction and before an IXEND instruction. (Error code 4231)

5.7 Miscellaneous instructions

5.7.1 STOP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				U
—	—	—	—	—	—	—	—	—	—

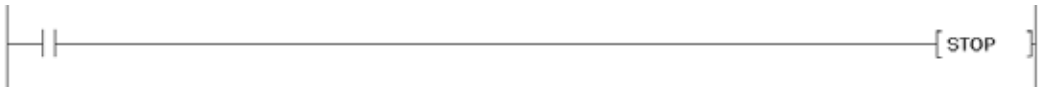
GX IEC Developer

MELSEC Instruction List

Ladder Diagram

IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
—	—	—

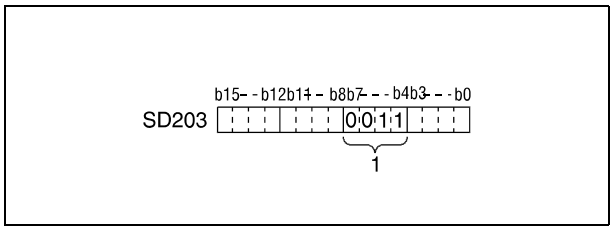
Functions

Sequence program stop

STOP Stop instruction

If the input condition of the STOP instruction is set, all outputs (Y) are reset and all operations of the PLC are terminated. The STOP instruction has the same function as the STOP position of the RUN/STOP key switch on the CPU.

On execution of the STOP instruction the bit b4 through bit b7 in special register SD203 store the binary value 3.



[1] Binary value 3

In order to restart the operation of the PLC the RUN/STOP switch has to be switched to STOP and then to RUN again.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

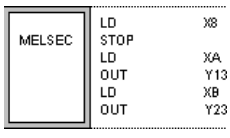
- The STOP instruction is executed after a CALL, FCALL, ECALL, EFCALL or XCALL instruction and before a RET instruction. (Error code 4211)
- The STOP instruction is executed after a FOR instruction and before a NEXT instruction. (Error code 4200)
- The STOP instruction is executed during an interrupt program and before an IRET instruction. (Error code 4221)
- The STOP instruction is executed after a CHKCIR instruction and before a CHKEND instruction. (Error code 4230)
- The STOP instruction is executed after an IX instruction and before an IXEND instruction. (Error code 4231)
- The STOP instruction was executed during the fixed scan execution type program. (For the Universal model QCPU and LCPU only) (Error code 4223)

Program Example

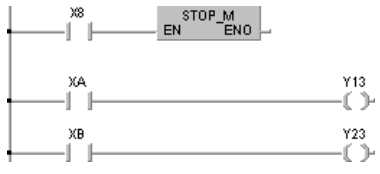
STOP

If X8 is set the following program terminates operation. All following program steps are executed after switching the RUN/STOP switch to STOP and to RUN again.

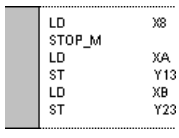
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



5.7.2 NOP, NOPLF, PAGE n

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

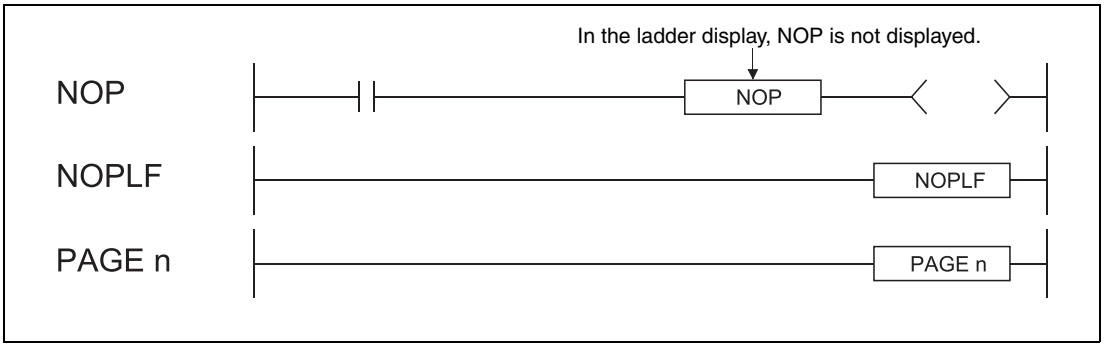
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
MELSEC NOP	—	—

NOTE *The NOP instruction does not work with the IEC editors. The only way to program this instructions is by using the MELSEC instruction list.*

GX Works2



Variables

Set Data	Meaning	Data Type
—	—	—

Functions **No operation program step**

NOP No operation program step

The NOP instruction is a **n**o-operation instruction that does not affect any other operations or program parts. The NOP instruction creates an empty logical program step that can be replaced by other program instructions during the development of a new program.

The NOP instruction is especially suitable for the following cases:

- To provide space for debugging sequence programs.
- To delete an instruction (over write it) without changing the number of steps.
- To delete an instruction temporarily for later editing.

NOTE *After finishing program editing the NOP instructions should be deleted where possible in order to shorten program scan time.*

NOPLF To change pages during printouts

The NOPLF instruction is a no-operation instruction that does not affect any other operations or program parts. The NOPLF is used when printing from a peripheral device to force a page change at any desired location.

When printing ladders:

- A page break will be inserted between ladder blocks with the presence of the NOPLF instruction.
- The ladder cannot be displayed correctly if an NOPLF instruction is inserted in the midst of a ladder block.
  Do not insert an NOPLF instruction in the midst of a ladder block.

When printing instruction lists:

- The page will be changed after the printing of the NOPLF instruction.

Refer to the Operating Manual for the peripheral device in use for details of printouts from peripheral devices.

PAGE n Subsequent programs will be controlled from step 0 of page n

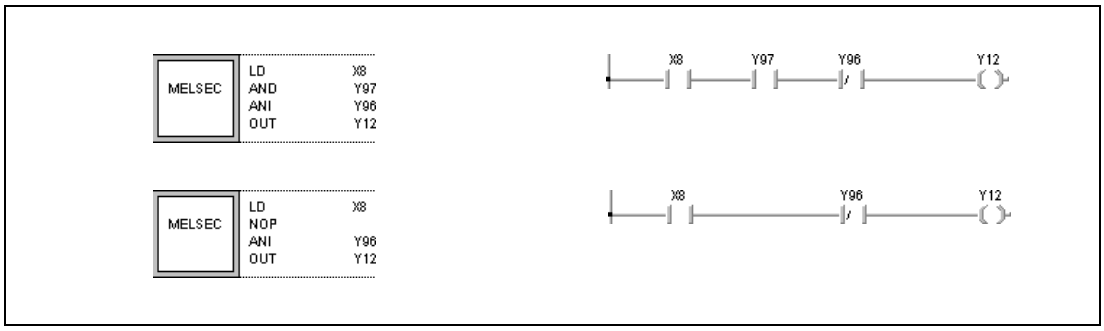
This is a no operation instruction that has no impact on any operations up to that point.

No processing is performed at peripheral devices with this instruction.

Program Example 1

NOP

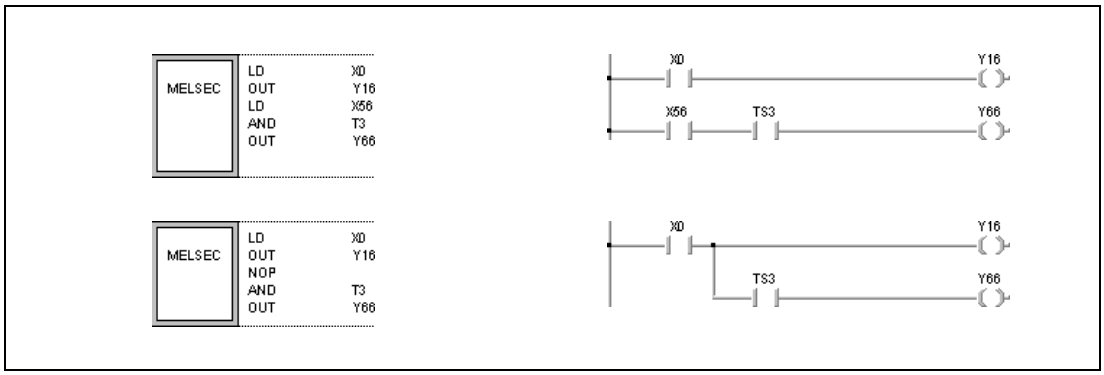
The following program contains a NOP instruction to replace the contact connection AND for debugging purposes.



Program Example 2

NOP

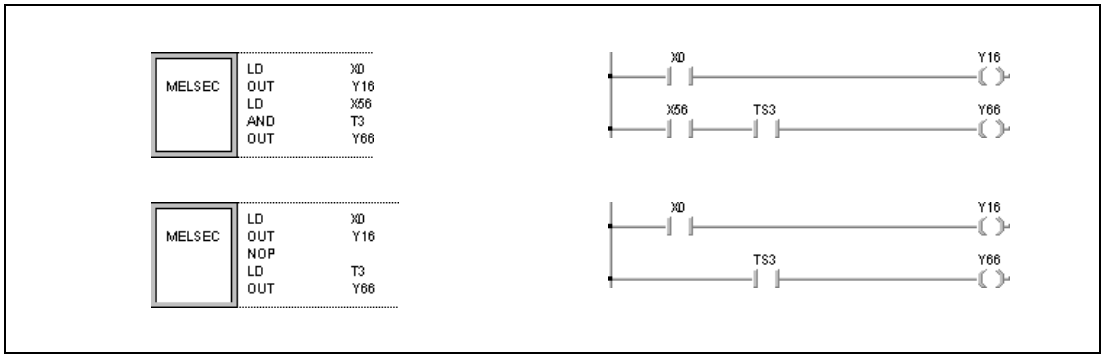
The following program example contains a NOP instruction to replace an LD instruction.



Program Example 3

NOP

The following program example contains a NOP instruction to replace an LD instruction.

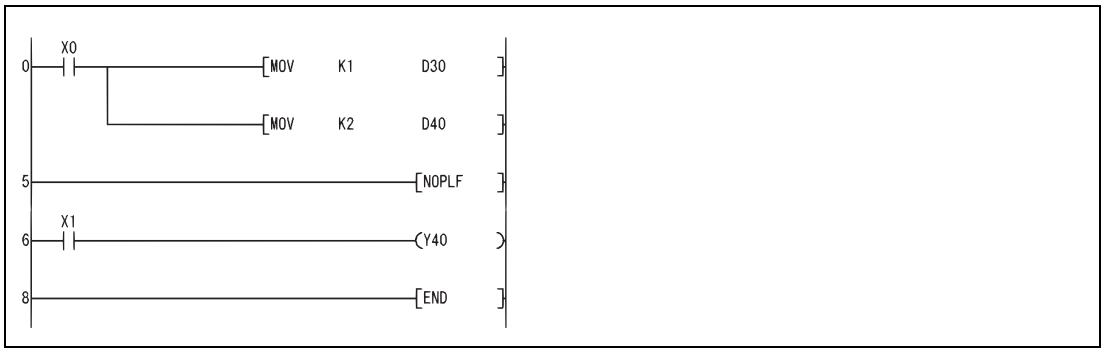


NOTE *Input contacts (LD, LDI) should be replaced by a NOP instruction carefully, because the logical structure of the program is changed considerably.*

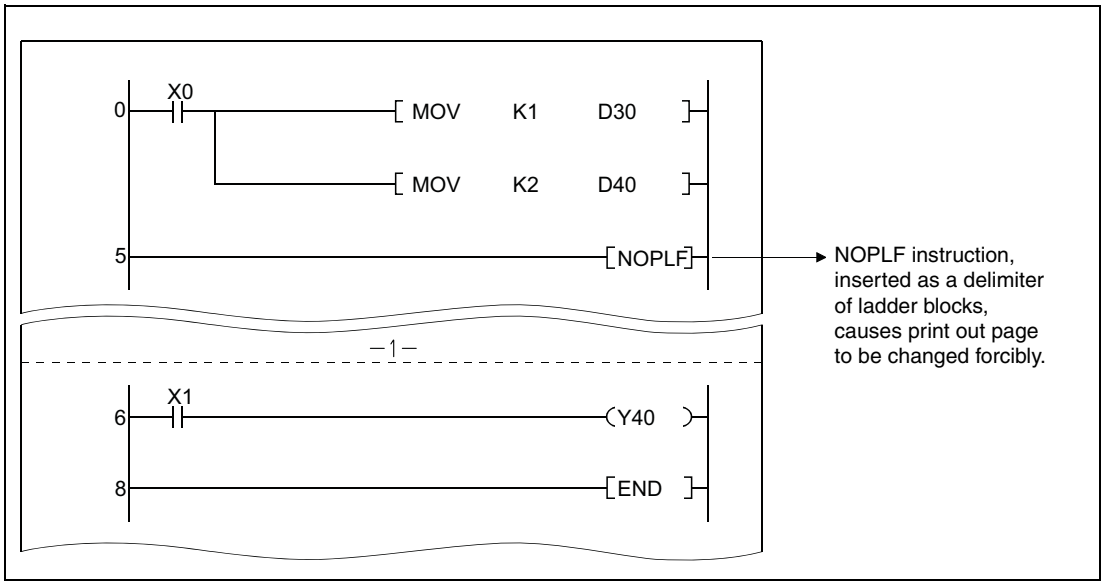
Program Example 4

NOPLF

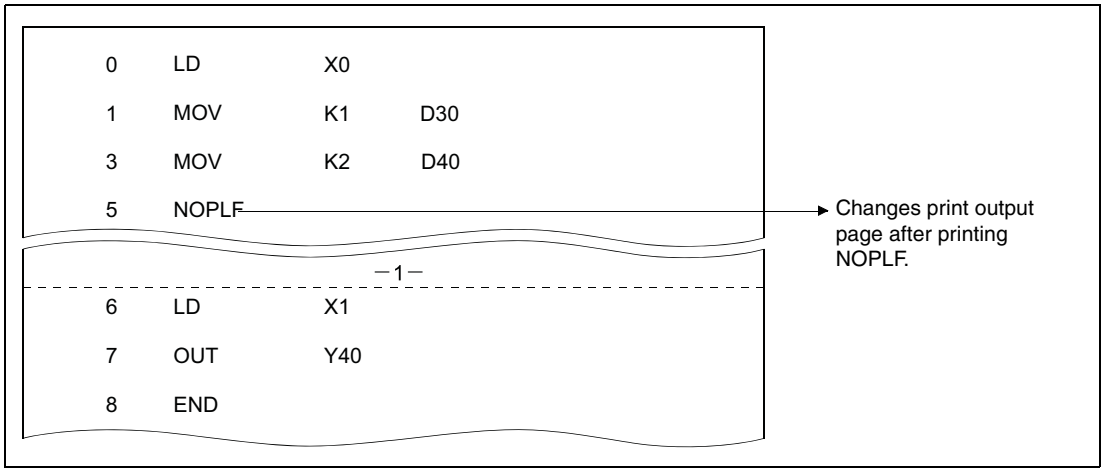
The following program example shows the results of a NOPLF instruction.



Printing the ladder will result in the following:

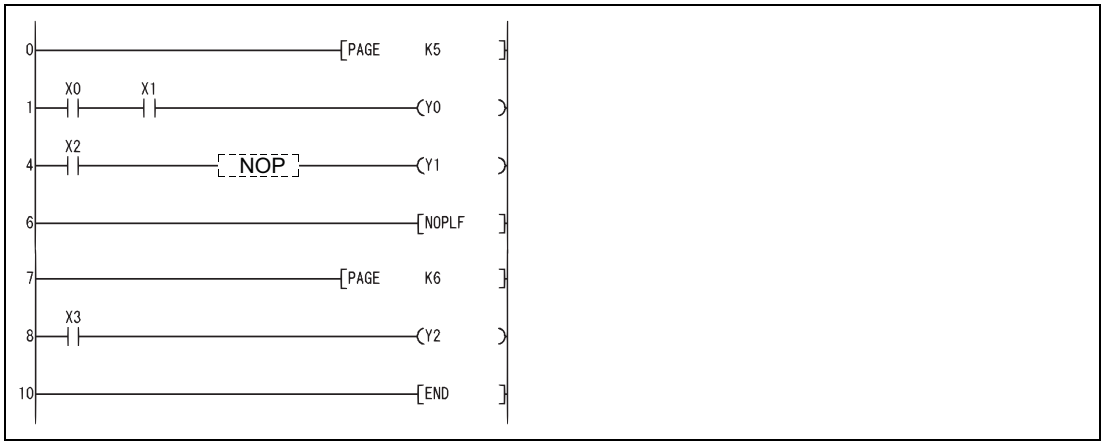


Printing an instruction list with the NOPLF instruction will result in the following:



Program Example 5

PAGE n



6 Application Instructions, Part 1

The application instructions, part 1 comprise instructions that process numerical 16-bit and 32-bit data, floating point data, and character string data. Commonly, these basic instructions perform comparison and arithmetic operations.

Instruction	Meaning
Comparison operation instruction	Compares data to data (e.g. =, >, ≥)
Arithmetic operation instruction	Adds, subtracts, multiplies, divides, increments, and decrements BIN and BCD data, floating point data, and BIN block data Links character strings
Data conversion instruction	Converts data types (e.g. BCD → BIN, BIN → BCD)
Data transfer instruction	Transmits designated data
Program branch instruction	Program jump commands
Program execution control instruction	Enables and disables program interrupts
Refresh instruction	Refreshes bit devices, links, and I/O interfaces
Other convenient instructions	Count 1- or 2-phase input up or down, teaching timer, special function timer, rotary table near path rotation control, ramp signal, pulse density measurement, fixed cycle pulse output, pulse width modulation, matrix input

6.1 Comparison operation instructions

Comparison operation instructions compare data values (e.g. equal to =, greater than >, less than <). Programming the comparison operation instructions is similar to the corresponding basic instructions:

LD, LDI ⇒ LD=, LDD=

AND, ANI ⇒ AND=, ANDD=

OR, ORI ⇒ OR=, ORD=

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor	Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
= equal	LD=	LD_EQ_M	≤ less equal	LD<=	LD_LE_M
	AND=	AND_EQ_M		AND<=	AND_LE_M
	OR=	OR_EQ_M		OR<=	OR_LE_M
	LDD=	LDD_EQ_M		LDD<=	LDD_LE_M
	ANDD=	ANDD_EQ_M		ANDD<=	ANDD_LE_M
	ORD=	ORD_EQ_M		ORD<=	ORD_LE_M
	LDE=	LD_EEQ_M		LDE<=	LD_ELE_M
	ANDE=	AND_EEQ_M		ANDE<=	AND_ELE_M
	ORE=	OR_EEQ_M		ORE<=	OR_ELE_M
	LDED=			LDED<=	
	ANDED=			ANDED<=	
	ORED=			ORED<=	
	LD$=	LD_STRING _EQ_M		LD$<=	LD_STRING _LE_M
	AND$=	AND_STRING _EQ_M		AND$<=	AND_STRING _LE_M
	OR$=	OR_STRING _EQ_M		OR$<=	OR_STRING _LE_M
	BKCMP=	BKCMP_EQ_M		BKCMP<=	BKCMP_LE_M
	BKCMP=P	BKCMP_EQP_M		BKCMP<=P	BKCMP_LEP_M
	DBKCMP=			DBKCMP<=	
	DBKCMP=P			DBKCMP<=P	

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor	Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
≠ not equal	LD<>	LD_NE_M	< less than	LD<	LD_LT_M
	AND<>	AND_NE_M		AND<	AND_LT_M
	OR<>	OR_NE_M		OR<	OR_LT_M
	LDD<>	LDD_NE_M		LDD<	LDD_LT_M
	ANDD<>	ANDD_NE_M		ANDD<	ANDD_LT_M
	ORD<>	ORD_NE_M		ORD<	ORD_LT_M
	LDE<>	LD_ENE_M		LDE<	LD_ELT_M
	ANDE<>	AND_ENE_M		ANDE<	AND_ELT_M
	ORE<>	OR_ENE_M		ORE<	OR_ELT_M
	LDED<>			LDED<	
	ANDED<>			ANDED<	
	ORED<>			ORED<	
	LD$<>	LD_STRING _NE_M		LD$<	LD_STRING _LT_M
	AND$<>	AND_STRING _NE_M		AND$<	AND_STRING _LT_M
	OR$<>	OR_STRING _NE_M		OR$<	OR_STRING _LT_M
	BKCMP<>	BKCMP_NE_M		BKCMP<	BKCMP_LT_M
	BKCMP<>P	BKCMP_NEP_M		BKCMP<P	BKCMP_LTP_M
	DBKCMP<>			DBKCMP<	
	DBKCMP<>P			DBKCMP<P	
> greater	LD>	LD_GT_M	≥ greater equal	LD>=	LD_GE_M
	AND>	AND_GT_M		AND>=	AND_GE_M
	OR>	OR_GT_M		OR>=	OR_GE_M
	LDD>	LDD_GT_M		LDD>=	LDD_GE_M
	ANDD>	ANDD_GT_M		ANDD>=	ANDD_GE_M
	ORD>	ORD_GT_M		ORD>=	ORD_GE_M
	LDE>	LD_EGT_M		LDE>=	LD_EGE_M
	ANDE>	AND_EGT_M		ANDE>=	AND_EGE_M
	ORE>	OR_EGT_M		ORE>=	OR_EGE_M
	LDED>			LDED>=	
	ANDED>			ANDED>=	
	ORED>			ORED>=	
	LD$>	LD_STRING _GT_M		LD$>=	LD_STRING _GE_M
	AND$>	AND_STRING _GT_M		AND$>=	AND_STRING _GE_M
	OR$>	OR_STRING _GT_M		OR$>=	OR_STRING _GE_M
	BKCMP>	BKCMP_GT_M		BKCMP>=	BKCMP_GE_M
	BKCMP>P	BKCMP_GTP_M		BKCMP>=P	BKCMP_GEP_M
	DBKCMP>			DBKCMP>=	
	DBKCMP>P			DBKCMP>=P	

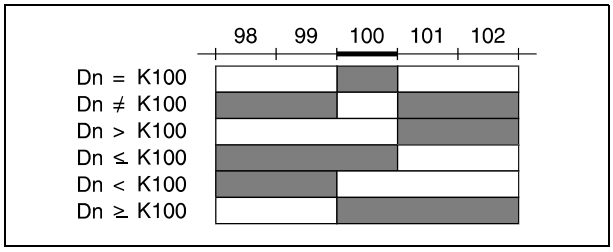
NOTE *Within the IEC editors please use the IEC commands.*

IEC Commands

Function	IEC Command	Meaning
=	EQ	Equal
<>	NE	Not equal
<=	LE	Less equal
<	LT	Less than
>=	GE	Greater equal
>	GT	Greater than

Execution Conditions

The following illustration shows the execution conditions for the various comparison operation instructions.



■ = 1 = ON

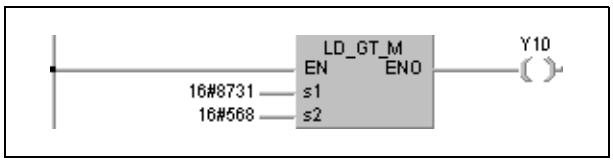
☐ = 0 = OFF

NOTE *When s1 and s2 are assigned by a hexadecimal constant and the numerical value (8 to F) whose most significant bit (b15) is "1" is designated as a constant, the value is considered as a negative BIN value in comparison operation.*

The result of the comparison operation 16#8000 > 16#7999 is FALSE (0), although TRUE (1) would be expected. The values are converted to BIN data and therefore bit 15 (b15) is set. If bit 15 is set, the value becomes negative.

Program Example 1

Comparison of two-digit BCD values:

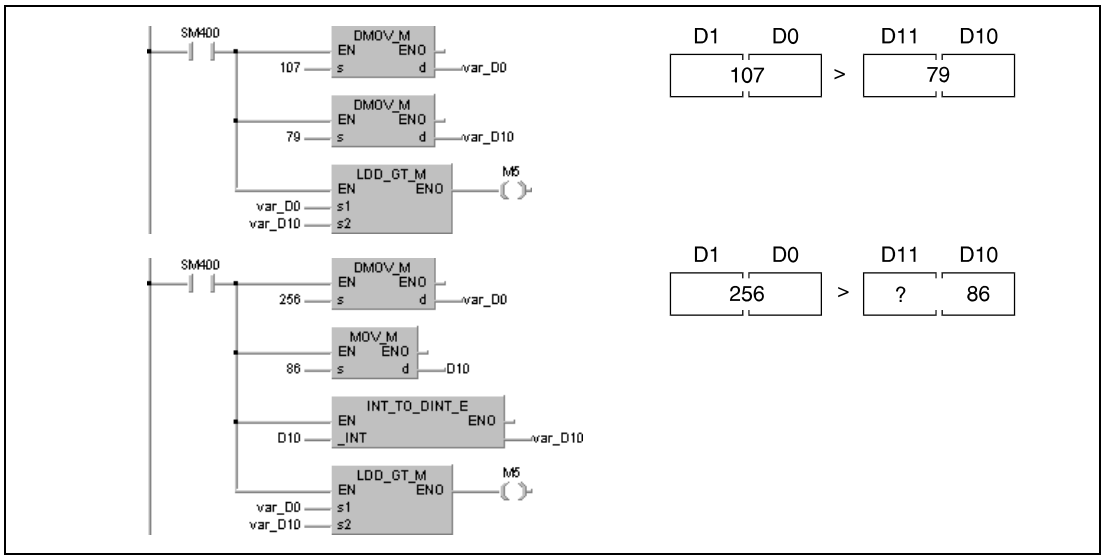


8731H is processed as -30927 and 568H as 1384. The comparison operation then is -30927 > 1384 and Y10 is not set.

NOTE *For comparison operation instructions with 32-bit data, the numerical input value has to be determined by a 32-bit instruction like DMOV. The instruction will not be carried out correctly, if the value was determined by a 16-bit instruction like MOV, because a 32-bit instruction always applies the n and (n+1) data value.*

Program Example 2

Comparison instruction with 32-bit data:



The example shows two comparison operations with 32-bit data. The first program sets M5, because both values are determined by the 32-bit instruction DMOV.

The second program has no definite result, because the value in the upper bytes is not defined definitely.

NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.1.1 =, < >, >, < =, <, > =

CPU

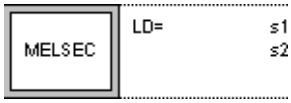
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

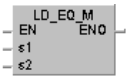
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—

GX IEC Developer

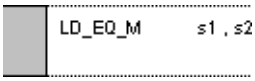
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
s1	Comparative data, or device storing comparative data	BIN 16-bit
s2		

Functions

BIN 16-bit data comparison

=, <>, >, <=, <, >= Comparison operation instructions

A 16-bit comparison operation instruction consists of the instruction itself and two designated devices s1 and s2 to be compared.

The result of the comparison operation is binary ("1" if the comparison is true, "0" if the comparison is false). The binary result can be processed as a logical connection result.

The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol	Comparison Operation Results	
	1	0
=	s1 = s2	s1 ≠ s2
<>	s1 ≠ s2	s1 = s2
>	s1 > s2	s1 ≤ s2
<=	s1 ≤ s2	s1 > s2
<	s1 < s2	s1 ≥ s2
>=	s1 ≥ s2	s1 < s2

NOTE

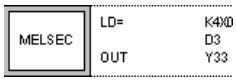
When s1 and s2 are assigned by a hexadecimal constant and the numerical value (8 to F) whose most significant bit (b15) is "1" is designated as a constant, the value is considered as a negative BIN value in comparison operation.

Program Example 1

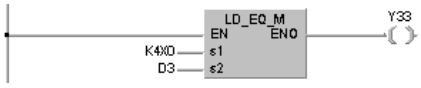
Comparison operation instruction =

The following program compares the data at X0 to XF with the data in D3. It turns ON Y33, if the data are equal.

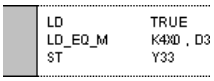
MELSEC Instruction List



Ladder Diagram



IEC Instruction List

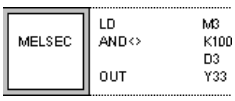


Program Example 2

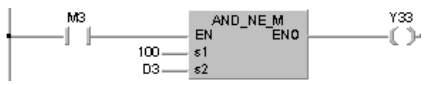
Comparison operation instruction <>

The following program compares BIN value 100 to the data in D3. It turns ON Y33, if the data in D3 is not equal to 100.

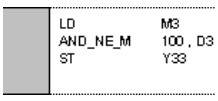
MELSEC Instruction List



Ladder Diagram



IEC Instruction List

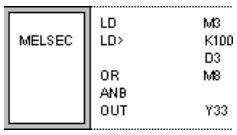


Program Example 3

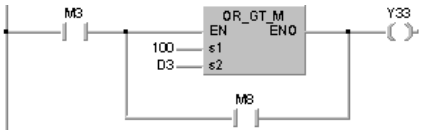
Comparison operation instruction >

The following program compares BIN value 100 to the data in D3. It turns ON Y33, if the data in D3 is less than 100 and M3 is set. Y33 is also switched ON, if M8 and M3 are set.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

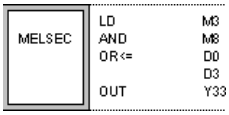
```
LD      M3
AND(
        M8
OR_GT_M 100 , D3
)
ST      Y33
```

Program Example 4

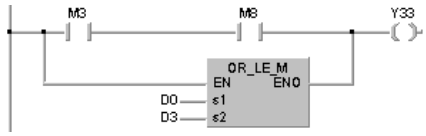
Comparison operation instruction <=

The following program compares the data in D0 to the data in D3. It turns ON Y33, if the data in D0 is less than or equal to D3. Y33 is also switched ON, if M8 and M3 are set.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD      M3
AND     M8
OR_LE_M D0 , D3
ST      Y33
```

6.1.2 D=, D<>, D>, D<=, D<, D>=

CPU

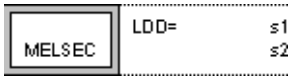
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

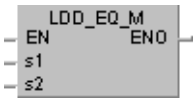
	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—

GX IEC Developer

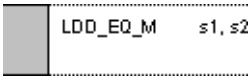
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
s1	Comparative data, or device storing comparative data.	BIN 32-bit
s2		

Functions

BIN 32-bit data comparison

D=, D<>, D>, D<=, D<, D>= Comparison operation instructions

A 32-bit comparison operation instruction consists of the instruction itself and two designated devices s1 and s2 to be compared.

The result of the comparison operation is binary ("1" if the comparison is true, "0" if the comparison is false). The binary result can be processed as a logical connection result.

The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol	Comparison Operation Results	
	1	0
D=	s1 = s2	s1 ≠ s2
D<>	s1 ≠ s2	s1 = s2
D>	s1 > s2	s1 ≤ s2
D<=	s1 ≤ s2	s1 > s2
D<	s1 < s2	s1 ≥ s2
D>=	s1 ≥ s2	s1 < s2

NOTE

When s1 and s2 are assigned by a hexadecimal constant and the numerical value (8 to F) whose most significant bit (b15) is "1" is designated as a constant, the value is considered as a negative BIN value in comparison operation.

Program Example 1

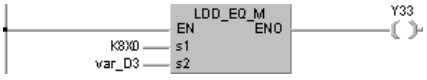
Comparison operation instruction D=

The following program compares the data at X0 to X1F with the data in D3 and D4. It turns ON Y33 if the data are equal.

MELSEC Instruction List

```
MELSEC  LDD=   K8X0
        OUT    D3
               Y33
```

Ladder Diagram



IEC Instruction List

```
LD      TRUE
LDD_EQ_M  K8X0 , var_D3
ST      Y33
```

Program Example 2

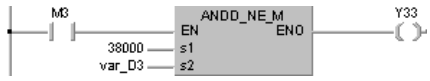
Comparison operation instruction D<>

The following program compares BIN value 38000 to the data in D3 and D4. It turns ON Y33, if M3 is set and the data in D3 and D4 are not equal to 38000.

MELSEC Instruction List

```
MELSEC  LD     M3
        ANDD<> K38000
               D3
        OUT    Y33
```

Ladder Diagram



IEC Instruction List

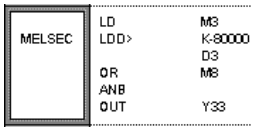
```
LD      M3
ANDD_NE_M  38000 , var_D3
ST      Y33
```

Program Example 3

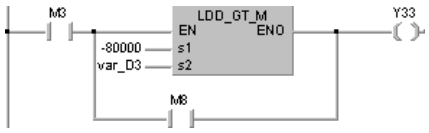
Comparison operation instruction D>

The following program compares BIN value -80000 to the data in D3 and D4. It turns ON Y33, if M3 is set and the data in D3 and D4 are less than -80000. Y33 is also switched ON, if M3 and M8 are set.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

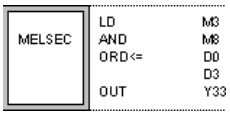
```
LD      M3
AND(
        M8
ORD_GT_M K-80000 , var_D3
)
ST      Y33
```

Program Example 4

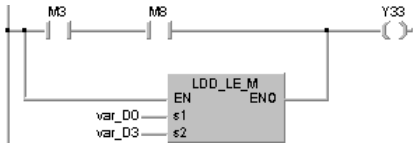
Comparison operation instruction D<=

The following program compares the data in D0 and D1 to the data in D3 and D4. Y33 is set, if the data in D3 and D4 are greater than or equal to D0 and D1. Y33 is also switched ON if M3 and M8 are set.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD      M3
AND     M8
ORD_LE_M var_D0 , var_D3
ST      Y33
```

NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.1.3 E=, E<>, E>, E< =, E<, E>=

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

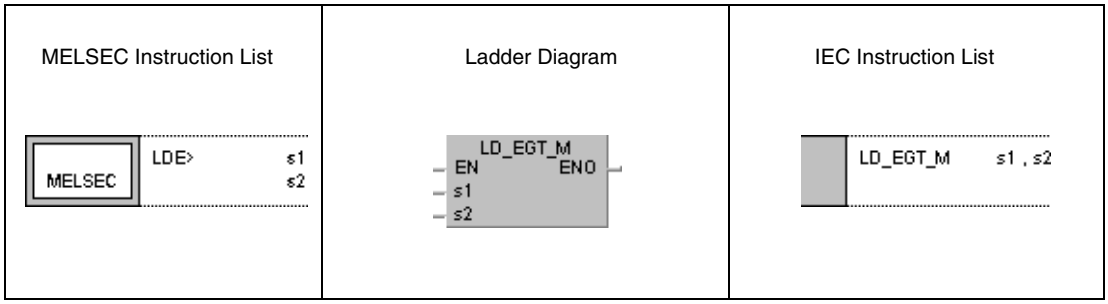
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

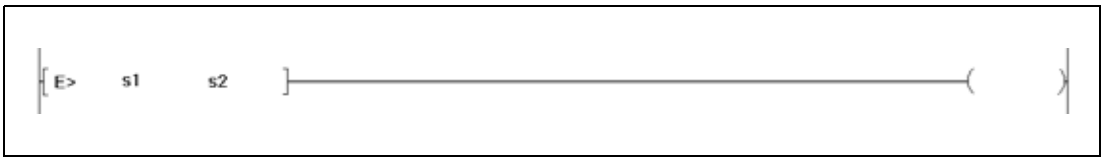
	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	●	●	●[1)]	●	—
s2	—	●	●	—	●	●	●[1)]	●	—

[1] Available only in multiple Universal model QCPU and LCPU

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
s1	Comparative data, or device storing comparative data.	Real number
s2		

Functions

Floating point data comparisons (Single precision)

E=, E<>, E>, E<=, E<, E>= Comparison operation instructions

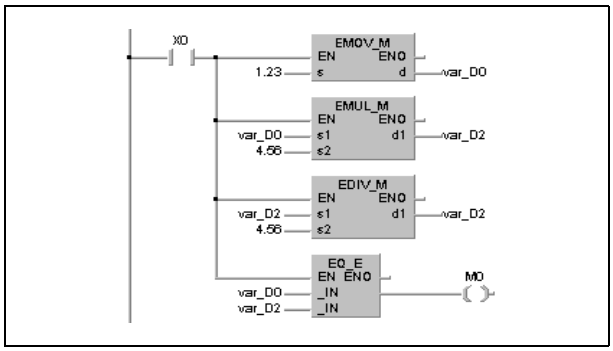
A comparison operation instruction for floating point data consists of the instruction itself and two designated devices s1 and s2 to be compared.

The result of the comparison operation is binary ("1" if the comparison is true, "0" if the comparison is false). The binary result can be processed as a logical connection result.

The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol	Comparison Operation Results	
	1	0
E=	s1 = s2	s1 ≠ s2
E<>	s1 ≠ s2	s1 = s2
E>	s1 > s2	s1 ≤ s2
E<=	s1 ≤ s2	s1 > s2
E<	s1 < s2	s1 ≥ s2
E>=	s1 ≥ s2	s1 < s2

NOTE *In some cases, rounding errors appear and floating point values that were equal before the comparison operation are not equal afterwards. In the following example M0 is not switched ON:*



NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the designated device is –0. (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code 4100)
  NOTE: There are CPU modules that will not result in an operation error if –0 is specified. Refer to section 3.5.1 for details.
- The value of the specified device is outside the following range:
  0, $\pm 2^{-126} \leq$ (Value) $< \pm 2^{128}$
  (For the Universal model QCPU, LCPU)
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or ± ∞.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example 1

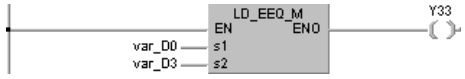
Comparison operation instruction E=

The following program compares floating point data in D0 and D1 to floating point data in D3 and D4. It turns ON Y33, if the data are equal.

MELSEC Instruction List

```
MELSEC  LDE=   D0
        OUT    D3
               Y33
```

Ladder Diagram



IEC Instruction List

```
LD          TRUE
LD_EEQ_M    var_D0 , var_D3
ST          Y33
```

Program Example 2

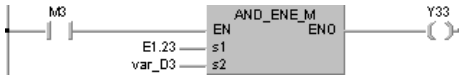
Comparison operation instruction E<>

The following program compares the floating point real number 1.23 to a floating point real number in D3 and D4. It turns ON Y33, if M3 is set and the data in D3 and D4 are not equal to 1.23.

MELSEC Instruction List

```
MELSEC  LD     M3
        ANDE<> E1.23
               D3
        OUT    Y33
```

Ladder Diagram



IEC Instruction List

```
LD          M3
AND_ENE_M   1.23 , var_D3
ST          Y33
```

Program Example 3

Comparison operation instruction E>

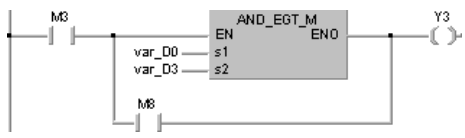
The following program compares floating point data in D0 and D1 to floating point data in D3 and D4. It turns ON Y3, if M3 is set and the data in D3 and D4 are less than the data in D0 and D1.

Y3 is also switched ON, if M3 and M8 are set.

MELSEC Instruction List

```
MELSEC  LD     M3
        LDE>   D0
               D3
        OR     M8
        ANB
        OUT    Y3
```

Ladder Diagram



IEC Instruction List

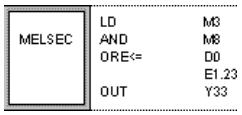
```
LD          M3
AND(
            M8
OR_EGT_M    var_D0 , var_D3
)
ST          Y3
```

Program Example 4

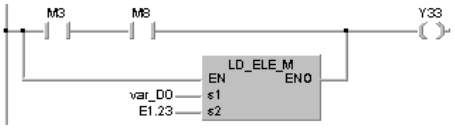
Comparison operation instruction E<=

The following example compares a floating point number in D0 and D1 to the floating point number 1.23. It turns ON Y33, if the data in D0 and D1 are less than or equal to 1.23. Y33 is also switched ON, if M3 and M8 are set.

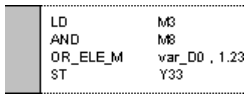
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.1.4 ED=, ED<>, ED>, ED< =, ED<, ED>=

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

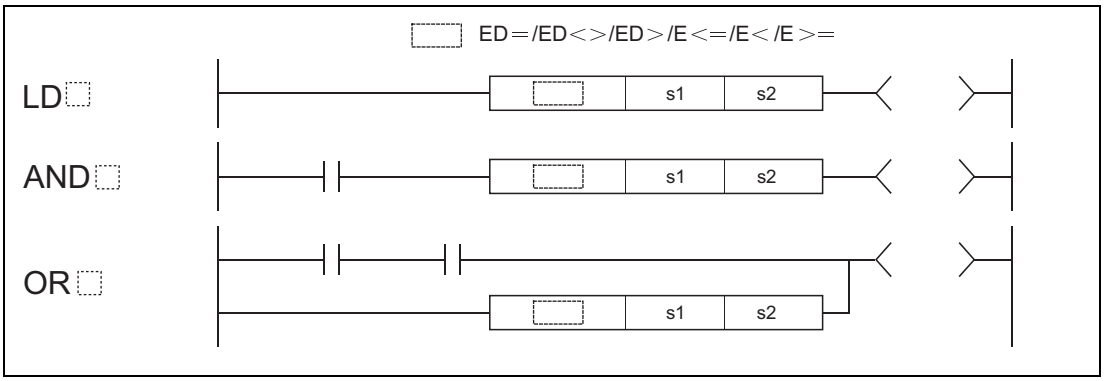
Devices

	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s1	Comparative data, or device storing comparative data.	Real number
s2		

Functions

Floating decimal point data comparisons (Double precision)

ED=, ED<>, ED>, ED< =, ED<, ED>= Comparison operation instructions

A comparison operation instruction for 64-bit floating point data consists of the instruction itself and two designated devices s1 and s2 to be compared.

The result of the comparison operation is binary ("1" if the comparison is true, "0" if the comparison is false). The binary result can be processed as a logical connection result.

The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol	Comparison Operation Results	
	1	0
ED=	s1 = s2	s1 ≠ s2
ED<>	s1 ≠ s2	s1 = s2
ED>	s1 > s2	s1 ≤ s2
ED<=	s1 ≤ s2	s1 > s2
ED<	s1 < s2	s1 ≥ s2
ED>=	s1 ≥ s2	s1 < s2

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is not within the following range:
  0, $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the designated device is –0.
  (Error code 4140)

Program Example 1

Comparison operation instruction ED=

The following program compares 64-bit floating decimal point real number data at D0 to D3 with 64-bit floating decimal point real number data at D4 to D7.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 2

Comparison operation instruction ED<>

The following program compares the floating decimal point real number 1.23 with the 64-bit floating decimal point real number data at D4 to D7.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 3

Comparison operation instruction ED>

The following program compares 64-bit floating decimal point real number data at D0 to D3 with 64-bit floating decimal point real number data at D4 to D7.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 4

Comparison operation instruction ED<=

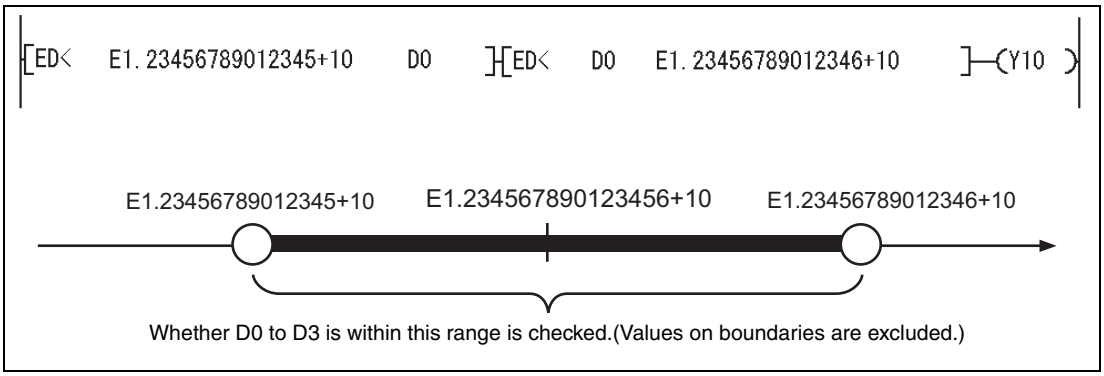
The following program compares the 64-bit floating decimal point data at D0 to D3 with the floating decimal point real number 1.23.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

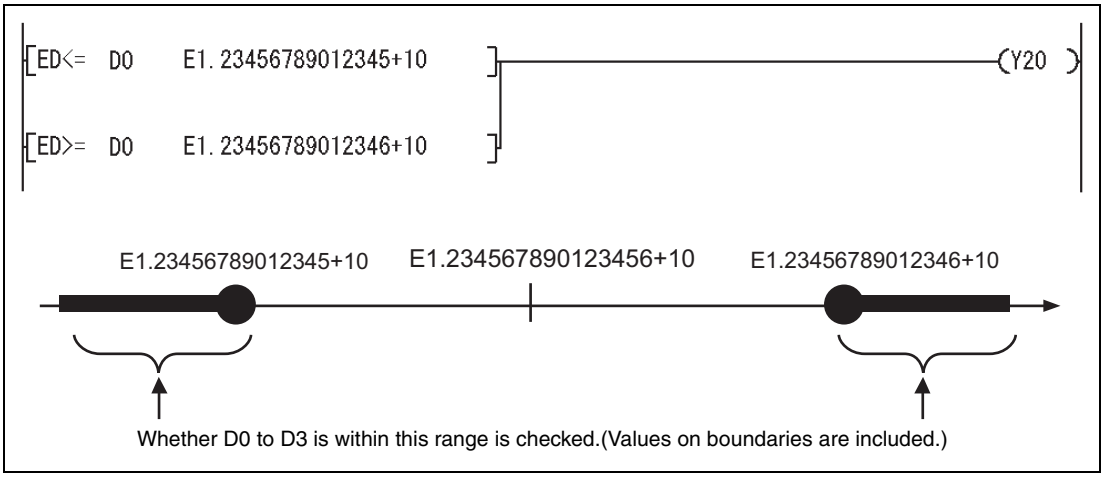
NOTE *Since the number of digits of the real number that can be input by a programing tool is up to 15 digits, the comparison with the real number whose number of significant digits is 16 or more cannot be made by the instruction shown in this section.*

When judging match/mismatch with the real number whose significant digits is 16 or more by the instruction in this section, compare it with the approximate values of the real number to be compared and judge by the sizes.

EXAMPLE 1 *When judging the match of E1.234567890123456+10 (number of significant digits is 16) and the double-precision floating-point data:*



EXAMPLE 2 *When judging the mismatch of E1.234567890123456+10 (Number of significant digits is 16) and the double-precision floating-point data:*



6.1.5 $ =, $ < >, $ >, $ < =, $ <, $ > =

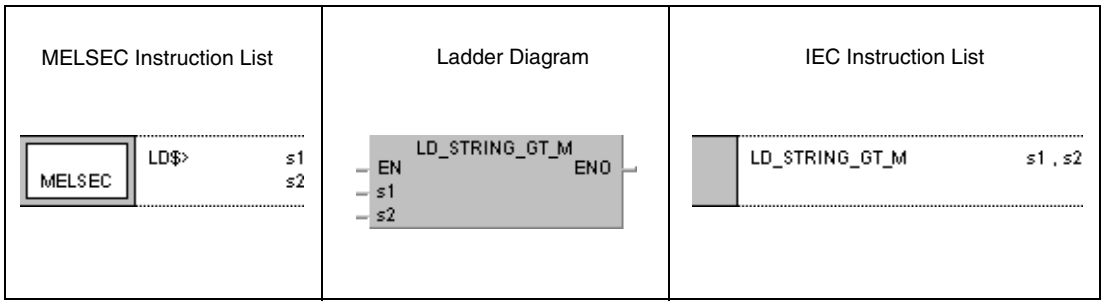
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

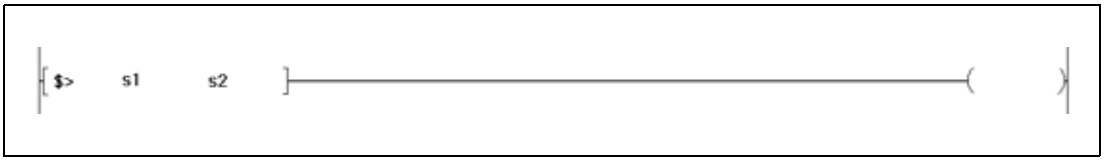
Devices

	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	●	—

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
s1	Comparative data, or first number of the device storing comparative data	Character string
s2		

Functions **Character string data comparison**

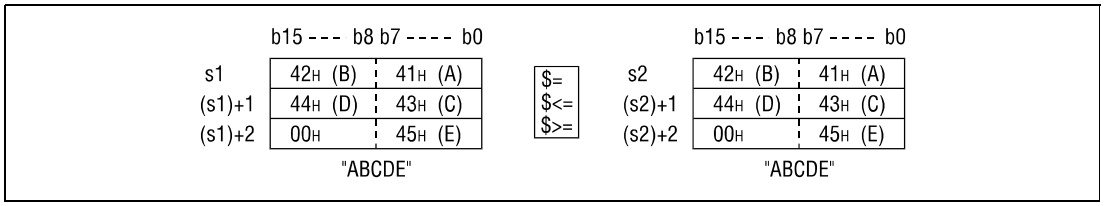
$=, $<>, $>, $<=, $<, $>= Comparison operation instructions

A comparison operation instruction for character string data consists of the instruction itself and two designated devices s1 and s2 to be compared.

The result of the comparison operation is binary ("1" if the comparison is true, "0" if the comparison is false). The binary result can be processed as a logical connection result. The comparison is performed with character string data in ASCII code character by character, beginning with the first character in the string.

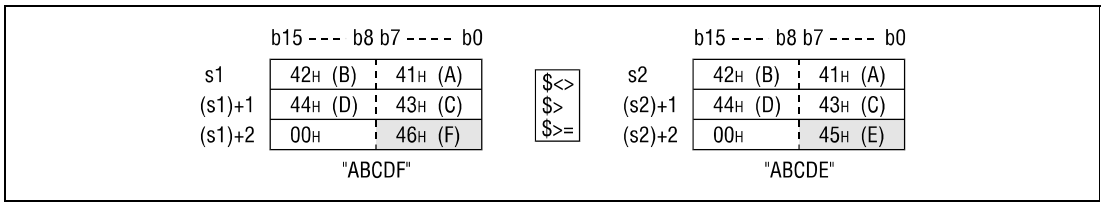
The s1 and s2 character strings include all characters from the designated device number up to the next device storing the code "00H".

If all character strings match, the comparison result for the operations $=, $<=, $>= is 1.



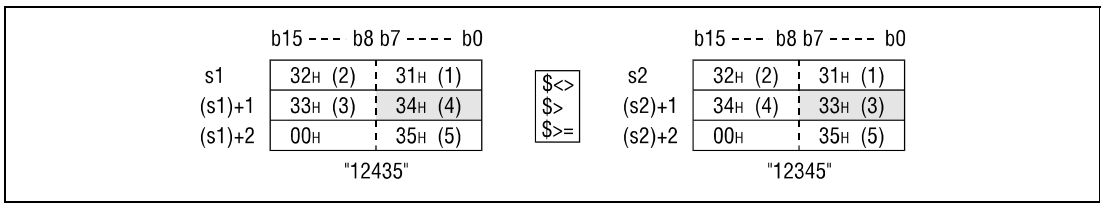
If the character strings are different, the character string with the larger character code will be the larger one.

Below, the comparison result for the operations $<>, $>, $>= is 1.



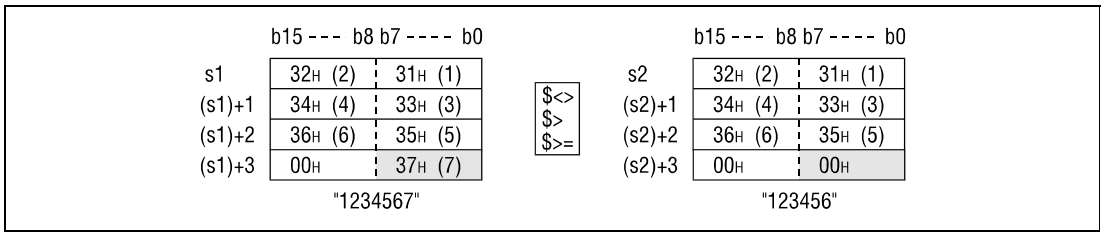
If the character strings are different, the first different sized character code determines whether the character string is larger or smaller.

Below, the comparison result for the operations $<>, $>, $>= is 1.



If the character strings are of different lengths, the data with the longer character string will be larger.

Below, the comparison result for the operations $<>, $>, $>=, is 1.



Operation Errors

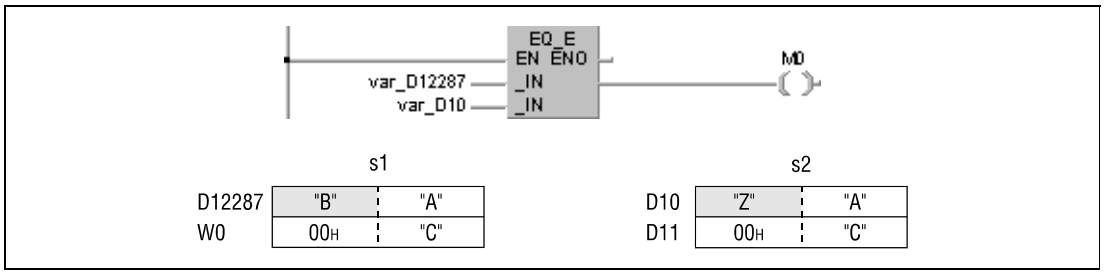
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The code "00H" does not exist within the relevant device range of s1 and s2. (Error code 4101)
- The character string of s1 and s2 exceeds 16383 characters. (Error code 4101)

NOTE

The character string data comparison instruction also checks the device range.

Even though, in cases where one character string exceeds the device range, character string data is being compared and non-matching characters within the device range are detected. The comparison operation results are output without returning an error code.



In the example shown above, the s1 character string exceeds the device range, and the most significant 16 bits (D12288) were renamed W0. Nevertheless, the comparison result is 0, because the second character in s1 is detected as different from that in s2. In this case no error code regarding the device range is returned.

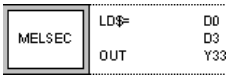
This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

Program Example 1

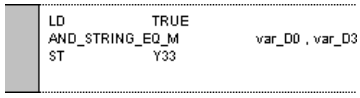
Comparison operation instruction $=

The following program compares character string data in D0 to character string data in D3. It turns ON Y33, if the data are equal.

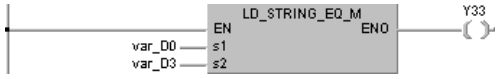
MELSEC Instruction List



IEC Instruction List



Ladder Diagram

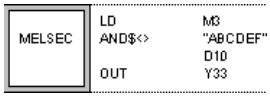


Program Example 2

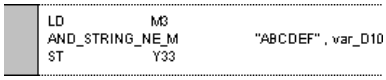
Comparison operation instruction $<>

The following program compares the character string "ABCDEF" to character string data in D10. It turns ON Y33, if the data are not equal.

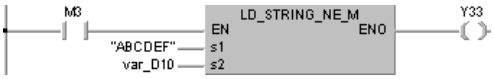
MELSEC Instruction List



IEC Instruction List



Ladder Diagram



Program Example 3

Comparison operation instruction $>

The following program compares character string data in D10 to character string data in D100. It turns ON Y33, if character string data in D10 is greater.

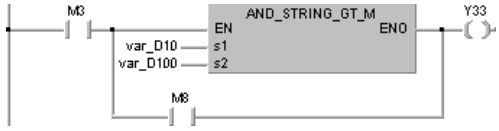
MELSEC Instruction List

```
MELSEC
LD      M3
LD$>    D10
        D100
OR      M8
ANB
OUT     Y33
```

IEC Instruction List

```
LD              M3
AND(
                M8
OR_STRING_GT_M      var_D10 , var_D100
)
ST              Y33
```

Ladder Diagram



Program Example 4

Comparison operation instruction $<=

The following program compares character string data in D0 to the character string "12345". Y33 is set, if character string data in D0 is less than or equal to "12345".

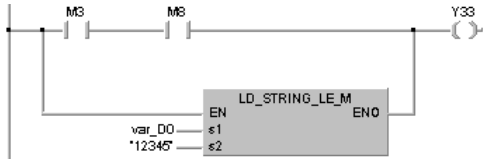
MELSEC Instruction List

```
MELSEC
LD      M3
AND     M8
OR$<=   D0
        "12345"
OUT     Y33
```

IEC Instruction List

```
LD              M3
AND             M8
OR_STRING_LE_M      var_D0 , "12345"
ST              Y33
```

Ladder diagram



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.1.6 BKCMP, BKCMPP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

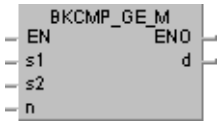
Devices

	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	—	—
d	●	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

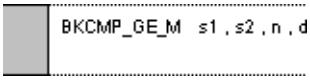
GX IEC Developer

MELSEC Instruction List

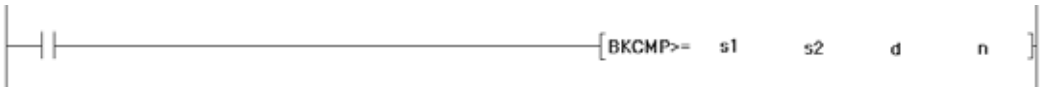
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s1	Comparative data, or first number of the device storing comparative data	BIN 16-bit
s2	First number of devices storing comparative data	BIN 16-bit
d	First number of device storing results of comparison operation	Bit
n	Number of data blocks compared	BIN 16-bit

Functions

BIN block data comparisons

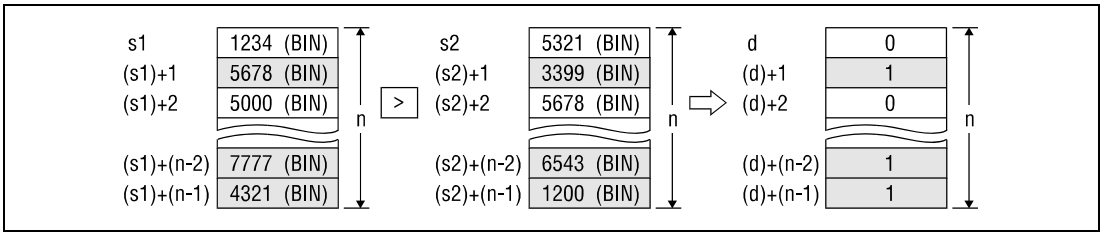
BKCMP Comparison operation instructions

A comparison operation instruction for BIN block data consists of the instruction itself, two designated devices s1 and s2 to be compared, a device d to store the result, and the number of datablocks to be compared.

It compares the nth BIN 16-bit block in s1 to the nth BIN 16-bit block in s2, beginning with the first number of device. The result of each block comparison is stored in d.

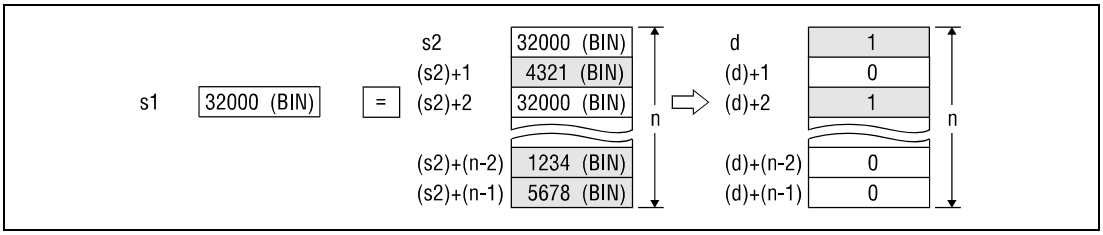
If the block comparison result is 1, then 1 is stored in d.

If the block comparison result is 0, then 0 is stored in d.



The comparison operation is conducted in 16-bit units.

The constant designated by s1 must be BIN 16-bit data ranging from -32768 to 32767.



The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol	Comparison Operation Results for nth 16-bit Block	
	1	0
BKCMP=	s1 = s2	s1 ≠ s2
BKCMP<>	s1 ≠ s2	s1 = s2
BKCMP>	s1 > s2	s1 ≤ s2
BKCMP<=	s1 ≤ s2	s1 > s2
BKCMP<	s1 < s2	s1 ≥ s2
BKCMP>=	s1 ≥ s2	s1 < s2

If all comparison results stored in d are 1, the block comparison signal SM704 is set.

If the device designated by d is already set (1), that device will not change. If the conditions designated by s1 and s2 are changed and the BKCMP_P instruction is executed, the device designated by d should be reset (0) before.

Operation Errors

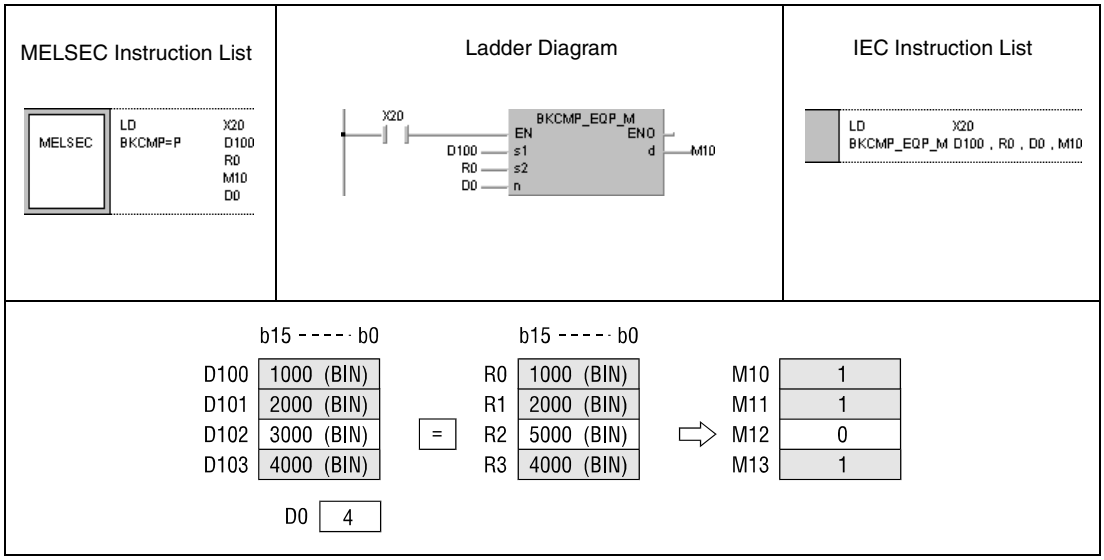
In following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The BIN block data at s1, s2, or d exceeds the relevant device range. (Error code 4101)
- The device range from [s1 to (s1) + (n-1)] overlaps with the device range [d to (d) + (n-1)]. (Error code 4101)
- The device range from [s2 to (s2) + (n-1)] overlaps with the device range [d to (d) + (n-1)]. (Error code 4101)
- The device range from [s1 to (s1) + (n-1)] overlaps with the device range [s2 to (s2) + (n-1)]. (Error code 4101)

Program Example 1

Comparison operation instruction BKCMP=P

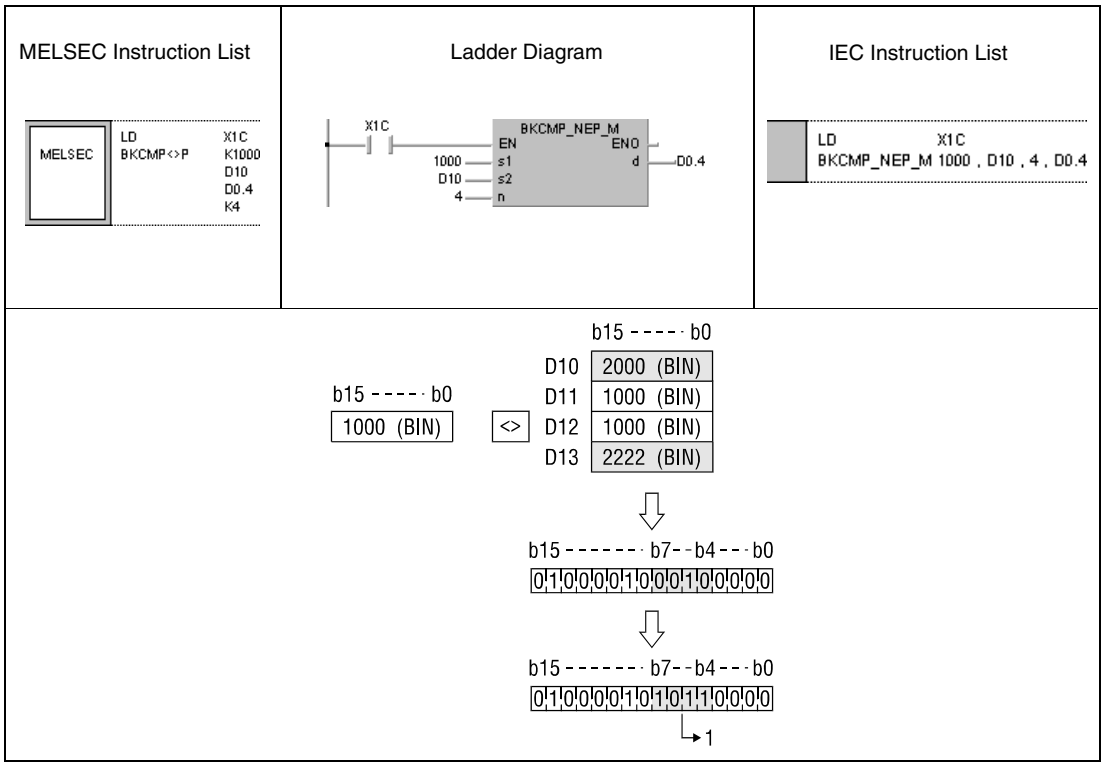
With leading edge from X20, the following program compares BIN block data in D100 to BIN block data in R0. The results of the comparison are stored from M10 onward. The number of blocks (4) to be compared is stored in D0



Program Example 2

Comparison operation instruction BKCMP<>P

With leading edge from X1C, the following program compares the constant K1000 to the block data beginning from D10. The number of blocks (4) to be compared is determined by the constant K4. The results of the comparison are stored in b4 through b7 of D0.



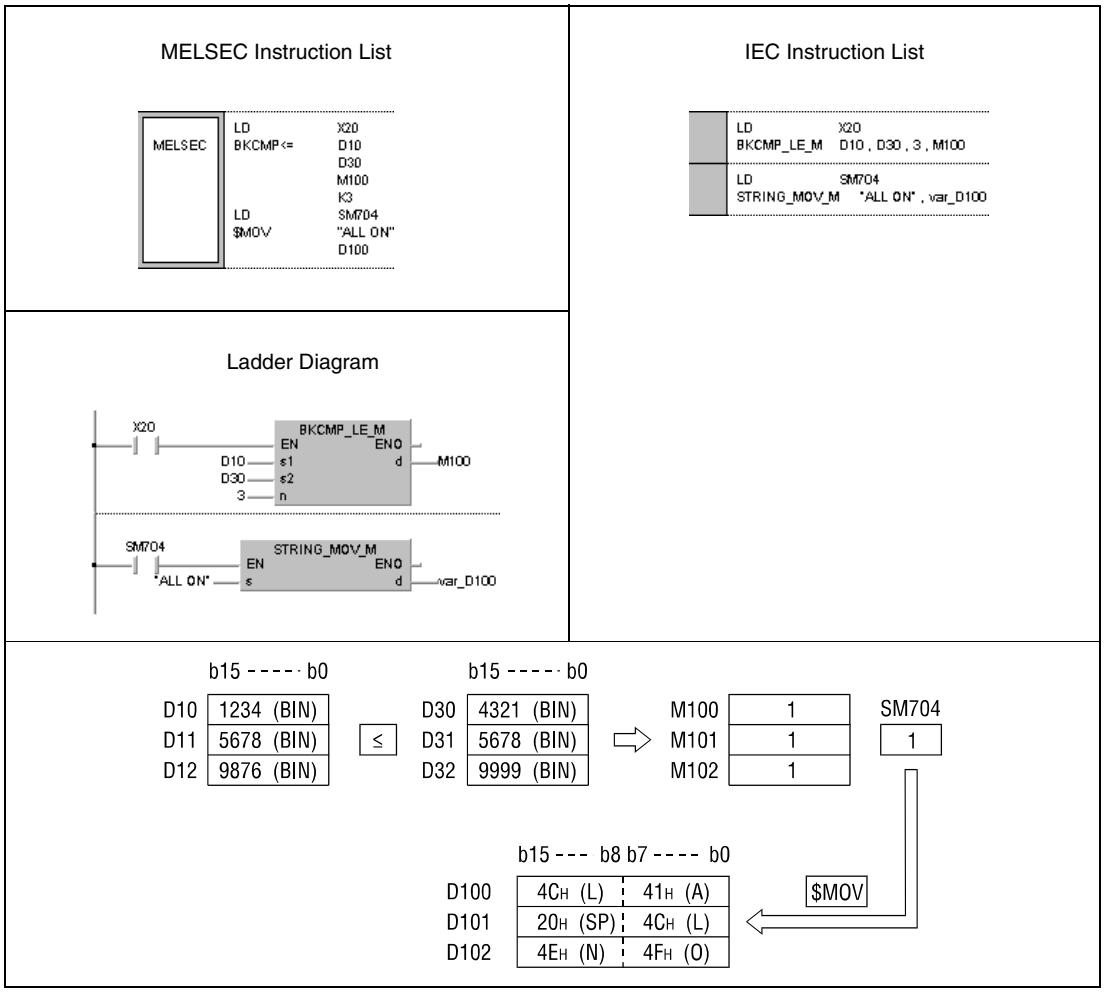
[1] Bits already in this state do not change (see function).

Program Example 3

Comparison operation instruction BKCMP<=

The following program compares, when X20 is turned ON, block data beginning from D10 to block data beginning from D30. The number of blocks (3) to be compared is determined by the constant K3. The results of the comparison are stored from M100 onward.

When all comparison results stored from M100 onward are 1, the block comparison signal SM704 is set and the character string "ALL ON" is transferred to D100.



NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.1.7 DBKCMP, DBKCMPP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

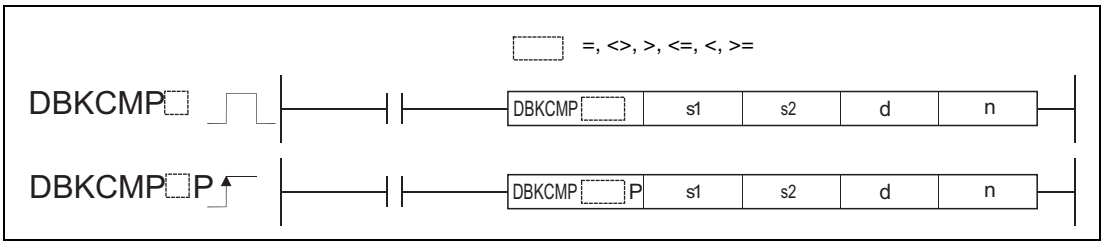
Devices

	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	—	—
d	●	—	●	—	—	—	—	—	—
n	—	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s1	Comparative data, or first number of the device storing comparative data	BIN 32-bit
s2	First number of devices storing comparative data	BIN 32-bit
d	First number of device storing results of comparison operation	Bit
n	Number of data blocks compared	BIN 16-bit

Functions **BIN 32-bit block data comparisons**

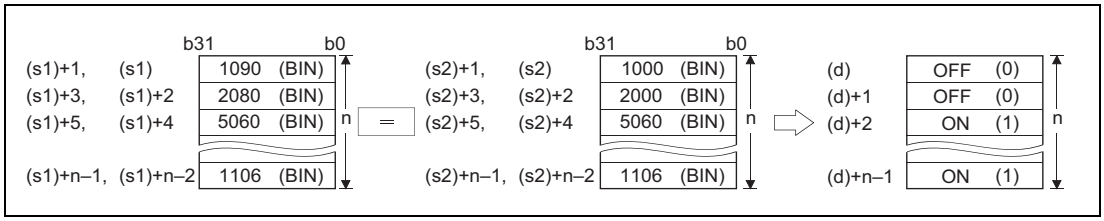
DBKCMP **Comparison operation instructions**

A comparison operation instruction for BIN 32-bit block data consists of the instruction itself, two designated devices s1 and s2 to be compared, a device d to store the result, and the number of datablocks to be compared.

It compares the nth BIN 32-bit block in s1 to the nth BIN 32-bit block in s2, beginning with the first number of device. The result of each block comparison is stored in d.

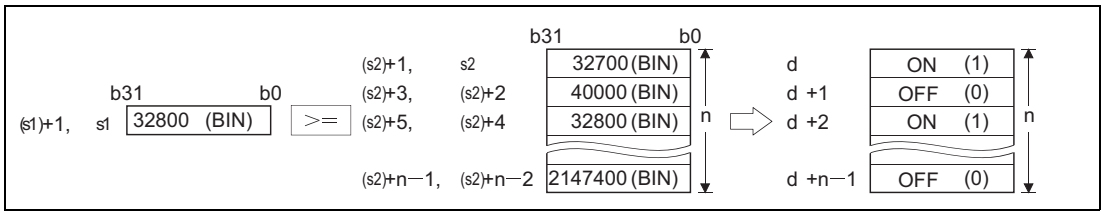
If the block comparison result is 1, then 1 is stored in d.

If the block comparison result is 0, then 0 is stored in d.



The comparison operation is conducted in 32-bit units.

The constant designated by s1 must be BIN 32-bit data ranging from -2147483648 to 2147483647.



The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol	Comparison operation results for nth 32-bit Block	
	1	0
DBKCMP=	s1 = s2	s1 ≠ s2
DBKCMP<>	s1 ≠ s2	s1 = s2
DBKCMP>	s1 > s2	s1 ≤ s2
DBKCMP<=	s1 ≤ s2	s1 > s2
DBKCMP<	s1 < s2	s1 ≥ s2
DBKCMP>=	s1 ≥ s2	s1 < s2

If all comparison results stored into the devices starting from the device specified by d to nth device are ON (1), or one of the results is OFF (0), the special relays will be ON or OFF in accordance with the conditions as follows.

Relay	All results of comparison operation are on (1)			All results of comparison operation are off (0)		
	Initial execution/scan	Interrupt (other than I45)/ Fixed scan execution	Interrupt (I45)	Initial execution/scan	Interrupt (other than I45)/ Fixed scan execution	Interrupt (I45)
SM704	ON	ON	ON	OFF	OFF	OFF
SM716	ON	—	—	OFF	—	—
SM717	—	ON	—	—	OFF	—
SM718	—	—	ON	—	—	OFF

In a standby program, a special relay depending on the caller program turns on or off.

If the value specified by n is 0, the instruction will be not processed.

Operation Errors

In following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

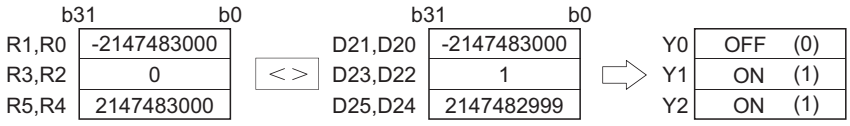
- A negative value is specified for n.
  (Error code 4100)
- The BIN block data at s1, s2, or d exceeds the relevant device range.
  (Error code 4101)
- The device range of the n-point devices starting from the device specified by s1 overlaps with the range of the n-point devices starting from the device specified by d.
  (Error code 4101)
- The device range of the n-point devices starting from the device specified by s2 overlaps with the range of the n-point devices starting from the device specified by d.
  (Error code 4101)

Program Example 1

Comparison operation instruction DBKCMP<>

The following program compares the value data stored at R0 to R5 with the value data stored at D20 to D25, and then stores the operation result into Y0 to Y2, when M0 is turned on.

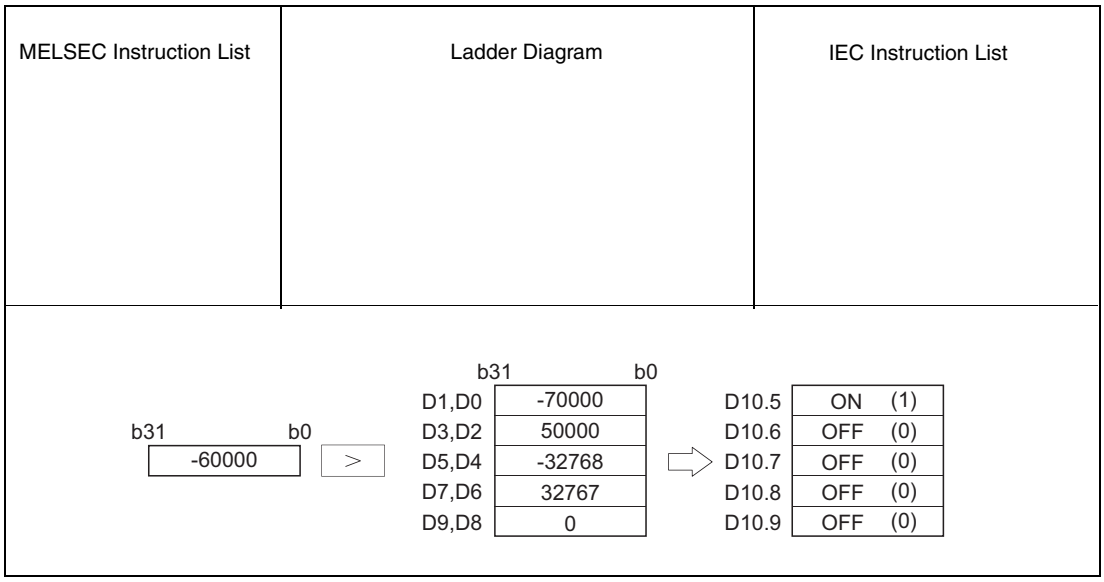
MELSEC Instruction List	Ladder Diagram	IEC Instruction List



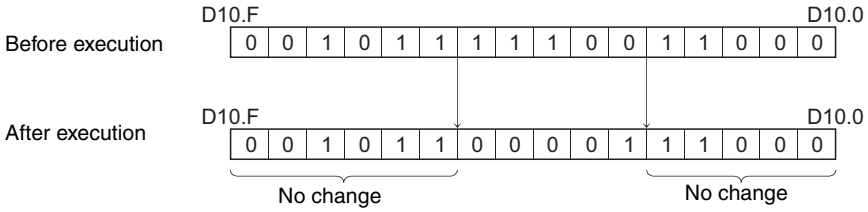
Program Example 2

Comparison operation instruction DBKCMP>=

The following program compares the constant with the value data stored at D0 to D9, and then stores the operation result into D10.5 to D10.9, when M0 is turned ON.



NOTE *When certain bits are specified in a word device, bits other than the certain bits that store the operation result do not change.*

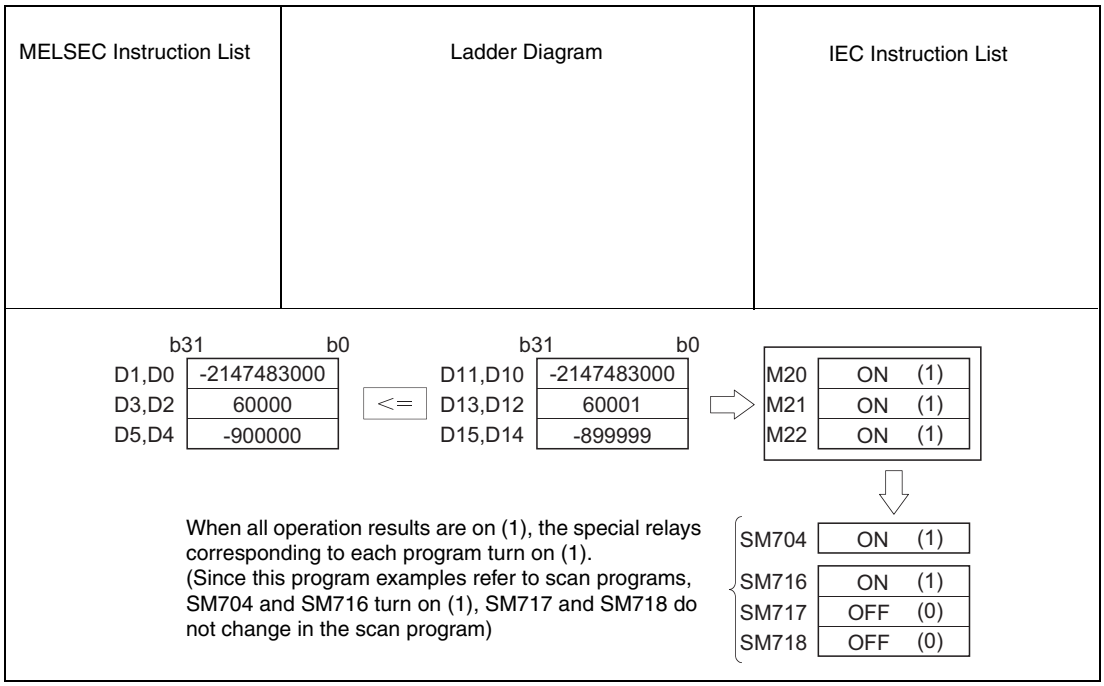


Program Example 3

Comparison operation instruction DBKCMP<=

The following program compares the value data stored at D0 to D5 with the value data stored at D10 to D15, and then stores the operation result into M20 to M22, when M0 is turned ON.

Also, the program transfers the character string "ALL ON" to D100 and up when all devices from M20 to M22 have reached the ON status.



NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.2 Arithmetic operation Instructions

Arithmetic operation instructions perform simple calculations like addition, subtraction, multiplication, and division.

Function	BIN		BCD	
	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
+ Addition	+	PLUS_M, PLUS_3_M	B+	BPLUS_M, BPLUS_3_M
	+P	PLUSP_M, PLUSP_3_M	B+P	BPLUSP_M, BPLUSP_3_M
	D+	DPLUS_M, DPLUS_3_M	DB+	DBPLUS_M, DBPLUS_3_M
	D+P	DPLUSP_M, DPLUSP_3_M	DB+P	DBPLUSP_M, DBPLUSP_3_M
− Subtraction	-	MINUS_M, MINUS_3_M	B-	BMINUS_M, BMINUS_3_M
	-P	MINUSP_M, MINUSP_3_M	B-P	BMINUSP_M, BMINUSP_3_M
	D-	DMINUS_M, DMINUS_3_M	DB-	DBMINUS_M, DBMINUS_3_M
	D-P	DMINUSP_M, DMINUSP_3_M	DB-P	DBMINUSP_M, DBMINUSP_3_M
× Multiplication	×	MULTI_3_M	B×	BMULTI_M
	×P	MULTIP_3_M	B×P	BMULTIP_M
	D×	DMULTI_3_M	DB×	DBMULTI_M
	D×P	DMULTIP_3_M	DB×P	DBMULTIP_M
/ Division	/	DIVID_3_M	B/	BDIVID_M
	/P	DIVIDP_3_M	B/P	BDIVIDP_M
	D/	DDIVID_3_M	DB/	DBDIVID_M
	D/P	DDIVIDP_3_M	DB/P	DBDIVIDP_M
+1 Increment	INC	INC_M		
	INCP	INCP_M		
	DINC	DINC_M		
	DINCP	DINCP_M		
−1 Decrement	DEC	DEC_M		
	DECP	DECP_M		
	DDEC	DDEC_M		
	DDECP	DDECP_M		

NOTE *Within the IEC editors please use the IEC commands.*

Function	Floating Point Data		BIN Block Data	
	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
+ Addition	E+	EPLUS_M, EPLUS_3_M	BK+	BKPLUS_M
	E+P	EPLUSP_M, EPLUSP_3_M	BK+P	BKPLUSP_M
	ED+		DBK+	
	ED+P		DBK+P	
− Subtraction	E-	EMINUS_M, EMINUS_3_M	BK-	BKMINUS_M
	E-P	EMINUSP_M, EMINUSP_3_M	BK-P	BKMINUSP_M
	ED-		DBK-	
	ED-P		DBK-P	
× Multiplication	E×	EMUL_M		
	E×P	EMULP_M		
	EDx			
	EDxP			
/ Division	E/	EDIV_M		
	E/P	EDIVP_M		
	ED/			
	ED/P			

Function	Character String Data	
	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
+ Addition	$+	STRING_PLUS_M, STRING_PLUS_3_M
	$+P	STRING_PLUSP_M, STRING_PLUSP_3_M

NOTE *Within the IEC editors please use the IEC commands.*

BIN data arithmetic operation instructions

If the result of the addition exceeds a BIN value 32767 (2147483647 for a 32-bit instruction), a negative value is generated (overflow).

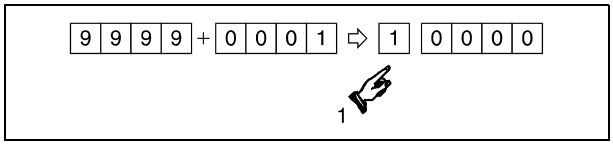
If the result of the subtraction falls below a BIN value -32768 (-2147483647 for a 32-bit instruction), a positive value is generated (underflow).

The calculation of positive and negative values appears as follows:

```
5  +  8  =  13
5  -  8  =  -3
5  ×  3  =  15
-5 ×  3  =  -15
-5 × (-3) =  15
5  /  3  =  1 remainder 2
-5 /  3  =  -1 remainder -2
5  / (-3) =  -1 remainder 2
-5 / (-3) =  1 remainder -2
```

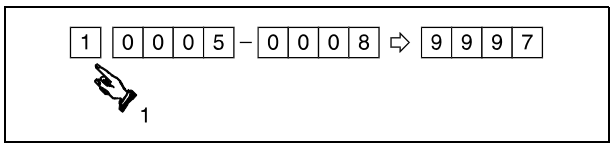
BCD data arithmetic operation instructions

If the result of the addition exceeds 9999 (99999999 for a 32-bit instruction), the higher bits are ignored (overflow). The carry flag in this case is not set.



[1] Carry ignored

If the result of the subtraction falls below 0000 (underflow), the carry is processed as shown:



[2] Carry

6.2.1 +, +P, -, -P

CPU

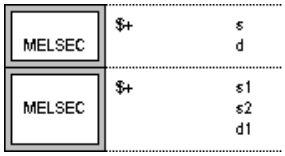
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

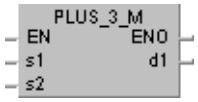
	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—

GX IEC Developer

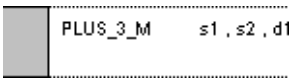
MELSEC Instruction List



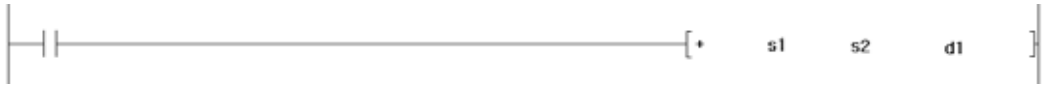
Ladder Diagram



IEC Instruction List



GX Works2



Variables

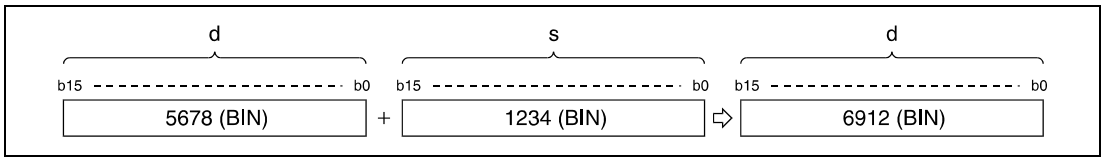
Set Data	Meaning	Data Type
s	Addition or subtraction data, or first number of device storing addition or subtraction data	BIN 16-bit
d	Data to be added to or subtracted from, or first number of device storing such data	
s1	Data to be added to or subtracted from, or first number of device storing such data	
s2	Addition or subtraction data, or first number of device storing addition or subtraction data	
d1	First number of device storing addition or subtraction data	

Functions **BIN 16-bit addition and subtraction operations**

+ BIN addition (16-bit)

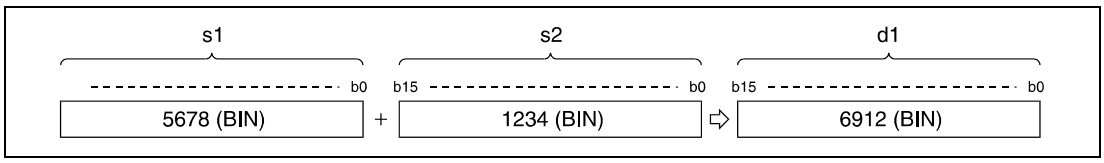
- Variation 1:

BIN 16-bit data in d is added to BIN 16-bit data in s. The result of the addition is stored in d.



- Varation 2:

BIN 16-bit data in s1 is added to BIN 16-bit data in s2. The result of the addition is stored in d1.



BIN 16-bit data designated by s, d, s1, s2, and d1 have to range within -32768 and 32767.

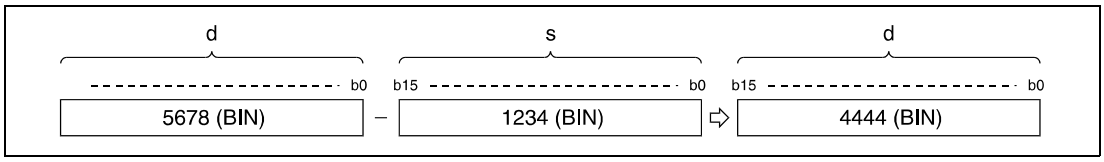
The most significant bit (b15) determines, whether data in s, d, s1, or d1 are positive (bit = 0) or negative (bit = 1).

If the least significant bit (b0) is fallen below or the most significant bit (b15) is exceeded, the carry flag is not set.

- BIN subtraction (16-bit)

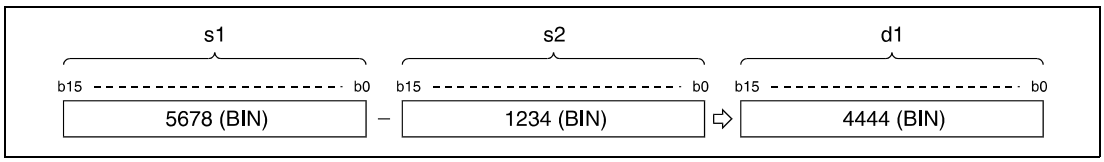
● Variation 1:

BIN 16-bit data in s is subtracted from BIN 16-bit data in d. The result of the subtraction is stored in d.



● Variation 2:

BIN 16-bit data in s2 is subtracted from BIN 16-bit data in s1. The result is stored in d1.



BIN 16-bit data designated by s, d, s1, s2, and d1 have to range within -32768 and 32767.

The most significant bit (b15) determines, whether data in s, d, s1, or d1 are positive (bit = 0) or negative (bit = 1).

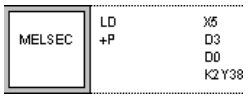
If the least significant bit (b0) is fallen below or the most significant bit (b15) is exceeded, the carry flag is not set.

Program Example 1

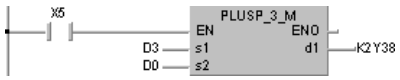
+P

WIth leading edge from X5, the following program adds data in D3 to data in D0. The result is stored from Y38 to Y3F.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

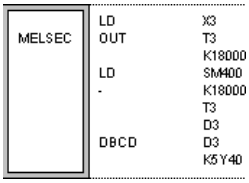
```
LD              X5
PLUSP_3_M       D3 , D0 , K2Y38
```

Program Example 2

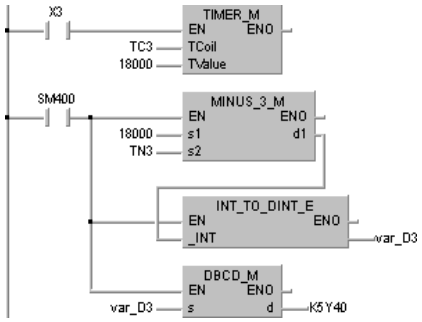
-

The following program outputs the difference between the nominal and the actual value of timer T3 to Y40 through Y53 in BCD.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD              X3
TIMER_M         TC3 , K18000
LD              SM400
MINUS_3_M       K18000 , TN3 , D3
INT_TO_DINT_E   D3 , var_D3
DBCD_M          var_D3 , K5Y40
```

NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.2.2 D+, D+P, D-, D-P

CPU

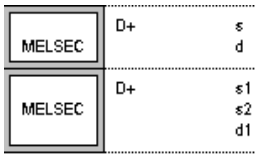
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

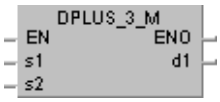
	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—

GX IEC Developer

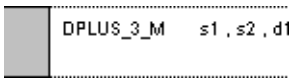
MELSEC Instruction List



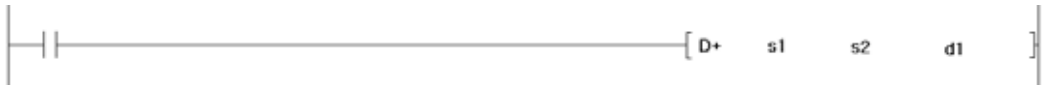
Ladder Diagram



IEC Instruction List



GX Works2



Variables

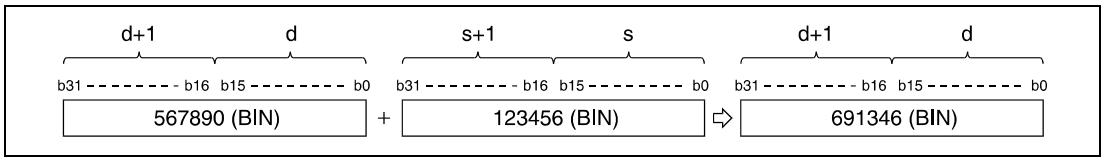
Set Data	Meaning	Data Type
s	Addition or subtraction data, or first number of device storing addition or subtraction data	BIN 32-bit
d	Data to be added to or subtracted from, or first number of device storing such data	
s1	Data to be added to or subtracted from, or first number of device storing such data	
s2	Addition or subtraction data, or first number of device storing addition or subtraction data	
d1	First number of device storing addition or subtraction data	

Functions **BIN 32-bit addition and subtraction operations**

D+ BIN addition (32-bit)

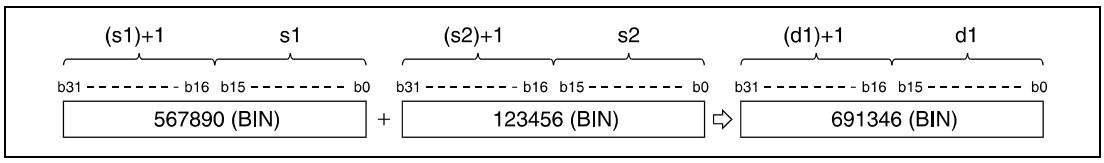
- Variation 1:

BIN 32-bit data in d is added to BIN 32-bit data in s. The result of the addition is stored in d.



- Variation 2:

BIN 32-bit data in s1 is added to BIN 32-bit data in s2. The result of the addition is stored in d1.



BIN 32-bit data designated by s, d, s1, s2, and d1 have to range within -2147483648 and 2147483647.

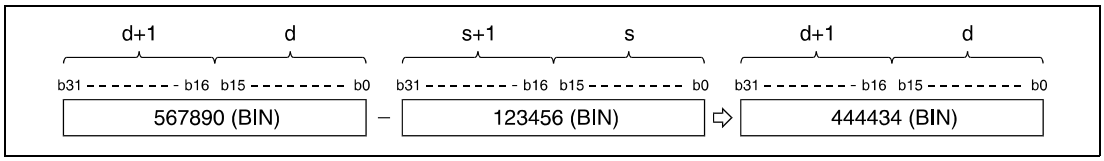
The most significant bit (b31) determines, whether data in s, d, s1, or d1 are positive (bit = 0) or negative (bit = 1).

If the least significant bit (b0) is fallen below or the most significant bit (b31) is exceeded, the carry flag is not set.

D- BIN subtraction (32-bit)

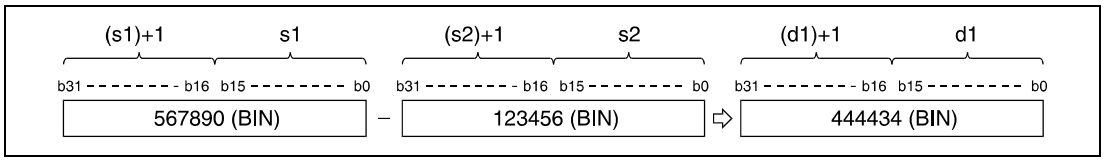
- Varation 1:

BIN 32-bit data in s is subtracted from BIN 32-bit data in d. The result of the subtraction is stored in d.



● Variation 2:

BIN 32-bit data in s2 is subtracted from BIN 32-bit data in s1. The result is stored in d1.



BIN 32-bit data designated by s, d, s1, s2, and d1 have to range within -2147483648 and 2147483647.

The most significant bit (b31) determines, whether data in s, d, s1, or d1 are positive (bit = 0) or negative (bit = 1).

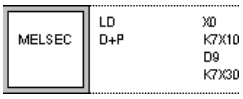
If the least significant bit (b0) is fallen below or the most significant bit (b31) is exceeded, the carry flag is not set.

Program Example 1

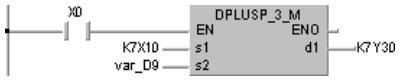
D+P

With leading edge from X0, the following program adds data in X10 through X2B to D9 and D10. The result is stored in Y30 through Y4B.

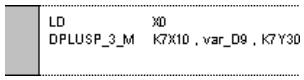
MELSEC Instruction List



Ladder Diagram



IEC Instruction List

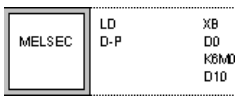


Program Example 2

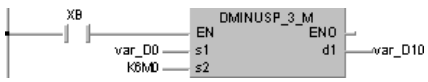
D-P

With leading edge from XB, the following program subtracts data in M0 through M23 from data in D0 and D1. The result is stored in D10 and D11.

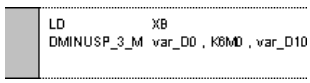
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.2.3 x, xP, /, /P

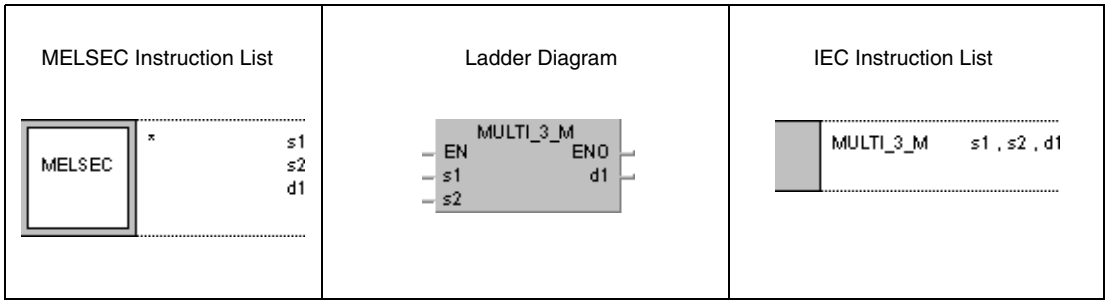
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

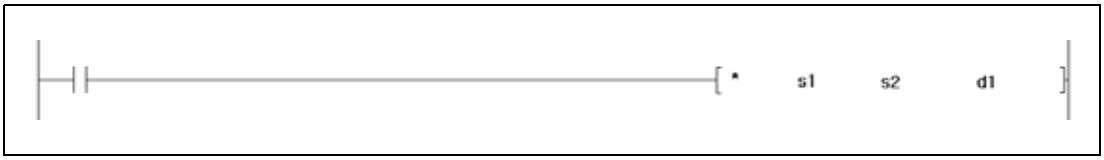
Devices

	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module Index Register U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—

GX IEC Developer



GX Works2



Variables

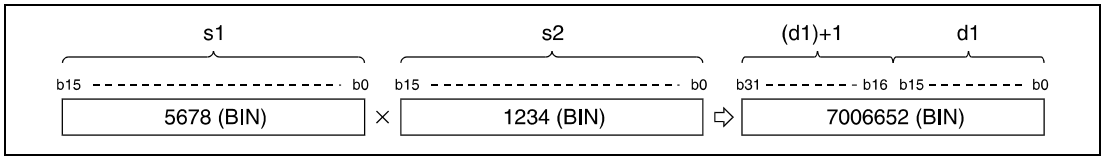
Set Data	Meaning	Data Type
s1	Data that will be multiplied or divided, or first number of device storing data that will be multiplied or divided	BIN 16-bit
s2	Data to multiply or divide by, or first number of device storing such data	BIN 16-bit
d1	First number of device storing the operation results of multiplication or division operation	BIN 32-bit

Functions

BIN 16-bit multiplication and division

x BIN multiplication (16-bit)

BIN 16-bit data in s1 is multiplied with BIN 16-bit data in s2. The result is stored in d1.



If the result in d1 is a bit device, designation is made from the lower bits.

Example:

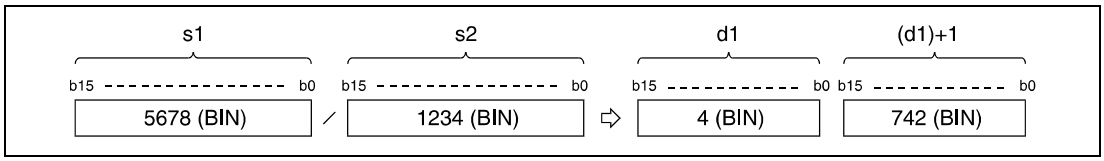
K1: lower 4 bits (b0 to b3)
K4: lower 16 bits (b0 to b15)
K8: 32 bits (b0 to b31)

BIN 16-bit data designated by s1 and s2 have to range within -32768 and 32767.

The most significant bit (b15 or b31) in d1 determines, whether data in s1, s2 or d1 are positive (bit = 0) or negative (bit = 1).

/ BIN division (16-bit)

BIN 16-bit data in s1 is divided by BIN 16-bit data in s2. The result is stored in d1.



If a word device is used, the result of the operation is stored as 32-bits, and both, the quotient and remainder are stored. The quotient is stored in the least significant 16-bits. The remainder is stored in the most significant 16-bits.

If a bit device is used, 16-bits are used and only the quotient is stored.

BIN 16-bit data designated by s1 and s2 have to range within -32768 and 32767.

The most signigicant bit (b15) in d1 determines, whether data in s1, s2, d1 or (d1)+1 is positive (bit = 0) or negative (bit = 1).

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

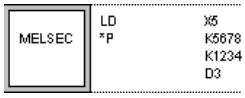
- Division by 0 (Error code 4100)

Program Example 1

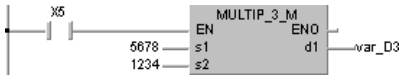
xP

With leading edge from X5, the following program multiplies 5678 and 1234. The result is stored in D3 and D4.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

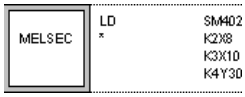
```
LD          X5
MULTIP_3_M  5678 , 1234 , var_D3
```

Program Example 2

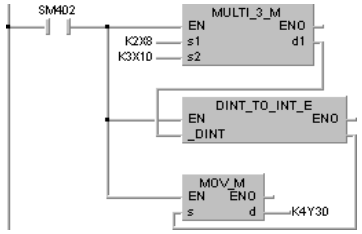
x

The following program multiplies BIN data at X8 through XF and BIN data at X10 through X1B. The result is output at Y30 through Y3F.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

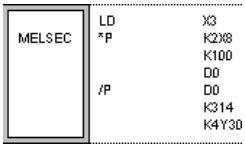
```
LD              SM402
MULTI_3_M       K2X8 , K3X10 , var_temp
DINT_TO_INT_E   var_temp , D0
MOV_M           D0 , K4Y30
```

Program Example 3

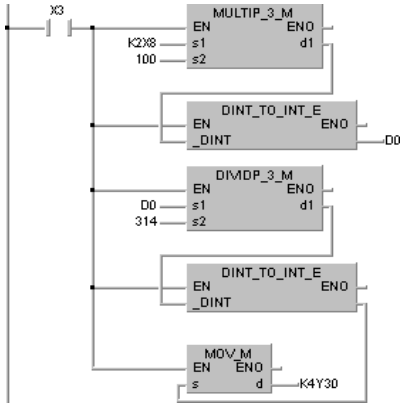
/P

With leading edge from X3, the following program divides data at X8 through XF by 3.14. The result is output at Y30 through Y3F.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD              X3
MULTIP_3_M      K2X8 , K100 , var_D0
DINT_TO_INT_E   var_D0 , D0
DIVDP_3_M       D0 , K314 , var_temp
DINT_TO_INT_E   var_temp , D0
MOV_M           D0 , K4Y30
```

NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.2.4 Dx, DxP, D/, D/P

CPU

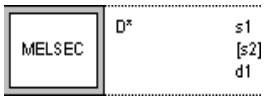
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function ModuleSpecial Function Module U□\G□	Index RegisterIndex Register Zn	ConstantConstant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	—	—	—	—	—	—

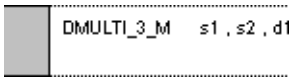
GX IEC Developer

MELSEC Instruction List



Ladder Diagram

IEC Instruction List



GX Works2

Variables

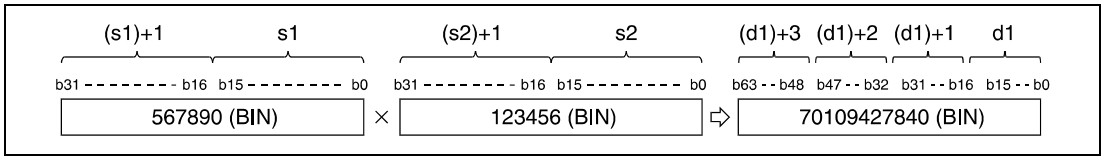
Set Data	Meaning	Data Type	
		MELSEC	IEC
s1	Data that will be multiplied or divided, or first number of device storing data that will be multiplied or divided	BIN 32-bit	ANY32
s2	Data to multiply or divide by, or first number of device storing such data	BIN 32-bit	ANY32
d1	First number of device storing the operation results of multiplication or division operation	BIN 64-bit	Array [1..2] of ANY32

Functions

BIN 32-bit multiplication and division

Dx BIN multiplication (32-bit)

BIN 32-bit data in s1 is multiplied with BIN 32-bit data in s2. The result is stored in d1.



If the result in d1 is a bit device, designation is made from the lower bits.

Example:

K1: lower 4 bits (b0 to b3)
K4: lower 16 bits (b0 to b15)
K8: 32 bits (b0 to b31)

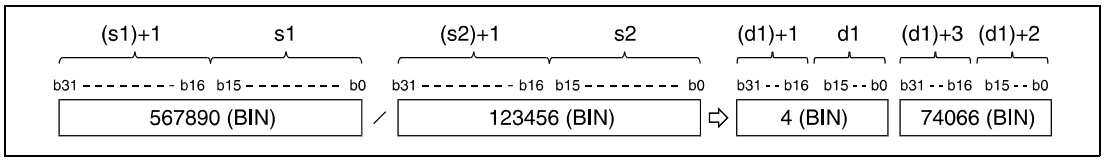
If the upper 32 bits of the bit device are required for the result of the multiplication operation, first temporarily store the data in a word device, then transfer the word device data to the bit device designated by (d1)+2 and (d1)+3.

BIN 32-bit data designated by s1 and s2 has to range within -2147483648 and 2147483647.

The most significant bit (b31 or b63) in d1 determines, whether data in s1, s2 or d1 is positive (bit = 0) or negative (bit = 1).

D/ BIN division (32-bit)

BIN 32-bit data in s1 is divided by BIN 32-bit data in s2. The result is stored in d1.



If a word device is used, the result of the division operation is stored as array of DINT (64-bit), and both the quotient and remainder are stored. The quotient is stored in the lower array elements (32-bit). The remainder is stored in the upper array elements (32-bit).

If a bit device is used, 32 bits are used and only the quotient is stored.

BIN 32-bit data designated by s1 and s2 has to range within -2147483648 and 2147483647.

The most significant bit (b31) in d1 determines, whether data in s1, s2, d1 or (d1)+2 is positive (bit = 0) or negative (bit = 1).

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- Division by 0 (Error code 4100)

Program Example 1

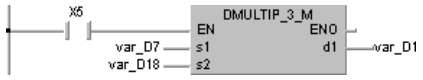
DxP

With leading edge from X5, the following program multiplies BIN data in D7 and D8 with BIN data in D18 and D19. The result is stored in D1 through D4.

MELSEC Instruction List

```
MELSEC   LD     X5
         D*P    D7
                D18
                D1
```

Ladder Diagram



IEC Instruction List

```
LD              X5
DMULTIP_3_M     var_D7 , var_D18 , var_D1
```

Program Example 2

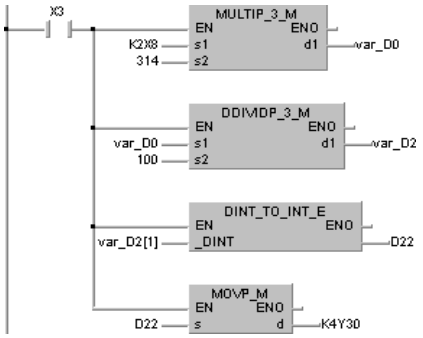
xP

With leading edge from X3, the following program multiplies data at X8 through XF and 3.14. The result is output at Y30 through Y3F.

MELSEC Instruction List

```
MELSEC   LD     X3
         *P     K2X8
                K314
                D0
         D/P    D0
                K100
                D2
         MOVP   D2
                K4Y30
```

Ladder Diagram



IEC Instruction List

```
LD              X3
MULTIP_3_M      K2X8 , K314 , var_D0
DDIVDP_3_M      var_D0 , K100 , var_D2
DINT_TO_INT_E   var_D2[1] , D2
MOVP_M          D2 , K4Y30
```

NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.2.5 B+, B+P, B-, B-P

CPU

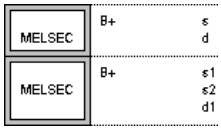
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

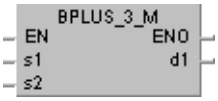
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—

GX IEC Developer

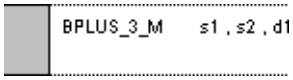
MELSEC Instruction List



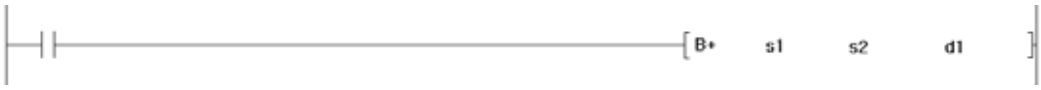
Ladder Diagram



IEC Instruction List



GX Works2



Variables

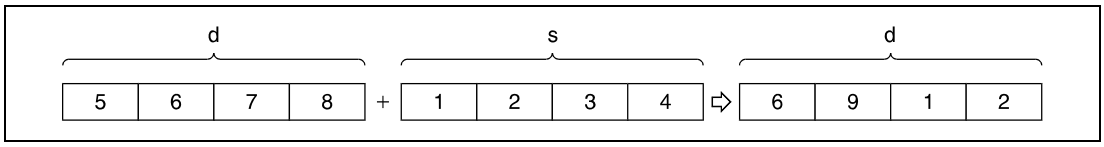
Set Data	Meaning	Data Type
s	Addition or subtraction data, or first number of device storing addition or subtraction data	BCD 4-digit
d	Data to be added to or subtracted from, or first number of device storing such data	
s1	Data to be added to or subtracted from, or first number of device storing such data	
s2	Addition or subtraction data, or first number of device storing addition or subtraction data	
d1	First number of device storing addition or subtraction data	

Functions **BCD 4-digit addition and subtraction operations**

B+ BCD addition (4-digit)

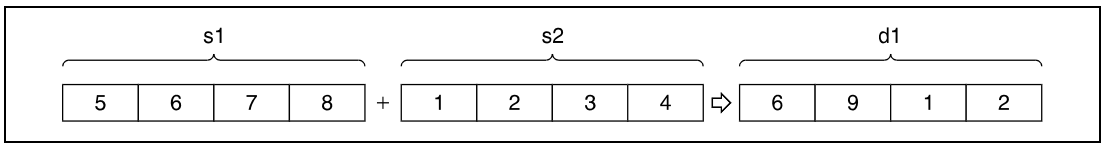
- Variation 1:

BCD 4-digit data in d is added to BCD 4-digit data in s. The result of the addition is stored in d.



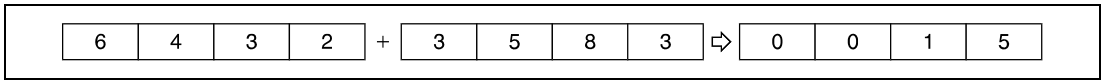
- Variation 2:

BCD 4-digit data in s1 is added to BCD 4-digit data in s2. The result is stored in d1.



BCD 4-digit data designated by s, d, s1, s2, and d1 have to range within 0 and 9999. Undesignated digits are read as 0 (e.g. 12 = 0012).

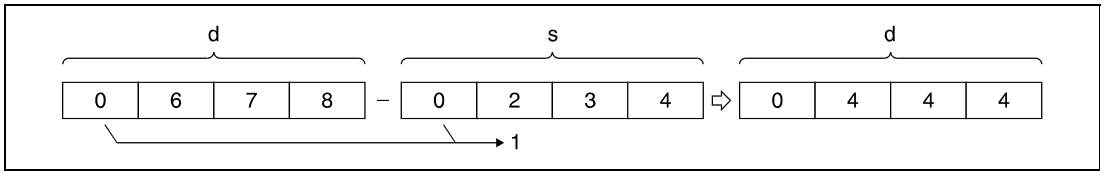
If the result of the addition exceeds 9999, the higher bits are ignored (overflow). The carry flag in this case is not set.



B- BCD subtraction (4-digit)

- Variation 1:

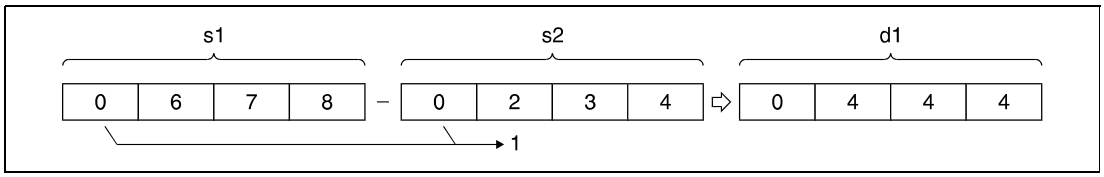
BCD 4-digit data in s is subtracted from BCD 4-digit data in d. The result is stored in d.



[1] Undesignated digits are read as 0.

● Variation 2:

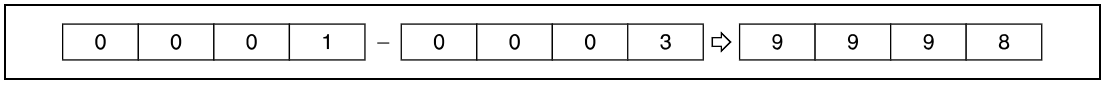
BCD 4-digit data in s2 is subtracted from BCD 4-digit data in s1. The result is stored in d1.



[1] Undesignated digits are read as 0.

BCD 4-digit data designated by s, d, s1, s2, and d1 have to range within 0 and 9999.

If the result of the subtraction operation is negative, the minuend is reduced by the number of steps determined by the subtrahend. The carry flag in this case is not set.



In the further course of a program, make sure that either positive or negative results are treated adequately.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

● The BCD 4-digit data designated by s, d, s1, s2, or d1 exceed the relevant device range of 0 to 9999. (Error code 4100)

Program Example 1

B+P (s, d)

The following program adds BCD data 5678 to BCD data 1234. The result is stored in D993 and output at Y30 through Y3F.

The first line of the program stores the value 5678 in D993.

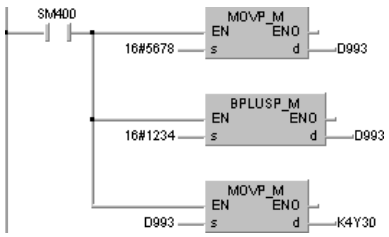
The following program step adds BCD data 1234 to BCD data in D993.

The MOV instruction in the last program step outputs the result in D993 at Y30 through Y3F.

MELSEC Instruction List

```
MELSEC
LD      SM400
MOVP    H5678
        D993
B+P     H1234
        D993
MOVP    D993
        K4Y30
```

Ladder Diagram



IEC Instruction List

```
LD      SM400
MOVP_M  16#5678 , D993
BPLUSP_M 16#1234 , D993
MOVP_M  D993 , K4Y30
```

Program Example 2

B-P (s, d)

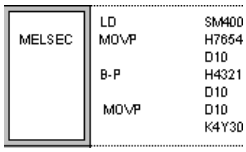
The following program subtracts BCD data 4321 from BCD data 7654. The result is stored in D10 and output at Y30 through Y3F.

The first line of the program stores the value 7654 in D10.

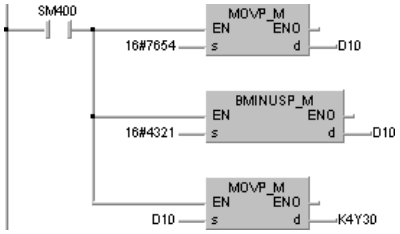
The following program step subtracts BCD data 4321 from BCD data in D10.

The MOV instruction in the last program step outputs the result in D10 at Y30 through Y3F.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

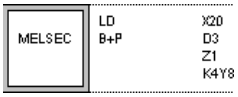
```
LD          SM400
MOVP_M      16#7654 , D10
BMINUSP_M   16#4321 , D10
MOVP_M      D10 , K4Y30
```

Program Example 3

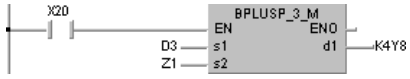
B+P (s1, s2, d1)

With leading edge from X20, the following program adds BCD data in D3 to BCD data in Z1. The result is output at Y8 through Y17.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

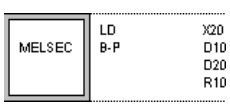
```
LD          X20
BPLUSP_3_M  D3 , Z1 , K4Y8
```

Program Example 4

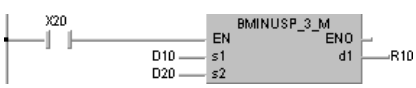
B-P (s1, s2, d1)

With leading edge from X20, the following program subtracts BCD data in D20 from BCD data in D10. The result is stored in R10.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD          X20
BMINUSP_3_M D10 , D20 , R10
```

6.2.6 DB+, DB+P, DB-, DB-P

CPU

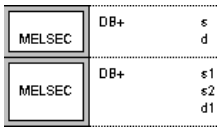
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

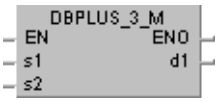
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—

GX IEC Developer

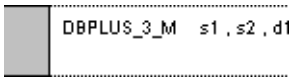
MELSEC Instruction List



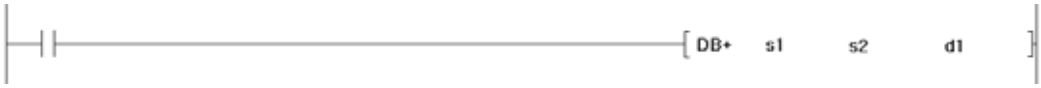
Ladder Diagram



IEC Instruction List



GX Works2



Variables

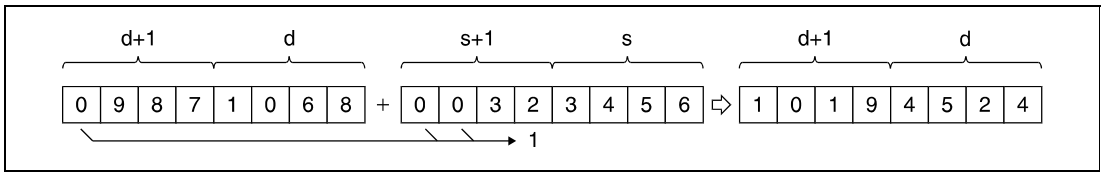
Set Data	Meaning	Data Type
s	Addition or subtraction data, or first number of device storing addition or subtraction data	BCD 8-digit
d	Data to be added to or subtracted from, or first number of device storing such data	
s1	Data to be added to or subtracted from, or first number of device storing such data	
s2	Addition or subtraction data, or first number of device storing addition or subtraction data	
d1	First number of device storing addition or subtraction data	

Functions **BCD 8-digit addition and subtraction operations**

DB+ BCD addition (8-digit)

● Variation 1:

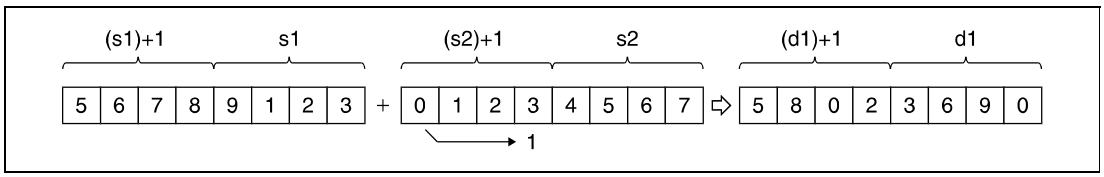
BCD 8-digit data in d is added to BCD 8-digit data in s. The result is stored in d.



[1] Undesignated digits are read as 0.

● Variation 2:

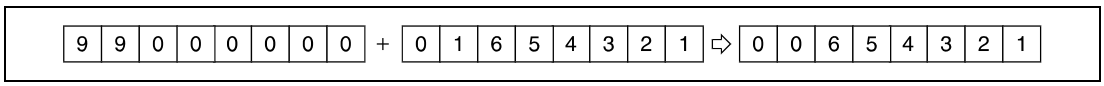
BCD 8-digit data in s1 is added to BCD 8-digit data in s2. The result is stored in d1.



[1] Undesignated digits are read as 0.

BCD 8-digit data designated by s, d, s1, and d1 have to range within 0 and 99999999. Undesignated digits are read as 0 (e.g. 12345 = 00012345).

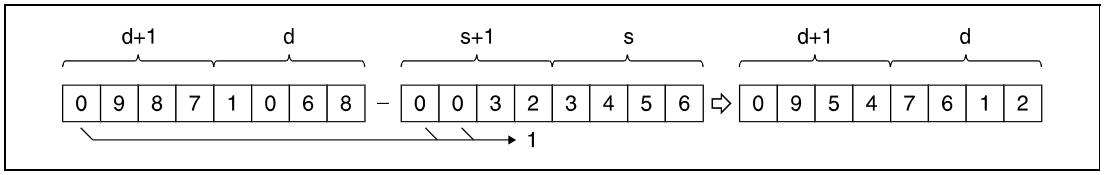
If the result of the addition exceeds 99999999, the higher bits are ignored (overflow). The carry flag in this case is not set.



DB- BCD subtraction (8-digit)

● Variation 1:

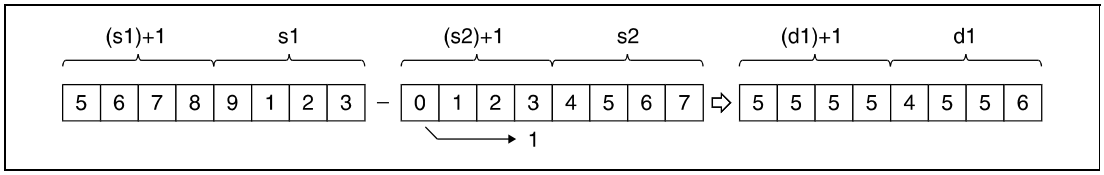
BCD 8-digit data in s is subtracted from BCD 8-digit data in d. The result is stored in d.



[1] Undesignated digits are read as 0

● Variation 2:

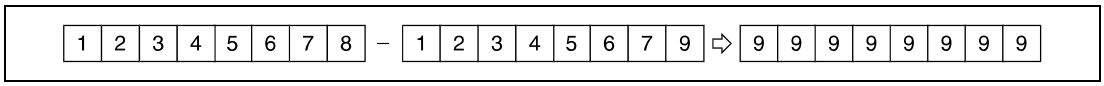
BCD 8-digit data in s2 is subtracted from BCD 8-digit data in s1. The result is stored in d1.



[1] Undesignated digits are read as 0

BCD 8-digit data designated by s, d, s1, and d1 have to range within 0 and 99999999. Undesignated digits are read as 0 (e.g. 12345 = 00012345).

If the result of the subtraction operation is negative, the minuend is reduced by the number of steps determined by the subtrahend. The carry flag in this case is not set.



In the further course of a program, make sure that either positive or negative results are treated adequately.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

● The BCD 8-digit data designated by s, d, s1, s2, or d1 exceed the relevant device range of 0 to 99999999. (Error code 4100)

Program Example 1

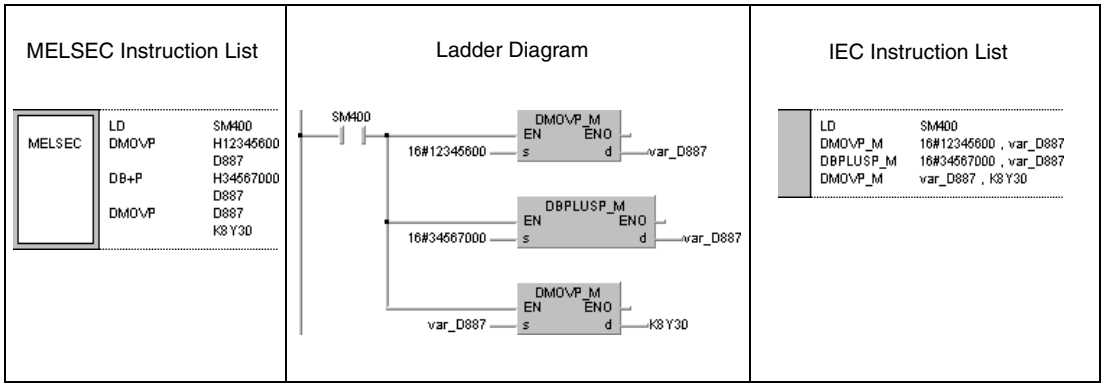
DB+P (s, d)

The following program adds BCD data 12345600 to BCD data 34567000. The result is stored in D887 and D888 and output at Y30 through Y4F.

The first line of the program stores the value 12345600 in D887 and D888.

The following program step adds BCD data 34567000 to BCD data in D887 and D888.

The DMOVP instruction in the last program step outputs the result in D887 and D888 at Y30 through Y4F.



Program Example 2

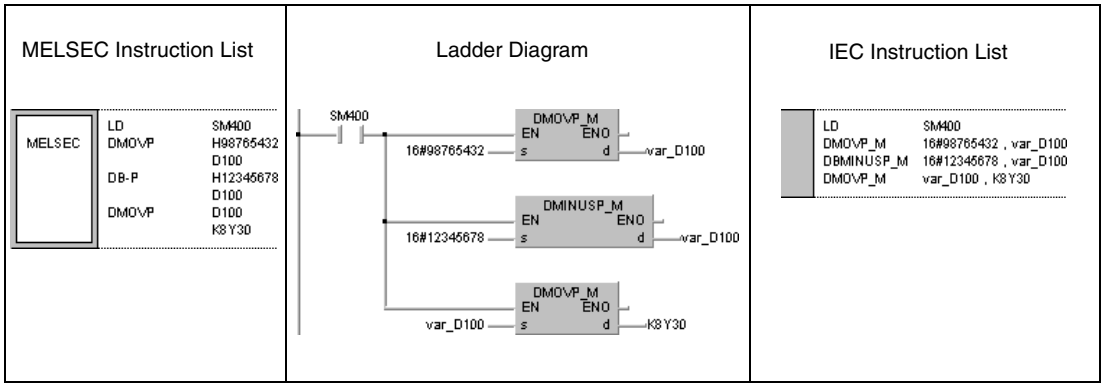
DB-P (s, d)

The following program subtracts BCD data 12345678 from BCD data 98765432. The result is stored in D100 and D101 and output at Y30 through Y4F.

The first line of the program stores the value 98765432 in D100 and D101.

The following program step subtracts BCD data 12345678 from BCD data in D100 and D101.

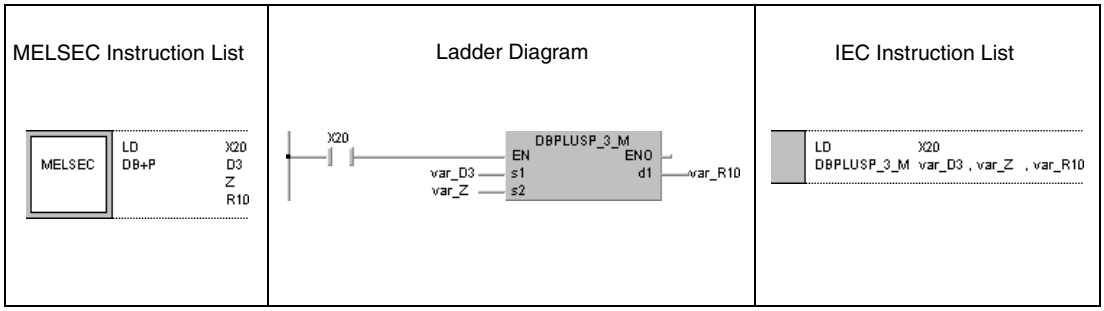
The DMOVP instruction in the last program step outputs the result in D100 and D101 at Y30 through Y4F.



Program Example 3

DB+P (s1, s2, d1)

With leading edge from X20, the following program adds BCD data in D3 and D4 to BCD data in Z and V. The result is stored in R10 and R11.



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.2.7 Bx, BxP, B/, B/P

CPU

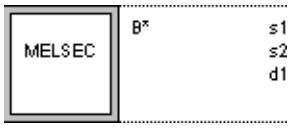
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□N□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—

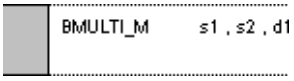
GX IEC Developer

MELSEC Instruction List

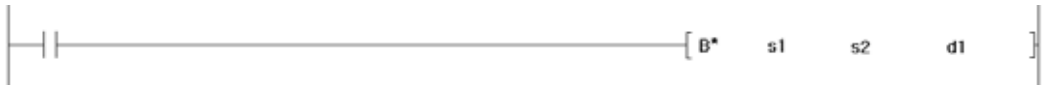


Ladder Diagram

IEC Instruction List



GX Works2



Variables

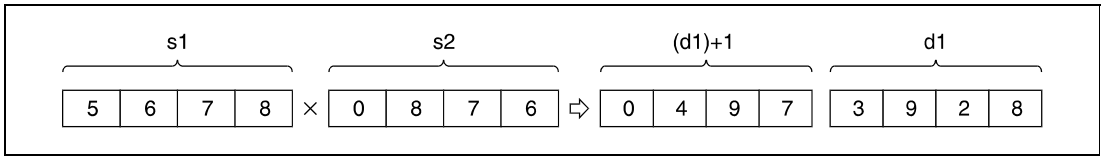
Set Data	Meaning	Data Type	
		MELSEC	IEC
s1	Data that will be multiplied or divided, or first number of device storing data that will be multiplied or divided	BCD 4-digit	WORD
s2	Data to multiply or divide by, or first number of device storing such data	BCD 4-digit	WORD
d1	First number of device storing the operation results of multiplication or division operation	BCD 8-digit	2 Arrays of WORD

Functions

BCD 4-digit multiplication and division operations

Bx BCD multiplication (4-digit)

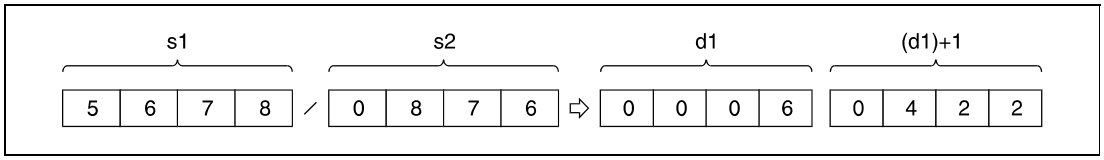
BCD 4-digit data in s1 is multiplied with BCD 4-digit data in s2. The result is stored in d1.



BCD 4-digit data designated by s1 and s2 have to range within 0 and 9999.

B/ BCD division (4-digit)

BCD 4-digit data in s1 is divided by BCD 4-digit data in s2. The result is stored in d1.



The result of the division is stored in two 16-bit WORD arrays. The lower array stores the quotient (BCD 4-digit) and the upper array stores the remainder (BCD 4-digit).

If d is a bit device, the remainder of the division is not stored.

Operation Errors

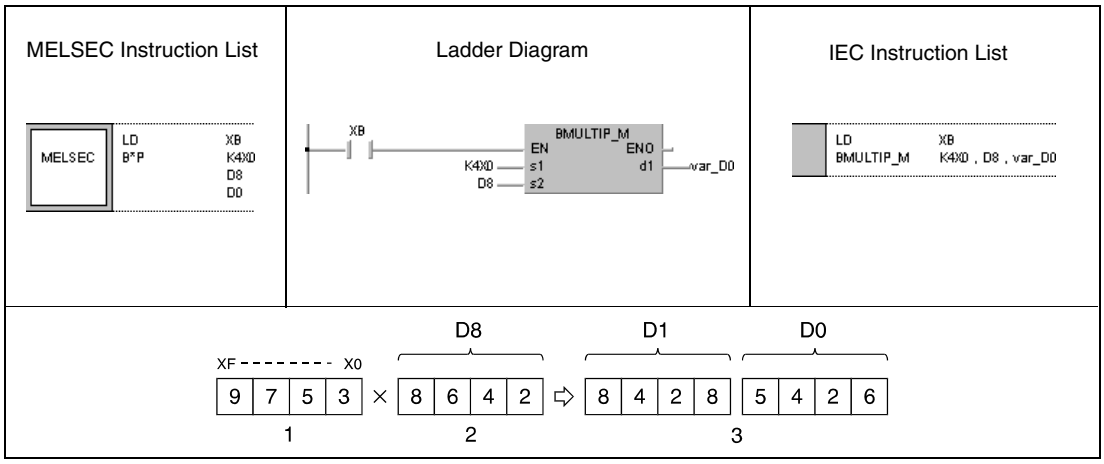
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The s1 or s2 BCD data is outside the 0 to 9999 range. (Error code 4101)
- Division by 0 (Error code 4100)

Program Example 1

BxP

With leading edge from XB, the following program multiplies BCD data at X0 through XF with BCD data in D8. The result is stored in D0 and D1.



[1] Multiplicand

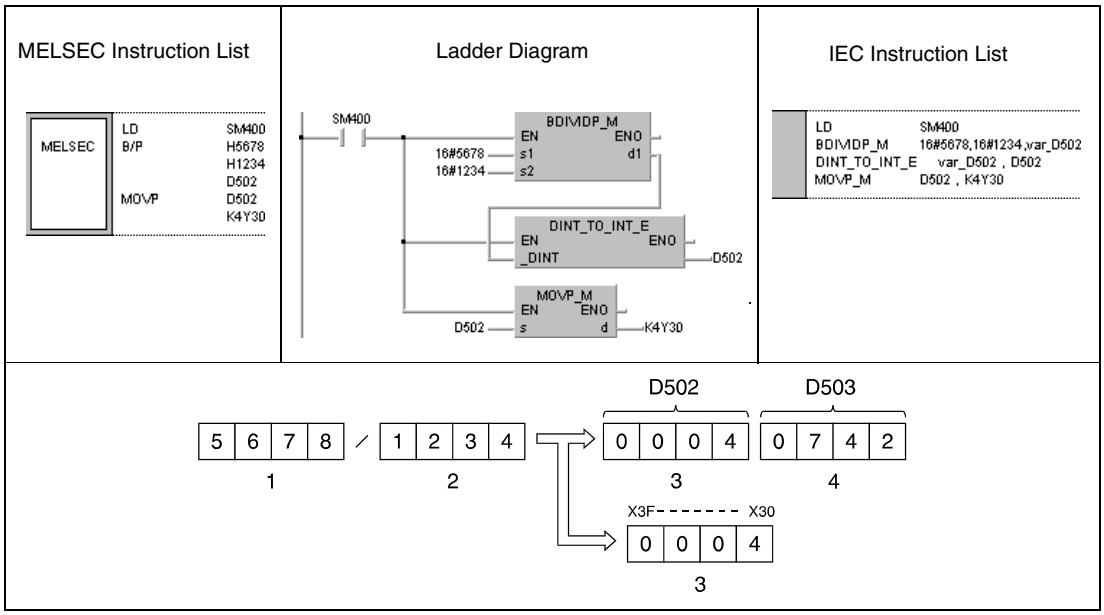
[2] Multiplier

[3] Result of multiplication

Program Example 2

B/P

The following program divides BCD data 5678 by BCD data 1234. The result is stored in D502 and the remainder is stored in D503. The last program step outputs the quotient in D502 at Y30 through Y3F.



[1] Dividend

[2] Divisor

[3] Quotient

[4] Remainder

NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.2.8 DBx, DBxP, DB/, DB/P

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

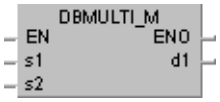
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	—	—	—	—	—	—

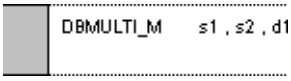
GX IEC Developer

MELSEC Instruction List

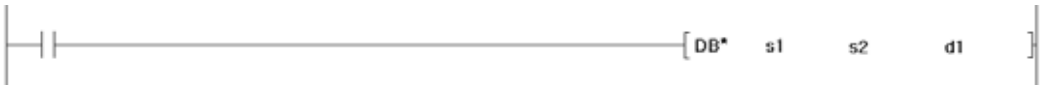
Ladder Diagram



IEC Instruction List



GX Works2



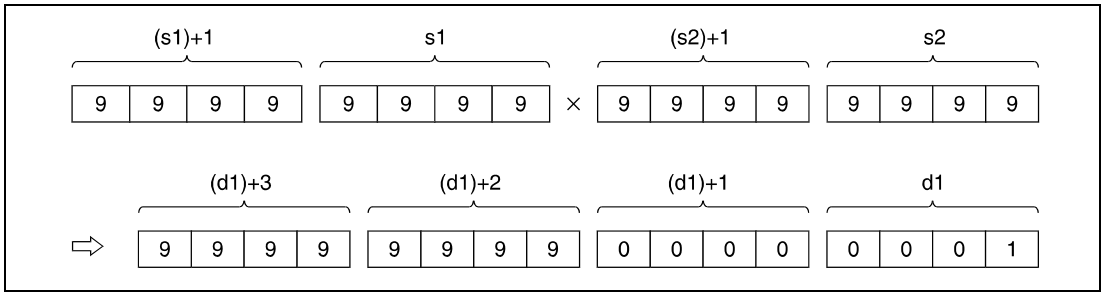
Variables

Set DataSet Data	Meaning	Data Type
s1	Data that will be multiplied or divided, or first number of device storing data that will be multiplied or divided	BCD 8-digit
s2	Data to multiply or divide by, or first number of device storing such data	BCD 8-digit
d1	First number of device storing the operation results of multiplication or division operation	BCD 16-digit

Functions **BCD 8-digit multiplication and division operations**

DBx BCD multiplication (8-digit)

BCD 8-digit data in s1 is multiplied with BCD 8-digit data in s2. The result is stored in d1.



If the result in d1 is a bit device, designation is made from the lower bits.

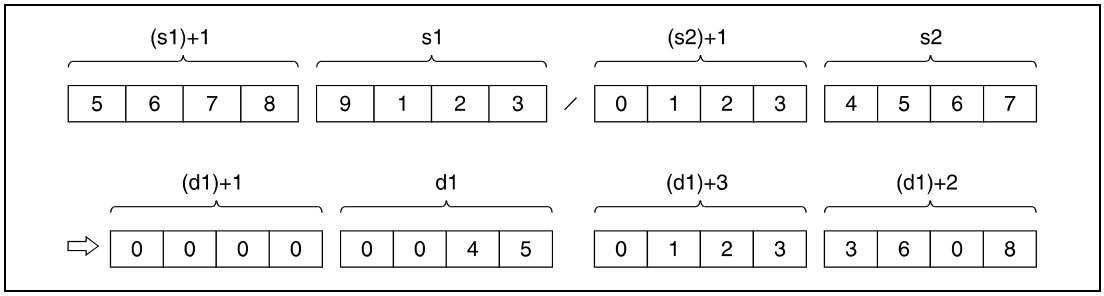
Example:

K1: lower 4 bits (b0 to b3)
K4: lower 16 bits (b0 to b15)
K8: 32 bits (b0 to b31)

BCD 8-digit data designated by s1 and s2 have to range within 0 and 99999999. Undesignated digits are read as 0 (e.g. 12345 = 00012345).

DB/ BCD division (8-digit)

BCD 8-digit data in s1 is divided by BCD 8-digit data in s2. The result is stored in d1.



The result of the division is stored in two 32-bit WORD arrays. The lower array stores the quotient (BCD 8-digit) and the upper array stores the remainder (BCD 8-digit).

If d is a bit device, the remainder of the division is not stored.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The s1 or s2 BCD data is outside the 0 to 99999999 range. (Error code 4101)
- Division by 0 (Error code 4100)

Program Example 1

DBxP

The following program multiplies BCD data 68347125 with BCD data 576682. The result is stored in D502 through D505. The following program step outputs the upper eight digits (D504, D505) at Y30 through Y4F.

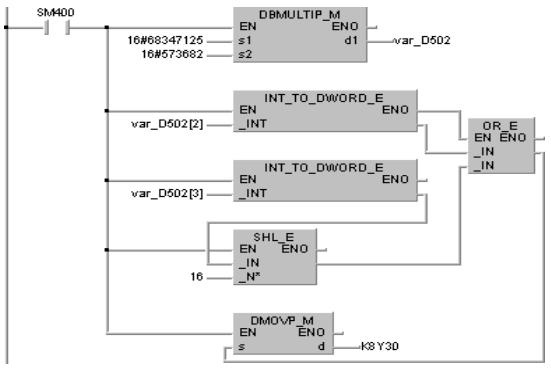
MELSEC Instruction List

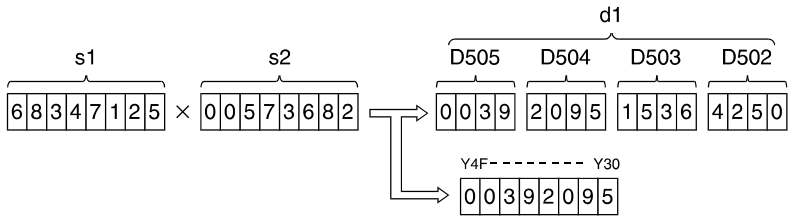
```
MELSEC   LD      SM400
         DB*P    H68347125
                 H573682
                 D502
         DMOV    D504
                 K8Y30
```

IEC Instruction List

```
         LD              SM400
         DBMULTIP_M      16#68347125,16#573682,var_D502
         INT_TO_DWORD_E  var_D502[3] , var_temp
         INT_TO_DWORD_E  var_D502[4] , var_D504
         SHL_E           var_D504 , K16 , var_D504
         LD              var_temp
         OR              var_D504
         ST              var_D504
         LD              SM400
         DMOV_M          var_D504 , K8Y30
```

Ladder Diagram

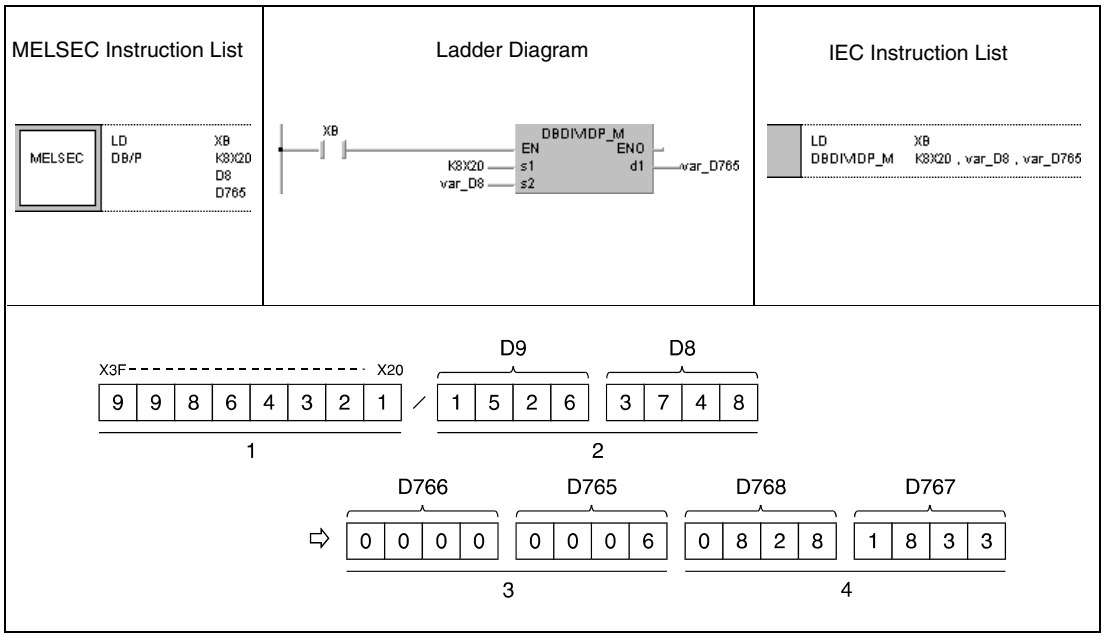




Program Example 2

DB/P

With leading edge from XB, the following program divides BCD data at X20 through X3F by BCD data in D8 and D9. The result is stored in D765 through D768.



[1] Dividend
[2] Divisor
[3] Quotient
[4] Remainder

NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.2.9 E+, E+P, E-, E-P

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

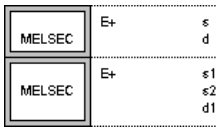
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—
s1	—	●	●	—	●	●	●[1)]	●	—
s2	—	●	●	—	●	●	●[1)]	●	—
d1	—	●	●	—	●	●	●[1)]	—	—

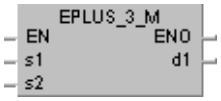
[1] Available only in multiple Universal model QCPU and LCPU

GX IEC Developer

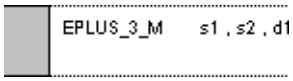
MELSEC Instruction List



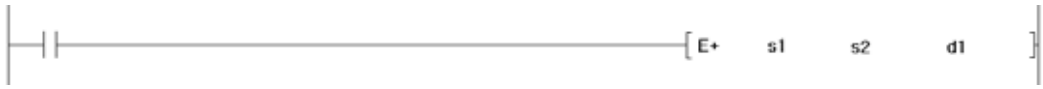
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	Addition or subtraction data, or first number of device storing addition or subtraction data	Real number
d	Data to be added to or subtracted from, or first number of device storing such data	
s1	Data to be added to or subtracted from, or first number of device storing such data	
s2	Addition or subtraction data, or first number of device storing addition or subtraction data	
d1	First number of device storing addition or subtraction data	

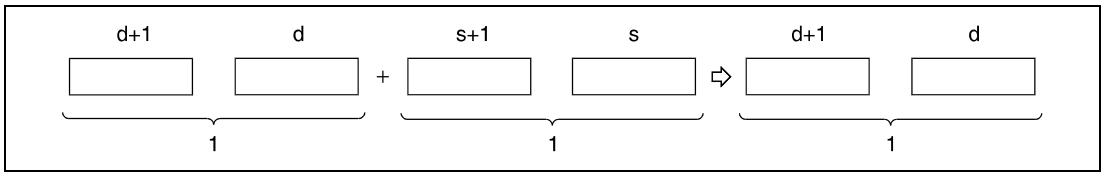
NOTE *Within the IEC editors please use the IEC commands.*

Functions **Floating point data addition and subtraction operations (single precision)**

E+ 32-bit floating point data addition

● Variation 1:

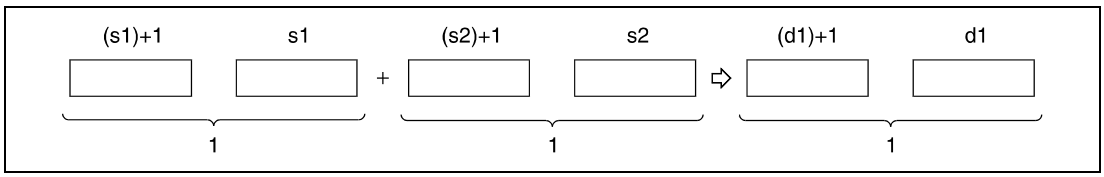
Floating point data in d is added to floating point data in s. The result is stored in d.



[1] 32-bit floating point data, data type real number

● Variation 2:

Floating point data in s1 is added to floating point data in s2. The result is stored in d1.



[1] 32-bit floating point data, data type real number

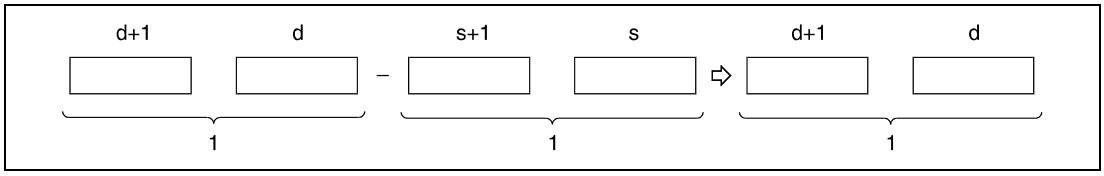
Floating point data designated by s, d, s1, s2, and d1 have to range within:

$0, \pm 2^{-126} \leq (s, d, s1, s2, d1) < \pm 2^{128}$

E- 32-bit floating point data subtraction

● Variation 1:

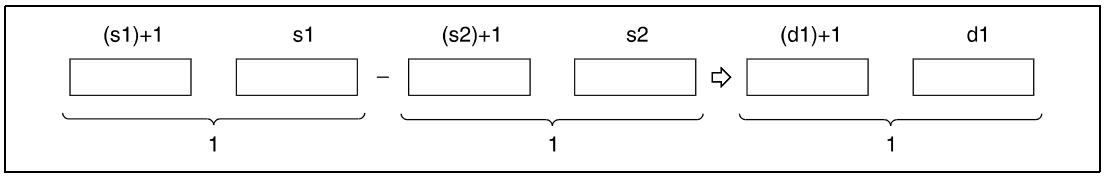
Floating point data in s is subtracted from floating point data in d. The result is stored in d.



[1] 32-bit floating point data, data type real number

- Variation 2:

Floating point data in s2 is subtracted from floating point data in s1. The result is stored in d1.



[1] 32-bit floating point data, data type real number

Floating point data designated by s, d, s1, s2, and d1 have to range within:

$0, \pm 2^{-126} \leq (s, d, s1, s2, d1) < \pm 2^{128}$

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The contents of the designated device or the result of the addition are not zero and not within the following range (Error code 4100):

  $\pm 2^{-126} \leq$ (Contents of designated device) $< \pm 2^{128}$

  (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)

- The value of the designated device is –0.
  (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code 4100)
  NOTE: There are CPU modules that will not result in an operation error if –0 is specified. Refer to section 3.5.1 for details.

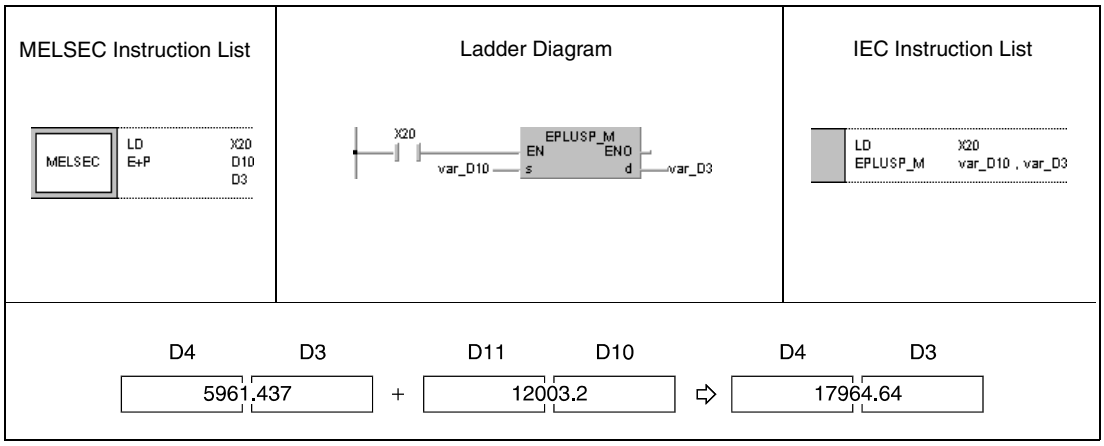
- The result of addition and subtraction exceeds the following range (overflow occurs):
  (For the Universal model QCPU, LCPU)
  $-2^{128} \leq$ (Result of addition and subtraction) $\leq 2^{128}$
  (Error code 4141)

- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm \infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example 1

E+P (s, d)

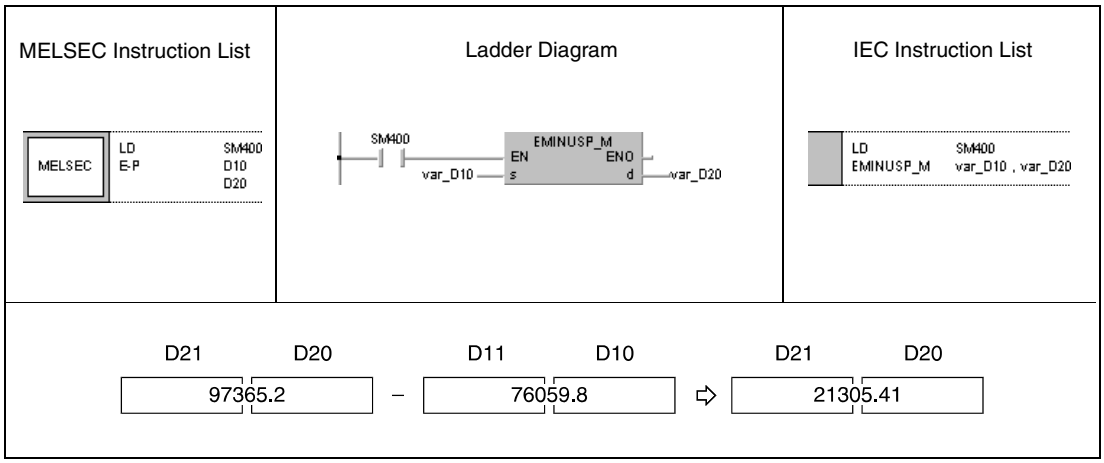
With leading edge from X20, the following program adds floating point data in D3 and D4 to floating point data in D10 and D11. The result is stored in D3 and D4.



Program Example 2

E-P (s, d)

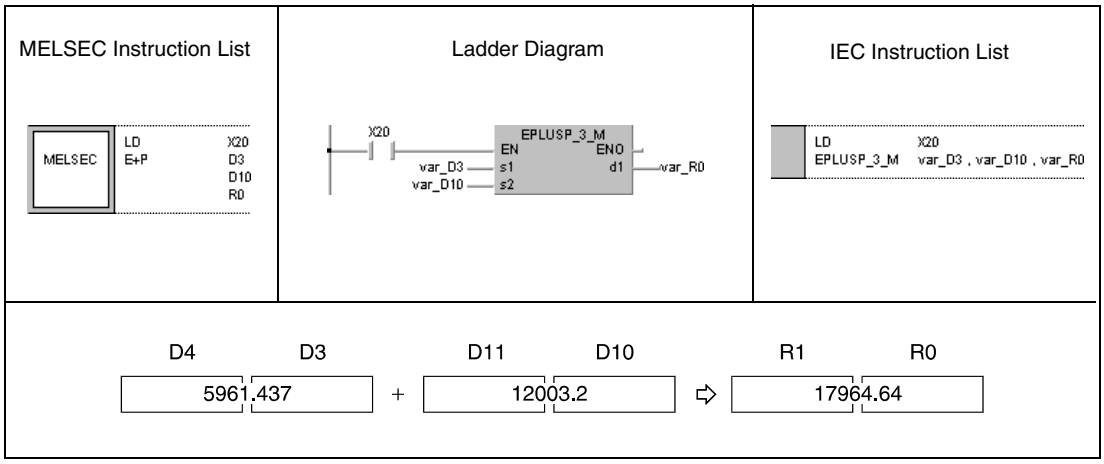
The following program subtracts floating point data in D10 and D11 from floating point data in D20 and D21. The result is stored in D20 and D21.



Program Example 3

E+P (s1, s2, d)

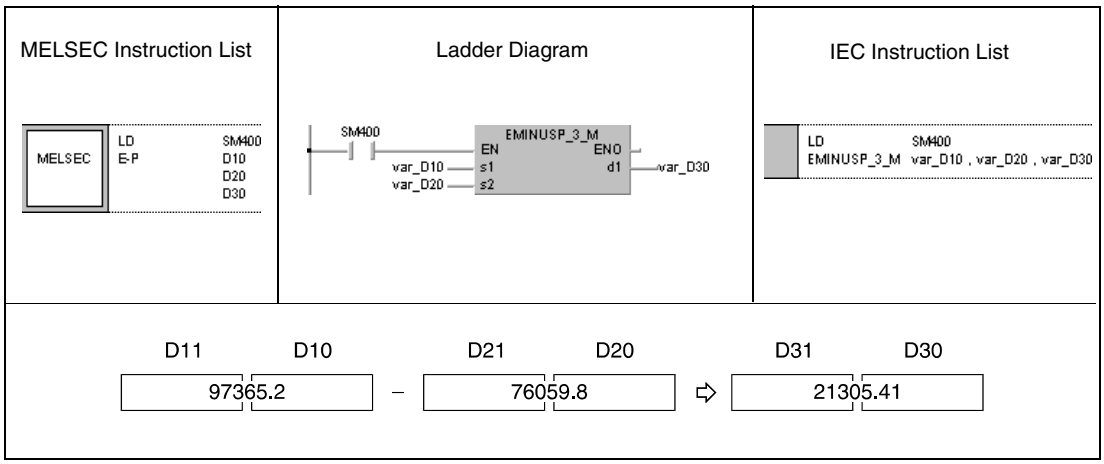
With leading edge from X20, the following program adds floating point data in D3 and D4 to floating point data in D10 and D11. The result is stored in R0 and R1.



Program Example 4

E-P (s1, s2, d)

The following program subtracts floating point data in D20 and D21 from floating point data in D10 and D11. The result is stored in D30 and D31.



NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.2.10 ED+, ED+P, ED-, ED-P

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

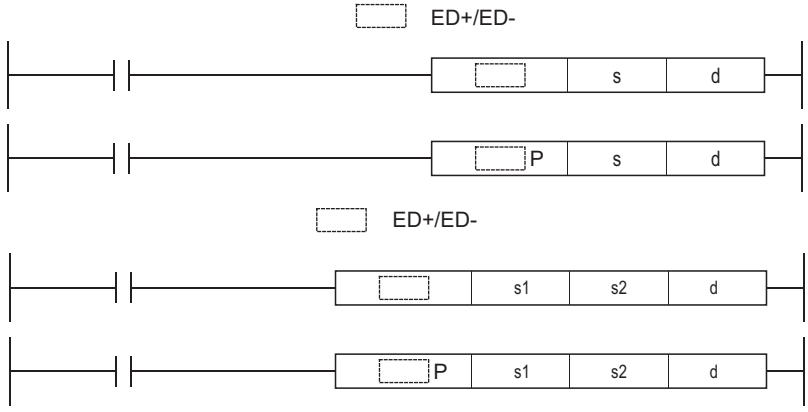
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	●	—
d1	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s	Addition or subtraction data, or first number of device storing addition or subtraction data	Real number
d	Data to be added to or subtracted from, or first number of device storing such data	
s1	Data to be added to or subtracted from, or first number of device storing such data	
s2	Addition or subtraction data, or first number of device storing addition or subtraction data	
d1	First number of device storing addition or subtraction data	

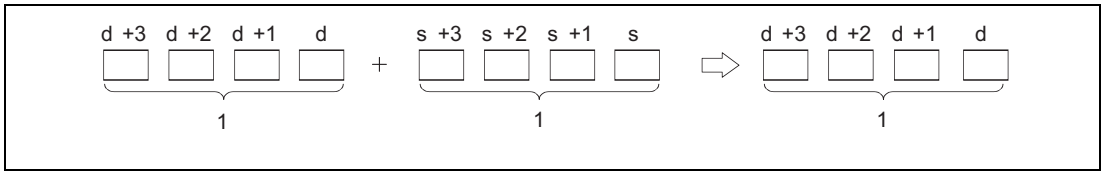
NOTE *Within the IEC editors please use the IEC commands.*

Functions **Floating point data addition and subtraction operations (double precision)**

ED+ 64-bit floating point data addition

● Variation 1:

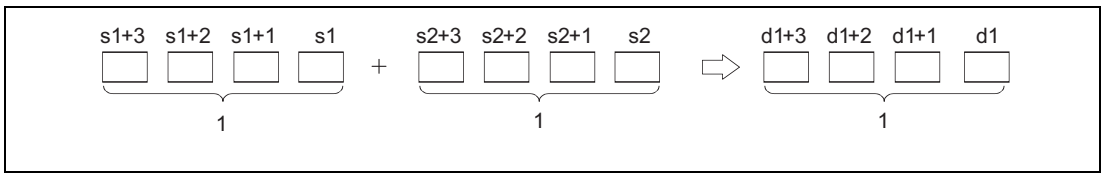
Floating point data in d is added to floating point data in s. The result is stored in d.



[1] 64-bit floating point data, data type real number

● Variation 2:

64-bit floating point data in s1 is added to floating point data in s2. The result is stored in d1.



[1] 64-bit floating point data, data type real number

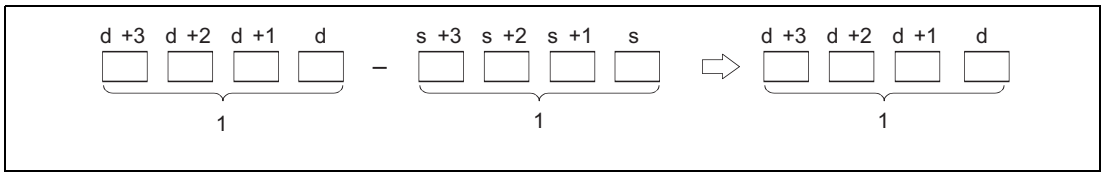
Floating point data designated by s, d, s1, s2, and d1 have to range within:

$0, \pm 2^{-1022} \leq (s, d, s1, s2, d1) < \pm 2^{1024}$

ED- 64-bit floating point data subtraction

● Variation 1:

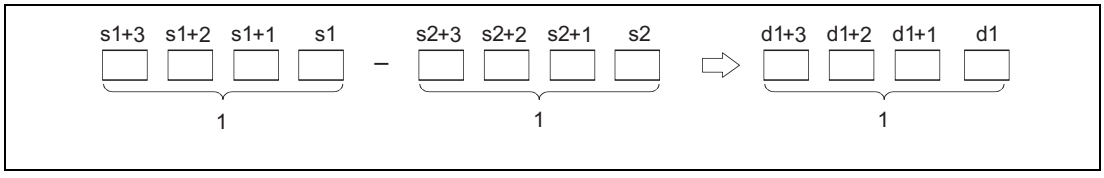
Floating point data in s is subtracted from floating point data in d. The result is stored in d.



[1] 64-bit floating point data, data type real number

● Variation 2:

Floating point data in s2 is subtracted from floating point data in s1. The result is stored in d1.



[1] 64-bit floating point data, data type real number

Floating point data designated by s, d, s1, s2, and d1 have to range within:

$0, \pm 2^{-1022} \leq (s, d, s1, s2, d1) < \pm 2^{1024}$

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

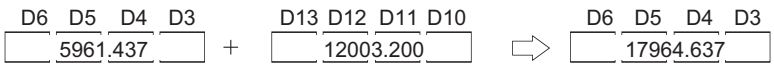
- The contents of the designated device or the result of the addition are not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Contents of designated device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the designated device is –0.
  (Error code 4140)
- The result of addition and subtraction exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Result of addition and subtraction) $\leq 2^{1024}$
  (Error code 4141)

Program Example 1

ED+P (s, d)

With leading edge from X20, the following program adds 64-bit floating point data in D3 to D6 to 64-bit floating point data in D10 to D13. The result is stored in D3 to D6.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

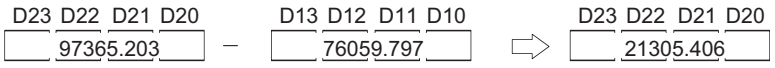


Program Example 2

ED-P (s, d)

The following program subtracts 64-bit floating point data in D10 to D13 from 64-bit floating point data in D20 to D23. The result is stored in D20 to D23.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

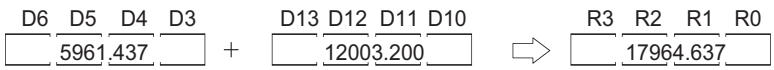


Program Example 3

ED+P (s1, s2, d)

With leading edge from X20, the following program adds 64-bit floating point data in D3 to D6 to 64-bit floating point data in D10 to D13. The result is stored in R0 to R3.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

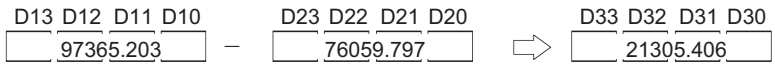


Program Example 4

ED-P (s1, s2, d)

The following program subtracts 64-bit floating point data in D20 to D23 from 64-bit floating point data in D10 to D13. The result is stored in D30 to D33.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.2.11 Ex, ExP, E/, E/P

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

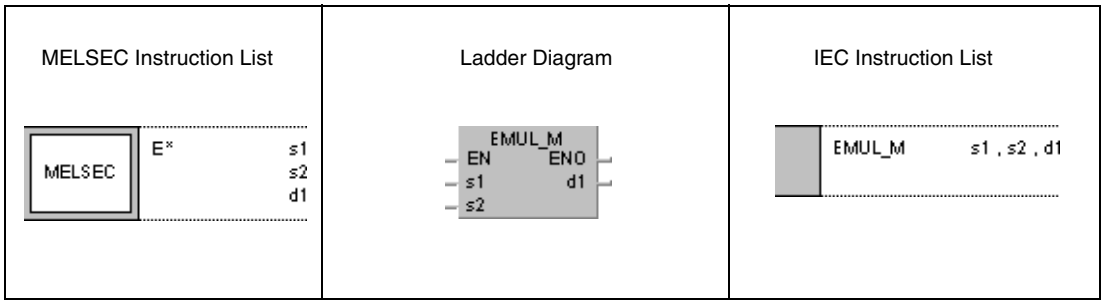
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

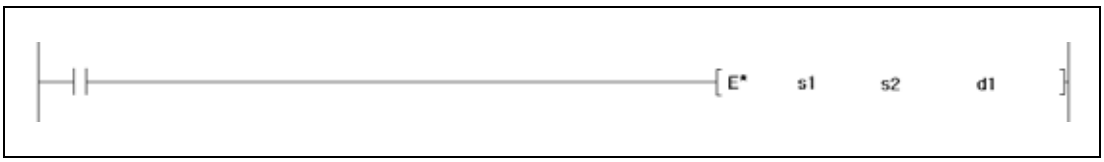
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants E	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	●	●	●[1)]	●	—
s2	—	●	●	—	●	●	●[1)]	●	—
d1	—	●	●	—	●	●	●[1)]	—	—

[1] Available only in multiple Universal model QCPU and LCPU

GX IEC Developer



GX Works2



Variables

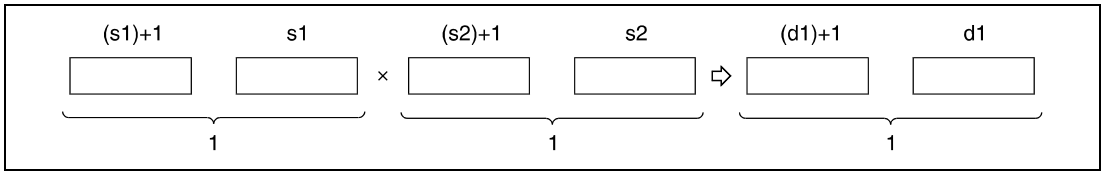
Set Data	Meaning	Data Type
s1	Data that will be multiplied or divided, or first number of device storing data that will be multiplied or divided	Real number
s2	Data to multiply or divide by, or first number of device storing such data	
d1	First number of device storing the operation results of multiplication or division operation	

Functions

Floating point data multiplication and division operations (single precision)

Ex 32-bit floating point data multiplication

Floating point data in s1 is multiplied with floating point data in s2. The result is stored in d1.



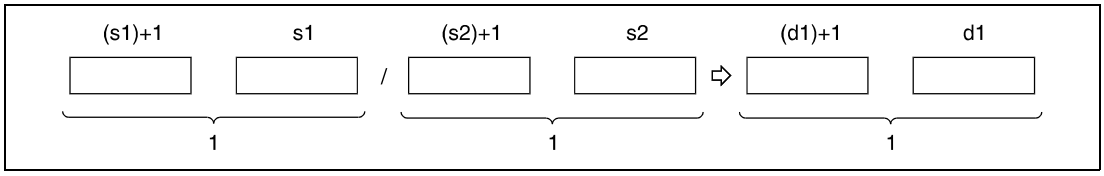
[1] 32-bit floating point data, data type real number

Floating point data designated by s1, s2, and d1 have to range within:

$0, \pm 2^{-126} \leq (s1, s2, d1) < \pm 2^{128}$

E/ 64-bit floating point data division

Floating point data in s1 is divided by floating point data in s2. The result is stored in d1.



[1] 32-bit floating point data, data type real number

Floating point data designated by s1, s2, and d1 have to range within:

$0, \pm 2^{-126} \leq (s1, s2, d1) < \pm 2^{128}$

Operation Errors

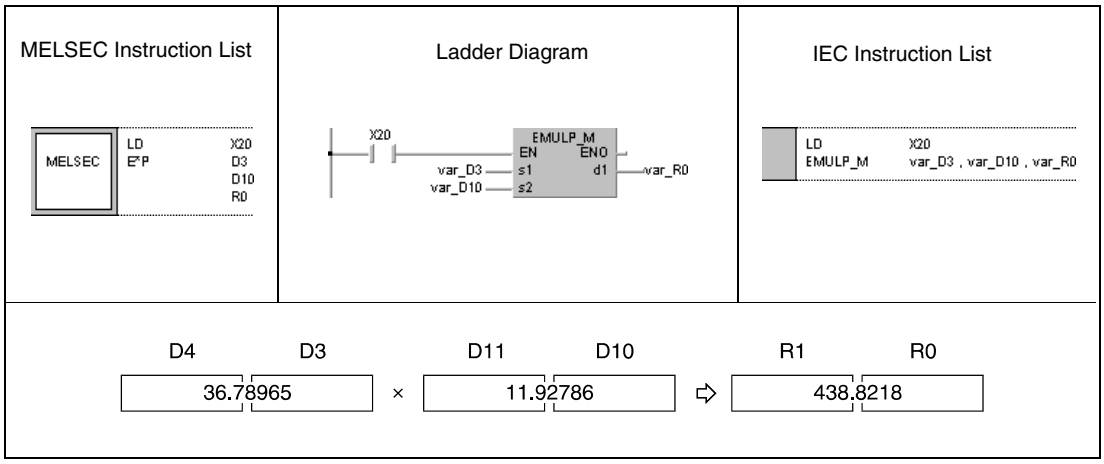
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The contents of the designated device or the result of the operation is not zero and not within the following range:
  $\pm 2^{-126} \leq$ (Contents of designated device or operation result) $< \pm 2^{128}$
  (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code 4100)
- The value of the specified device is –0.
  (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code 4100)
  NOTE: There are CPU modules that will not result in an operation error if –0 is specified. Refer to page 3-17 for details.
- Division by 0 (Error code 4100)
- The result of multiplication and division exceeds the following range (overflow occurs):
  (For the Universal model QCPU, LCPU)
  $-2^{128} \leq$ (Result of multiplication and division) $\leq 2^{128}$
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm \infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example 1

ExP

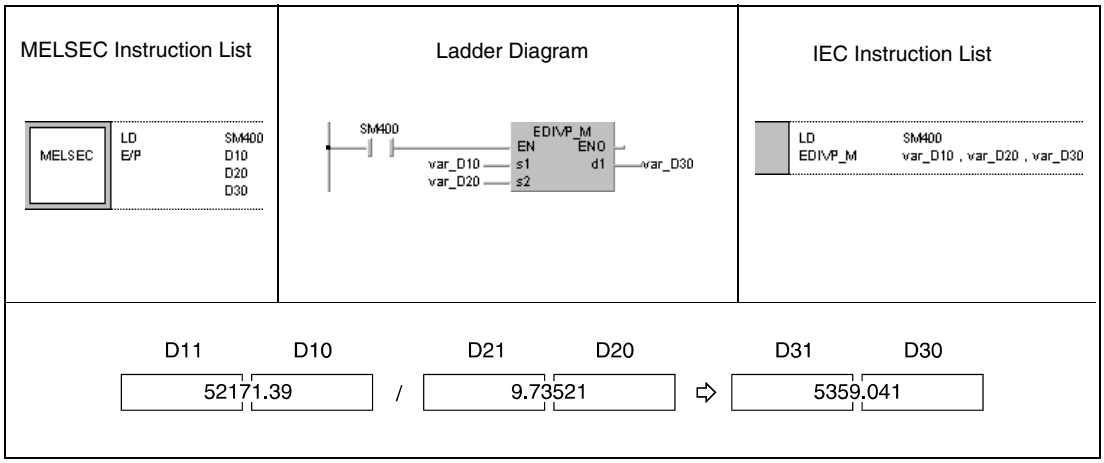
With leading edge from X20, the following program multiplies floating point data in D3 and D4 with floating point data in D10 and D11. The result is stored in R0 and R1.



Program Example 2

E/P

The following program divides floating point data in D10 an D11 by floating point data in D20 and D21. The result is stored in D30 and D31.



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.2.12 EDx, EDxP, ED/, ED/P

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

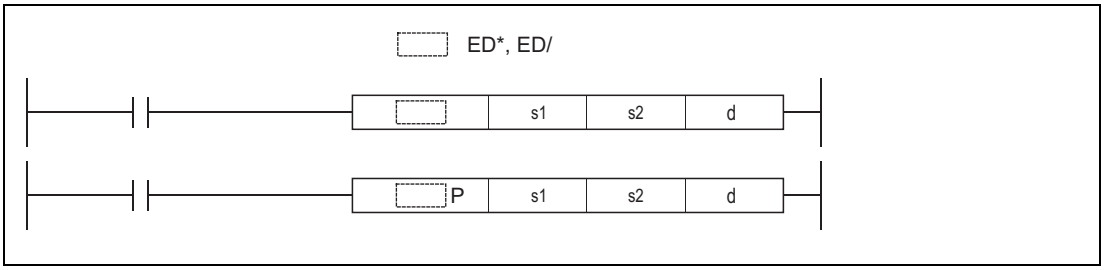
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants E	Other
	Bit	Word		Bit	Word				U
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	●	—
d1	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s1	Data that will be multiplied or divided, or first number of device storing data that will be multiplied or divided	Real number
s2	Data to multiply or divide by, or first number of device storing such data	
d1	First number of device storing the operation results of multiplication or division operation	

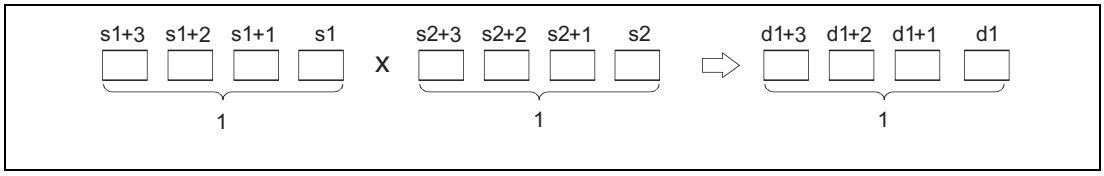
NOTE *Within the IEC editors please use the IEC commands.*

Functions

Floating point data multiplication and division operations (double precision)

EDx 64-bit floating point data multiplication

Floating point data in s1 is multiplied with floating point data in s2. The result is stored in d1.



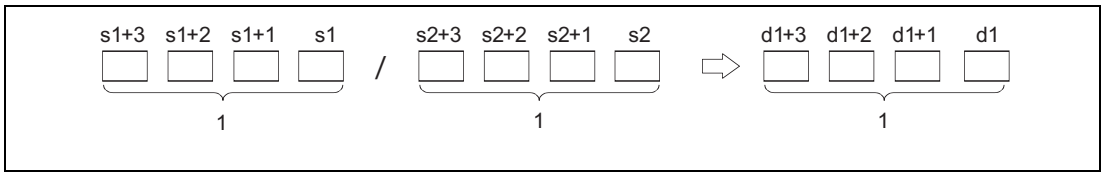
[1] 64-bit floating point data, data type real number

Floating point data designated by s1, s2, and d1 have to range within:

$0, \pm 2^{-1022} \leq (s1, s2, d1) < \pm 2^{1024}$

ED/ 64-bit floating point data division

Floating point data in s1 is divided by floating point data in s2. The result is stored in d1.



[1] 64-bit floating point data, data type real number

Floating point data designated by s1, s2, and d1 have to range within:

$0, \pm 2^{-1022} \leq (s1, s2, d1) < \pm 2^{1024}$

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

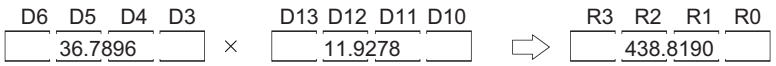
- The contents of the designated device or the result of the operation is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Contents of designated device or result of operation) $< \pm 2^{1024}$
  (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4140)
- The value of the designated device is –0.
  (Error code 4140)
- Division by 0
  (Error code 4100)
- The result of multiplication or division exceeds the following range. (The overflow occurs.)
  (For the Universal model QCPU, LCPU)
  $-2^{1024} \leq$ (Result of multiplication or division) $\leq 2^{1024}$
  (Error code 4141)

Program Example 1

EDxP

With leading edge from X20, the following program multiplies 64-bit floating point data in D3 to D6 with 64-bit floating point data in D10 to D13. The result is stored in R0 to R3.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

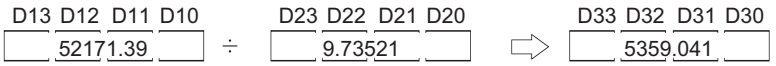


Program Example 2

ED/P

The following program divides 64-bit floating point data in D10 to D13 by 64-bit floating point data in D20 to D23. The result is stored in D30 to D33.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.2.13 BK+, BK+P, BK-, BK-P

CPU

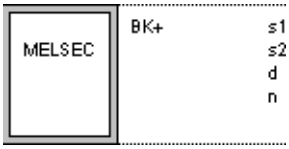
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

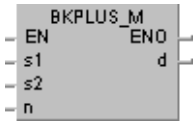
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

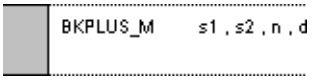
MELSEC Instruction List



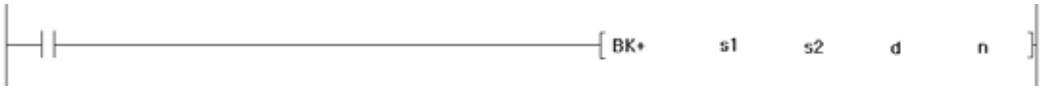
Ladder Diagram



IEC Instruction List



GX Works2



Variables

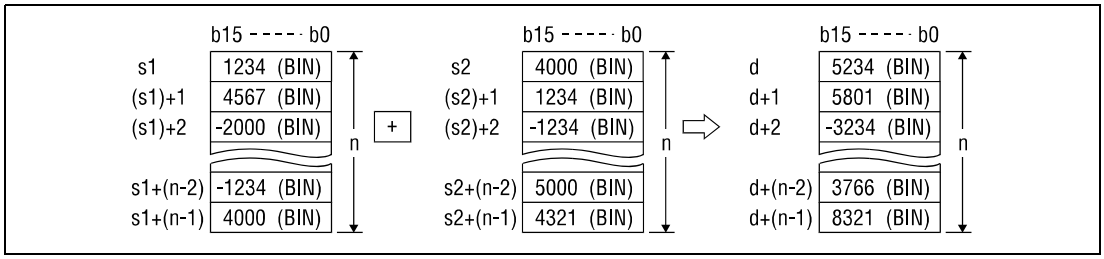
Set Data	Meaning	Data Type
s1	Data to be added to or subtracted from, or first number of device storing such data	BIN 16-bit
s2	Addition or subtraction data, or first number of device storing addition or subtraction data	
d	First number of device storing results of operation	
n	Number of data blocks	

Functions **BIN 16-bit data addition and subtraction operations**

BK+ BIN 16-bit data block addition

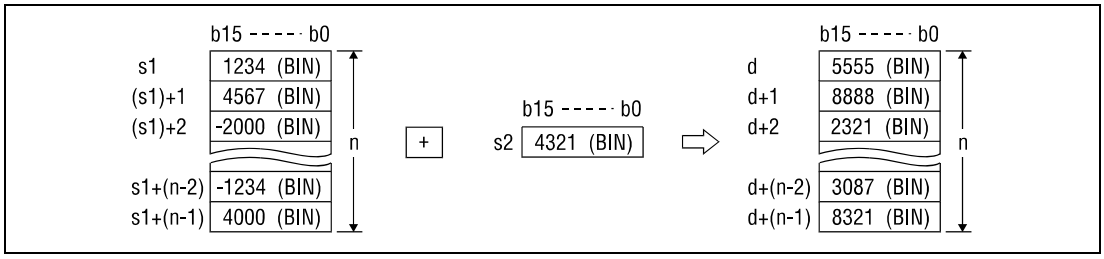
An addition operation instruction for BIN 16-bit data block data consists of the instruction itself, two designated devices s1 and s2 to be added, a device d to store the result, and the number of data blocks to be added.

It adds the nth 16-bit block in s1 to the nth 16-bit block in s2, beginning with the first number of device. The result of each block addition is stored in d.



The addition operation is conducted in 16-bit units.

The constant designated by s2 must be BIN 16-bit data ranging from -32768 to 32767.



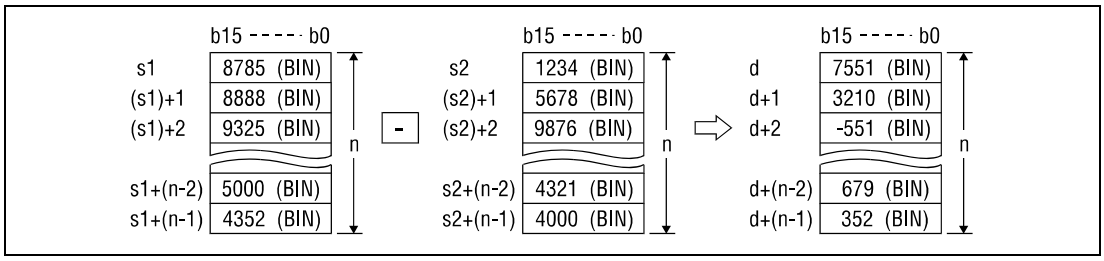
The most significant bit of each block determines, whether data in s1, s2 or d are positive (bit = 0) or negative (bit = 1).

If the least significant bit of a block is fallen below or the most significant bit of a block is exceeded, the carry flag is not set.

BK- BIN 16-bit data block subtraction

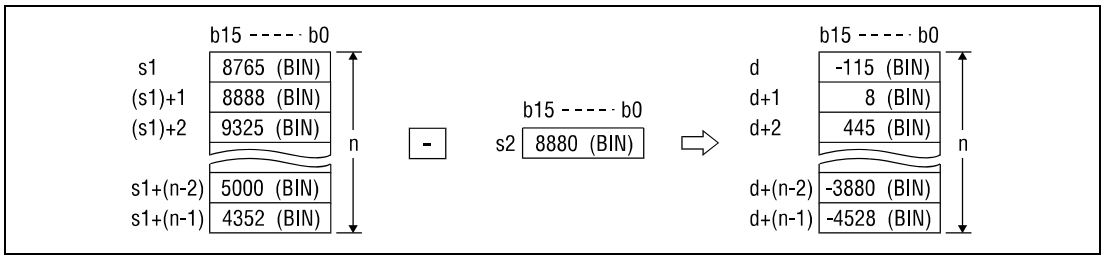
A subtraction operation instruction for BIN 16-bit data block data consists of the instruction itself, two designated devices s1 and s2 to be added, a device d to store the result, and the number of data blocks to be subtracted.

It subtracts the nth 16-bit block in s2 from the nth 16-bit block in s1, beginning with the first number of device. The result of each block addition is stored in d.



The subtraction operation is conducted in 16-bit units.

The constant designated by s2 must be BIN 16-bit data ranging from -32768 to 32767.



The most significant bit of each block determines, whether data in s1, s2 or d are positive (bit = 0) or negative (bit = 1).

If the least significant bit of a block is fallen below or the most significant bit of a block is exceeded, the carry flag is not set.

Operation Errors

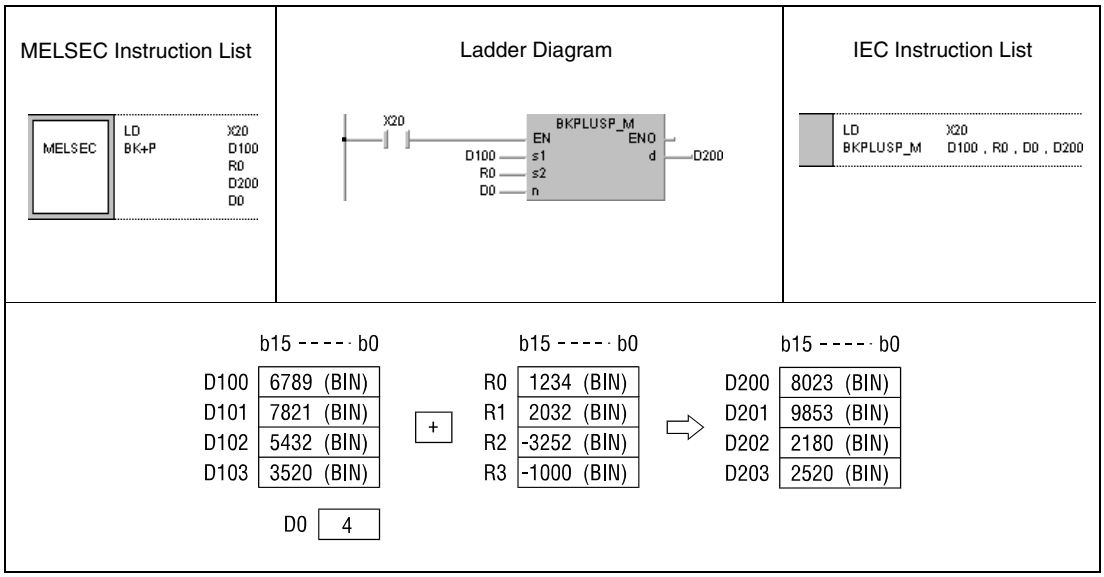
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of data blocks in s1, s2 or d exceeds the relevant device range. (Error code 4101)
- The device ranges of s1 and s2 overlap. (Except when the same device is assigned to s1 and d) (Error code: 4101)
- The device ranges of s2 and d overlap. (Except when the same device is assigned to s2 and d) (Error code: 4101)

Program Example 1

BK+P

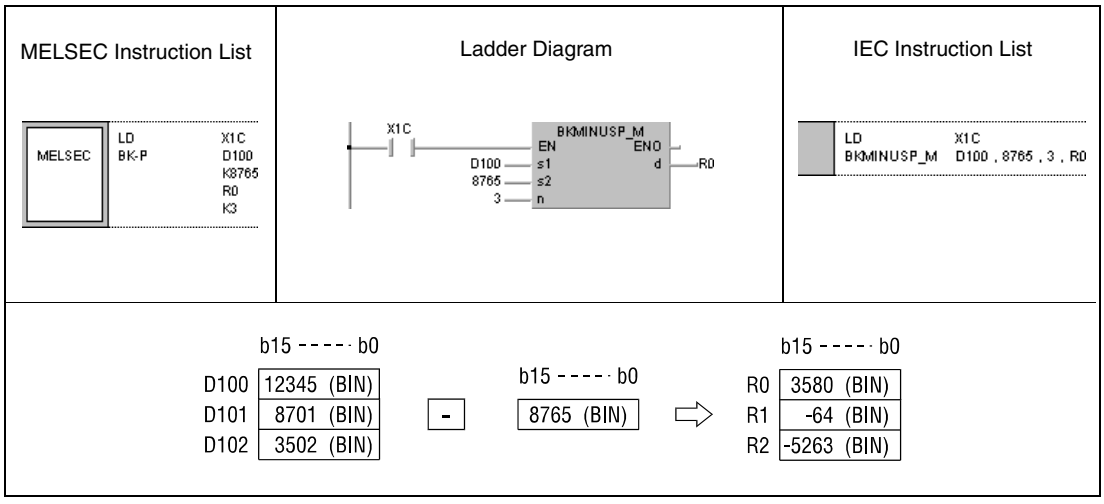
With leading edge from X20, the following program adds BIN block data beginning from D100 to BIN block data beginning from R0. The result of the operation is stored beginning from D200. The number of blocks (4) added is stored in D0.



Program Example 2

BK-P

With leading edge from X1C, the following program subtracts a constant 8765 from BIN block data beginning from D100. The result of the operation is stored beginning from R0. The number of data blocks (3) subtracted is designated by a constant K3.



6.2.14 DBK+, DBK+P, DBK-, DBK-P

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

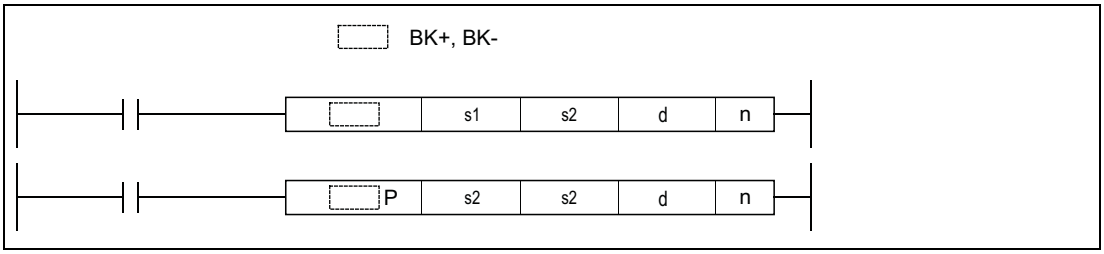
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n	—	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s1	Data to be added to or subtracted from, or first number of device storing such data	BIN 16-bit
s2	Addition or subtraction data, or first number of device storing addition or subtraction data	
d	First number of device storing results of operation	
n	Number of data blocks	

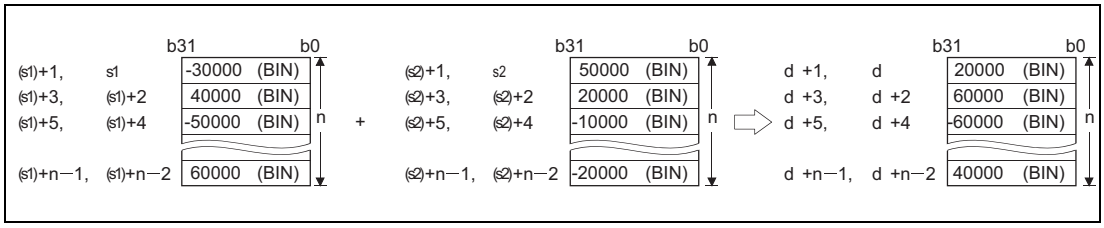
NOTE *Within the IEC editors please use the IEC commands.*

Functions **BIN 32-bit data block addition and subtraction operations**

DBK+ BIN 32-bit data block addition

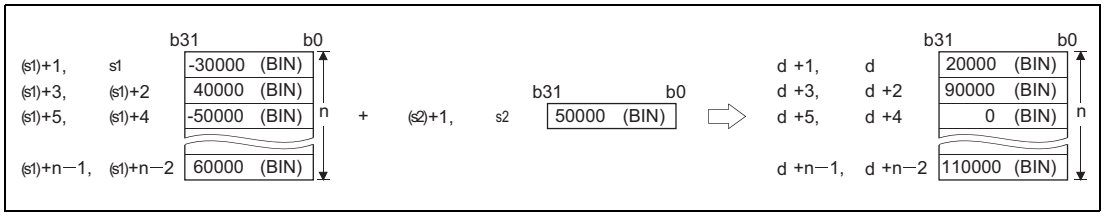
An addition operation instruction for BIN 32-bit data block data consists of the instruction itself, two designated devices s1 and s2 to be added, a device d to store the result, and the number of data blocks to be added.

It adds the nth 32-bit block in s1 to the nth 32-bit block in s2, beginning with the first number of device. The result of each block addition is stored in d.



The addition operation is conducted in 32-bit units.

The constant designated by s1 must be BIN 32-bit data ranging from -2147483648 to 2147483647.



If the value specified by n is 0, the instruction will be not processed.

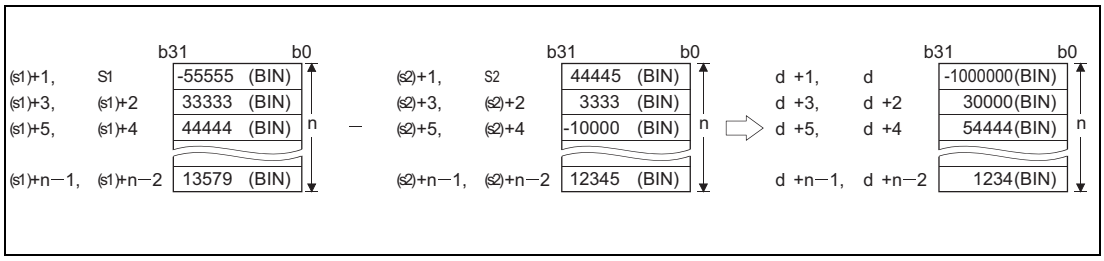
The most significant bit of each block determines, whether data in s1, s2 or d are positive (bit = 0) or negative (bit = 1).

If the least significant bit of a block is fallen below or the most significant bit of a block is exceeded, the carry flag is not set.

DBK- BIN 32-bit data block subtraction

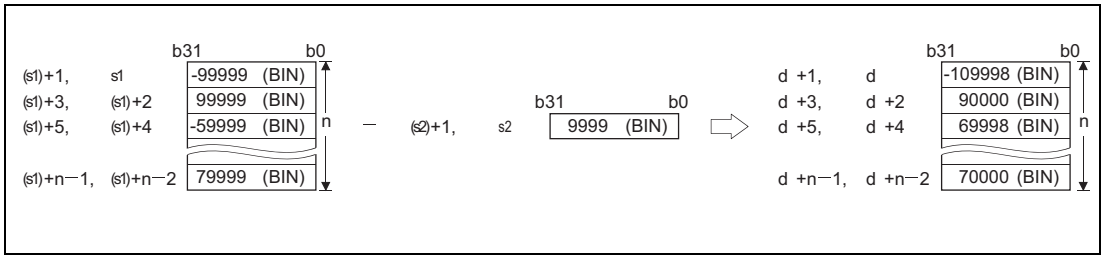
A subtraction operation instruction for BIN 32-bit data block data consists of the instruction itself, two designated devices s1 and s2 to be subtracted, a device d to store the result, and the number of data blocks to be subtracted.

It subtracts the nth 32-bit block in s2 from the nth 32-bit block in s1, beginning with the first number of device. The result of each block addition is stored in d.



The subtraction operation is conducted in 32-bit units.

The constant designated by s2 must be BIN 32-bit data ranging from –2147483648 to 2147483647.



If the value specified by n is 0, the instruction will be not processed.

The most significant bit of each block determines, whether data in s1, s2 or d are positive (bit = 0) or negative (bit = 1).

If the least significant bit of a block is fallen below or the most significant bit of a block is exceeded, the carry flag is not set.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

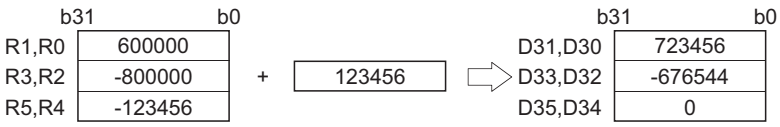
- A negative value is specified for n. (Error code 4100)
- The range of the n-point devices starting from the device specified by s1, s2, or d exceeds the specified device range.
  (Error code: 4101)
- The range of the n-point devices starting from the device specified by s1 overlaps with the range of the n-point devices starting from the device specified by d.
  (Except when s1 and d specify the same device.)
  (Error code: 4101)
- The range of the n-point devices starting from the device specified by s2 overlaps with the range of the n-point devices starting from the device specified by d.
  (Except when s2 and d specify the same device.)
  (Error code: 4101)

Program Example 1

DBK+P

The following program adds the value data stored at R0 to R5 to the constant, and then stores the operation result into D30 to D35, when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

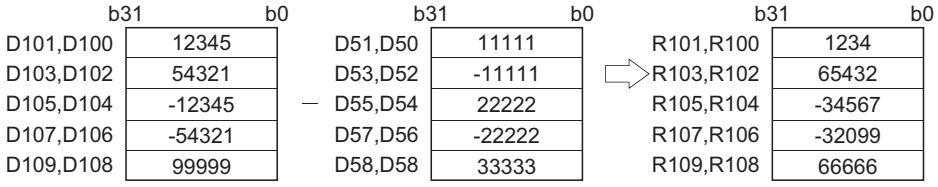


Program Example 2

DBK-P

The following program subtracts the value data stored at D50 to D59 from the value data stored at D100 to D109, and then stores the operation result into R100 to R109, when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



6.2.15 $+, $+P

CPU

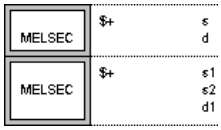
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

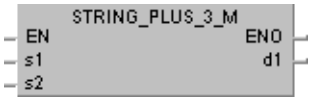
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	●	—
d1	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
STRING_PLUS_3_M    s1 , s2 , d1
```

GX Works2

Variables

Set Data	Meaning	Data Type
s	Data to be linked, or first number of device storing such data	Character string
d	First number of device storing results of operation	
s1	Data to be linked, or first number of device storing such data	
s2	Data to be linked, or first number of device storing such data	
d1	First number of device storing results of operation	

Functions

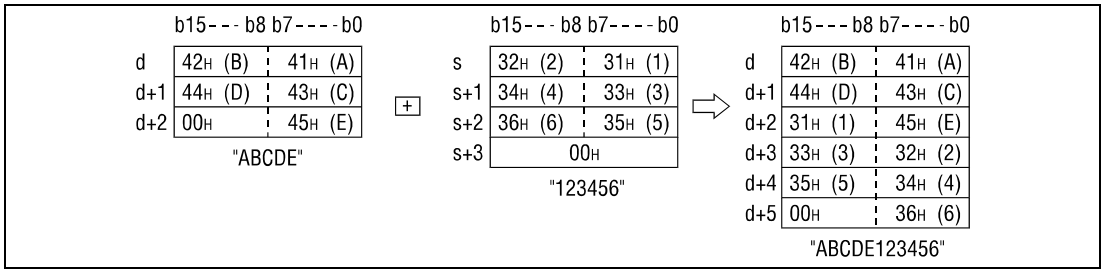
Character string linking operations

$+ Character string linking

● Variation 1:

Character string data in s is appended to character data in d. The linked character string is stored in d.

The linked character string begins with the character at the least significant byte in d and ends with the code "00H" in s.

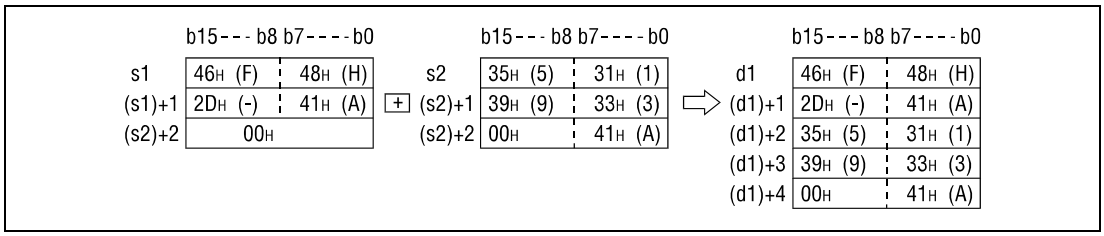


The code "00H" indicates the end of a character string. When two strings are linked, in the first string this code is ignored and the "00H" of the second string marks the end of the linked string.

● Variation 2:

Character string data in s2 is appended to character string data in s1. The linked character string is stored in d1.

The linked character string begins with the character at the least significant byte in s1 and ends with the code "00H" in s2.



The code "00H" indicates the end of a character string. When two strings are linked, in the first string this code is ignored and the "00H" of the second string marks the end of the linked string.

Operation Errors

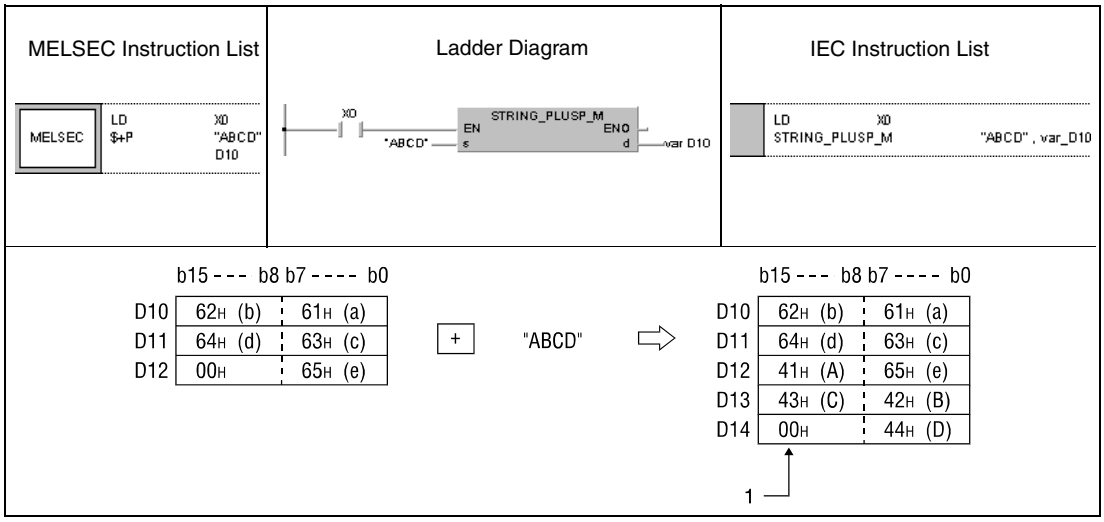
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The linked character string cannot be stored.
  (Error code 4100)
- The storage device numbers designated by s, d, s1, s2, and d1 overlap.
  (Error code 4101)
- The character string of s, d, s1, s2, and d1 exceeds 16383 characters.
  (Error code 4101)

Program Example 1

$+P

With leading edge from X0, the following program links character string data in D10 through D12 to the character string "ABCD". The linked character string is stored in D10 through D14.

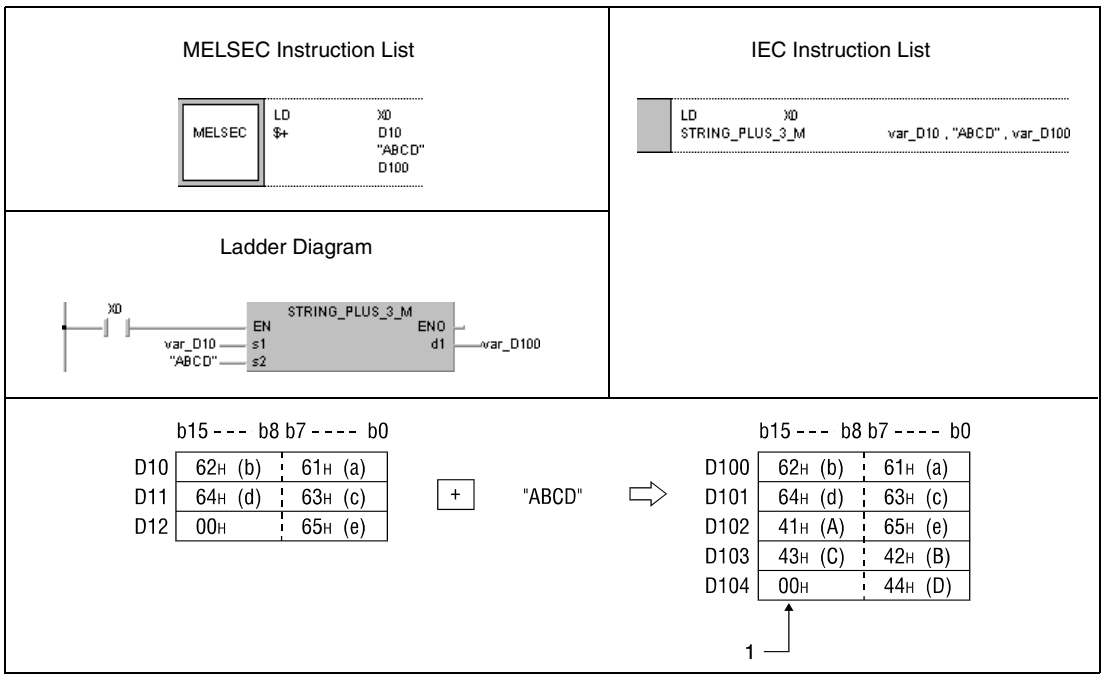


[1] "00H" indicates the end of character strings and is stored automatically.

Program Example 2

$+

While X0 is set (1), the following program links character string data in D10 through D12 to a character string "ABCD". The linked character string is stored from D101 through D104.



[1] "00H" indicates the end of character strings and is stored automatically.

NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.2.16 INC, INCP, DEC, DECP

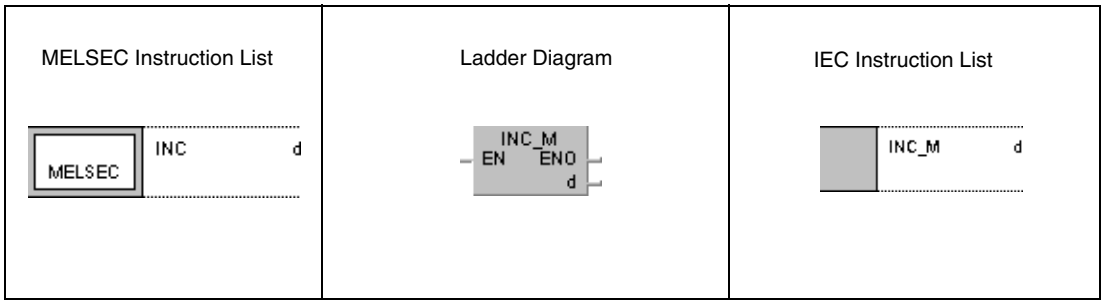
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

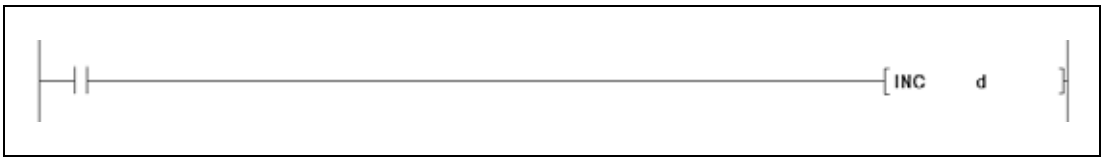
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
d	●	●	●	●	●	●	●	—	—

GX IEC Developer



GX Works2



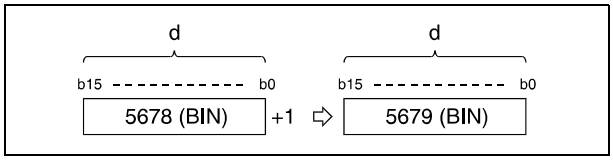
Variables

Set Data	Meaning	Data Type
d	First number of device conducted by INC (add 1) or DEC (subtract 1) operation.	BIN 16-bit

Functions **BIN 16-bit increment and decrement operations**

INC BIN 16-bit increment

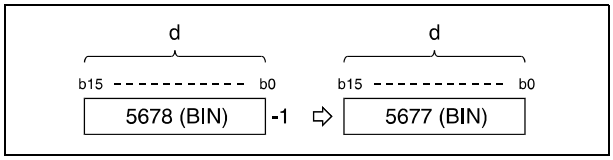
Adds 1 to device designated by d (16-bit).



If the content of d is 32767, the result after incrementing is -32768.

DEC BIN 16-bit decrement

Subtracts 1 from device designated by d (16-bit).



If the content of d is 0, the result after decrementing is -1.

If the content of d is -32768, the result after decrementing is 32767.

Program Example 1

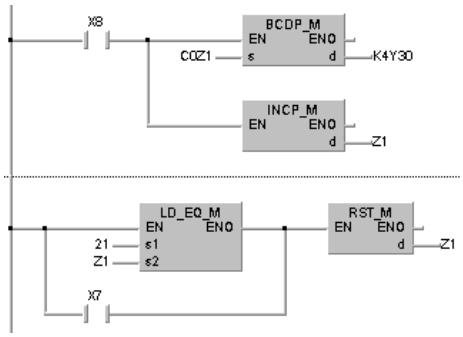
INCP

With leading edge from X8, the following program outputs the actual value of the counter (nominal value = 9999) C0 through C20 (C0 plus Z1) at Y30 through Y3F as BCD data. Z1 is reset (RST Z1), if Z1 is equal to 21 (LD = K21 Z1) or if the reset input X7 is set.

MELSEC Instruction List

```
MELSEC
LD    X8
BCDP  C0Z1
      K4Y30
INCP  Z1
LD=   K21
      Z1
OR    X7
RST   Z1
```

Ladder Diagram



IEC Instruction List

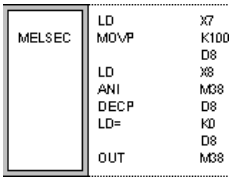
```
LD      X8
BCDP_M  C0Z1 , K4Y30
INCP_M  Z1

LD_EQ_M 21 , Z1
OR      X7
RST_M   Z1
```

Program Example 2

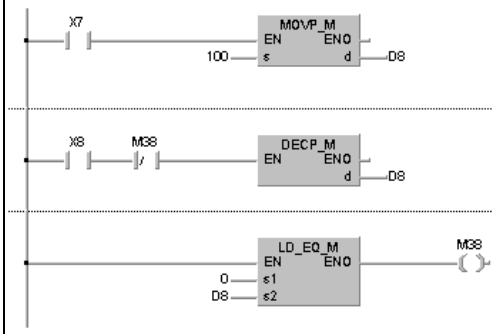
DECP

The following example shows a down counter program. With leading edge from X7, this program stores a value 100 in D8. While M38 is not set, data in D8 is decremented by 1 with leading edge from X8. At D8 = 0, M38 is set.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD    X7
MOVP_M  100 , D8

LD    X8
ANDN  M38
DECP_M  D8

LD_EQ_M  0 , D8
ST    M38
```

6.2.17 DINC, DINCP, DDEC, DDECP

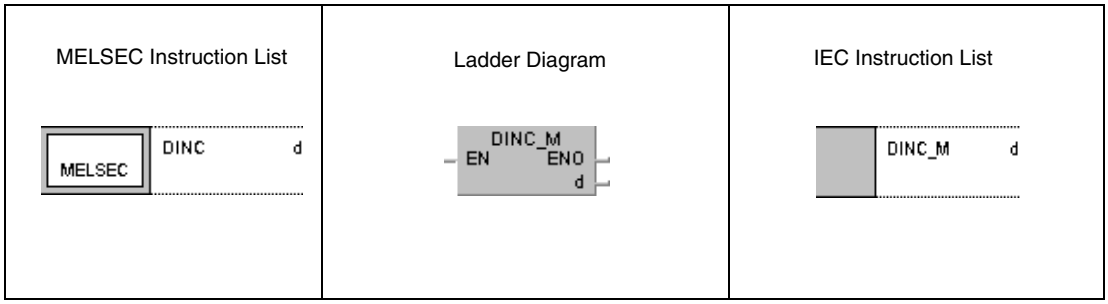
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

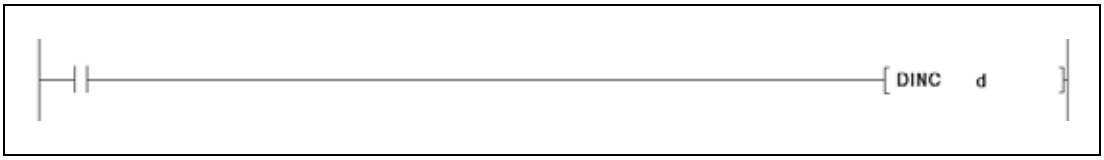
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□N□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				U
d	●	●	●	●	●	●	●	—	—

GX IEC Developer



GX Works2



Variables

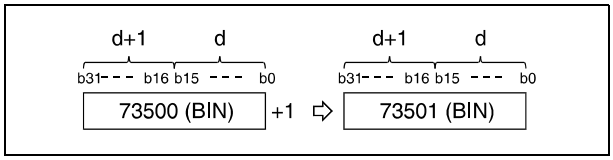
Set Data	Meaning	Data Type
d	First number of device conducted by DINC (add 1) or DDEC (subtract 1) operation.	BIN 32-bit

Functions

BIN 32-bit increment and decrement operations

DINC BIN 32-bit increment

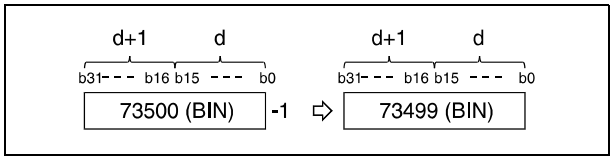
Adds 1 to device designated by d (32-bit).



If the content of d is 2147483647, the result after incrementing is -2147483648.

DDEC BIN 32-bit decrement

Subtracts 1 from device designated by d (16-bit).



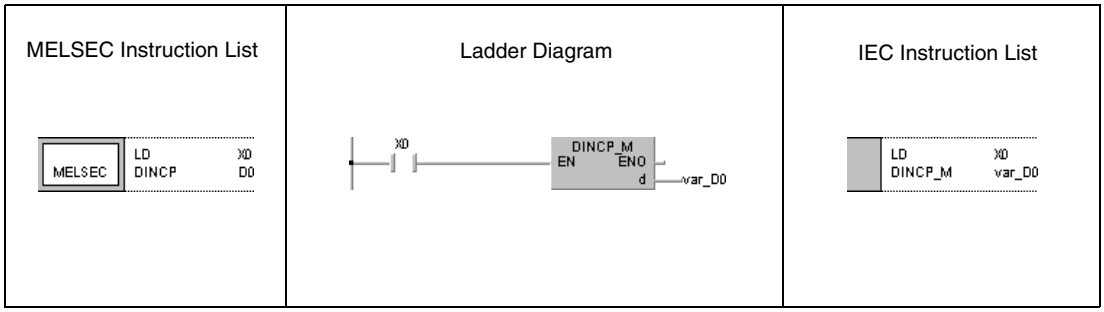
If the content of d is 0, the result after decrementing is -1.

If the content of d is -2147483647, the result after decrementing is 2147483647.

Program Example 1

DINCP

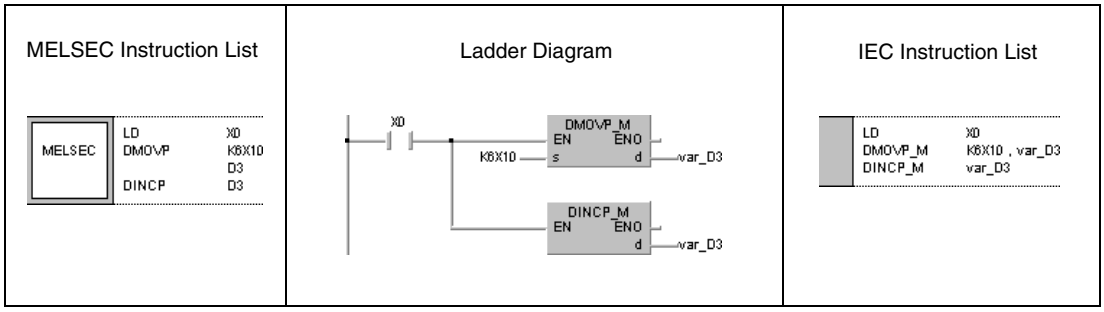
With leading edge from X0, the following program adds 1 to data in D0.



Program Example 2

DINCP

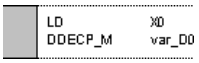
With leading edge from X0, the following program adds 1 to data at X10 through X27. The result is stored in D3 and D4.



Program Example 3

DDECP

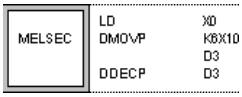
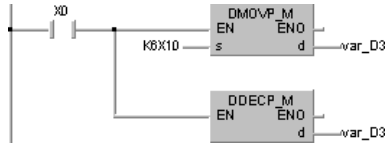
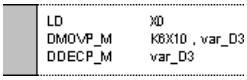
With leading edge from X0, the following program subtracts 1 from data in D0.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

Program Example 4

DDECP

With leading edge from X0, the following program subtracts 1 from data in X10 through X27. The result is stored in D3 and D4.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.3 Data conversion instructions

The instructions described in the following section convert different data types.

NOTE *Within the IEC editors the IEC commands should be used.*

Conversion	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
BIN (16-/32-bit) ⇓ BCD (4-/8-digit)	BCD	BCD_M
	BCDP	BCDP_M
	DBCD	DBCD_M
	DBCDP	DBCDP_M
BCD (4-/8-digit) ⇓ BIN (16-/32-bit)	BIN	BIN_M
	BINP	BINP_M
	DBIN	DBIN_M
	DBINP	DBINP_M
BIN (16-/32-bit) ⇓ Floating point data (Single precision)	FLT	FLT_M
	FLTP	FLTP_M
	DFLT	DFLT_M
	DFLTP	DFLTP_M
BIN (16-/32-bit) ⇓ Floating point data (Double precision)	FLTD	
	FLTPD	
	DFLTD	
	DFLTPD	
Floating point data (Single precision ⇓ BIN (16-/32-bit)	INT	INT_MD
		INT_E_MD
	INTP	INT_P_MD
		INT_P_E_MD
	DINT	DINT_MD
		DINT_E_MD
	DINTP	DINT_P_MD
		DINT_P_E_MD
Floating point data (Double precision) ⇓ BIN (16-/32-bit)	INTD	
	INTPD	
	DINTD	
	DINTPD	
BIN 16-bit ⇓ BIN 32-bit	DBL	DBL_M
	DBLP	DBLP_M
BIN 32-bit ⇓ BIN 16-bit	WORD	WORD_M
	WORDP	WORDP_M
BIN (16-/32-bit) ⇓ GRAY CODE Data	GRY	GRY_M
	GRYP	GRYP_M
	DGRY	DGRY_M
	DGRYP	DGRYP_M
GRAY CODE Data ⇓ BIN (16-/32-bit)	GBIN	GBIN_M
	GBINP	GBINP_M
	DGBIN	DGBIN_M
	DGBINP	DGBINP_M

Conversion	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Sign Reversal BIN (16-/32-bit) (Complement of 2)	NEG	NEG_M
	NEGP	NEGP_M
	DNEG	DNEG_M
	DNEGP	DNEGP_M
Sign Reversal Floating point data	ENEG	ENEG_M
	ENEGP	ENEGP_M
	EDNEG	
	EDNEGP	
BIN Block (16-bit) ⇓ BCD Block (4-digit)	BKBCD	BKBCD_M
	BKBCDP	BKBCDP_M
BCD Block (4-digit) ⇓ BIN Block (16-bit)	BKBIN	BKBIN_M
	BKBINP	BKBINP_M
Floating point data (Single precision) ⇓ Floating point data (Double precision)	ECON	
	ECONP	
Floating point data (Double precision) ⇓ Floating point data (Single precision)	EDCON	
	EDCONP	

6.3.1 BCD, BCDP, DBCD, DBCDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

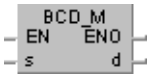
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

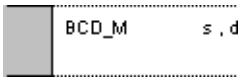
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
s	BIN data, or first number of device storing BIN data.	BIN 16-/32-bit
d	First number of device storing BCD data.	BCD 4-/8-digit

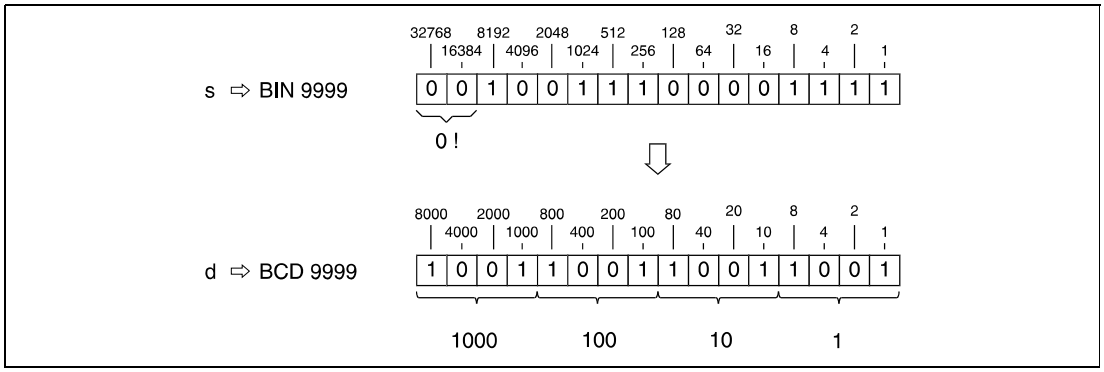
Functions

Conversion from BIN data into BCD data

BCD Conversion from BIN 16-bit data into BCD 4-digit data

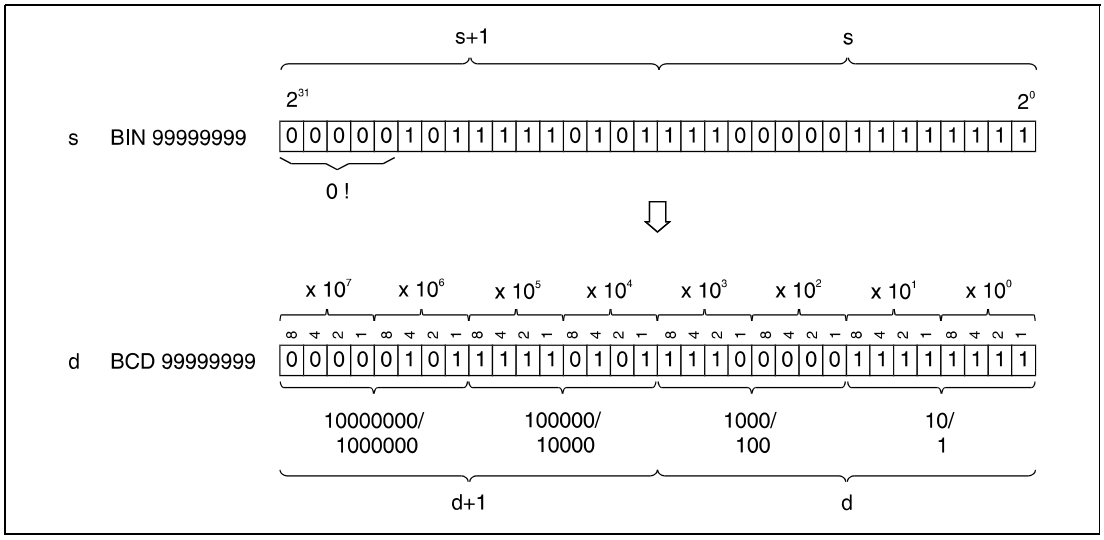
BIN data in s (0 to 9999) is converted into BCD data. The result is stored in d.

The most significant two bits of BIN data in s must be reset (0) when converted into BCD 4-digit data.



DBCD Conversion from BIN 32-bit data into BCD 8-digit data

BIN data in s (0 to 99999999) is converted into BCD data. The result is stored in d. The most significant five bits of BIN data in s must be reset (0) when converted to BCD 8-digit data.



Operation Errors

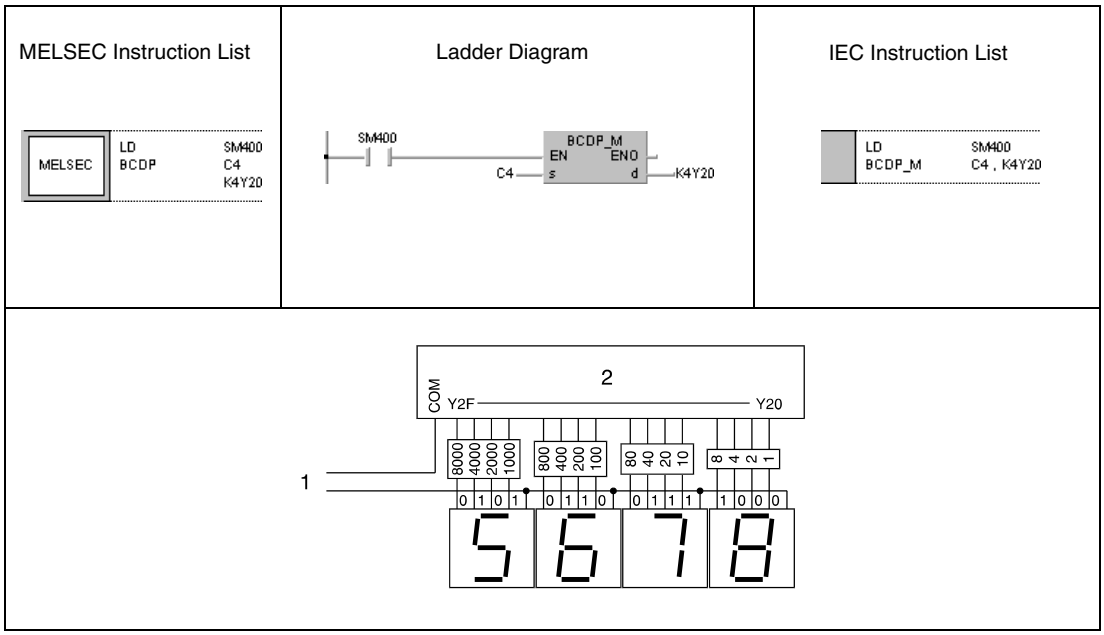
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- BIN 16-bit data in s exceeds the relevant device range of 0 to 9999. (Error code 4100)
- BIN 32-bit data in s+1 or s exceed the relevant device range of 0 to 99999999. (Error code 4100)

Program Example

BCDP

The following program outputs the current value in C4 (5678) to Y20 through Y2F. The output module displays the value on the display unit.



[1] Output power supply

[2] Output module

6.3.2 BIN, BINP, DBIN, DBINP

CPU

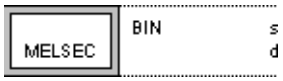
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				U
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

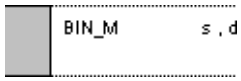
GX IEC Developer

MELSEC Instruction List



Ladder Diagram

IEC Instruction List



GX Works2

Variables

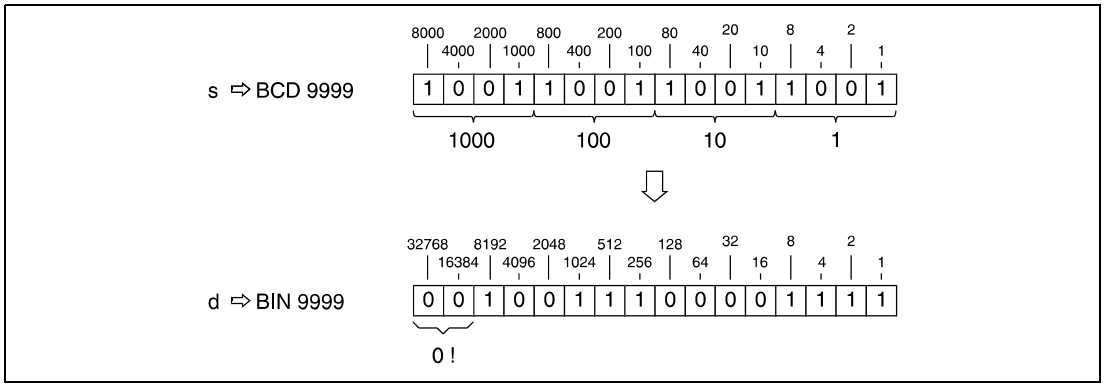
Set Data	Meaning	Data Type
s	BCD data, or first number of device storing BCD data.	BCD 4-/8-digit
d	First number of device storing BIN data.	BIN 16-/32-bit

Functions **Conversion from BCD data into BIN data**

BIN Conversion from BCD 4-digit data into BIN 16-bit data

BCD data in s (0 to 9999) is converted into BIN data. The result is stored in d.

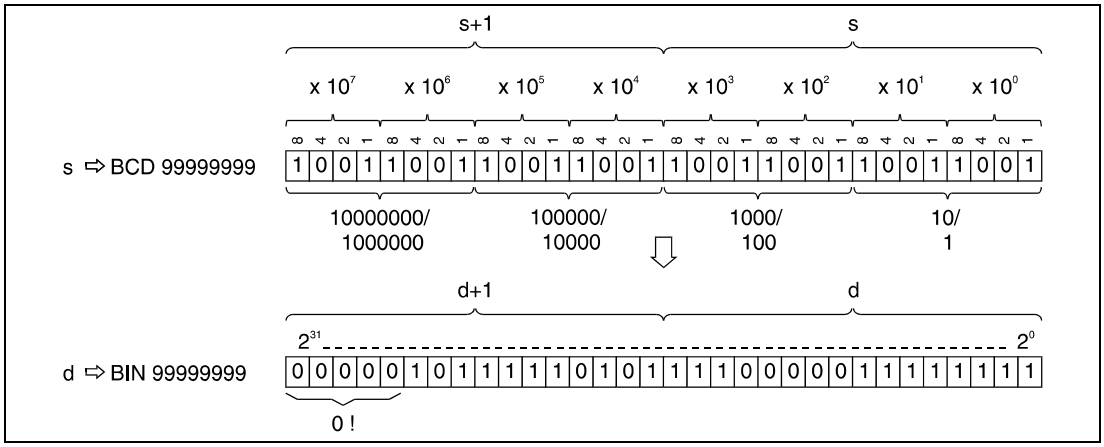
The most significant two bits of BIN data in d must be reset (0) when converted from BCD 4-digit it data.



DBIN Conversion from BCD 8-digit data into BIN 32-bit data

BCD data in s (0 to 99999999) is converted to BIN data. The result is stored in d.

The most significant five bits of BIN data in d must be reset (0) when converting from BCD 8-digit data.



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The individual digits in s do not range within 0 to 9. (Error code 4100)

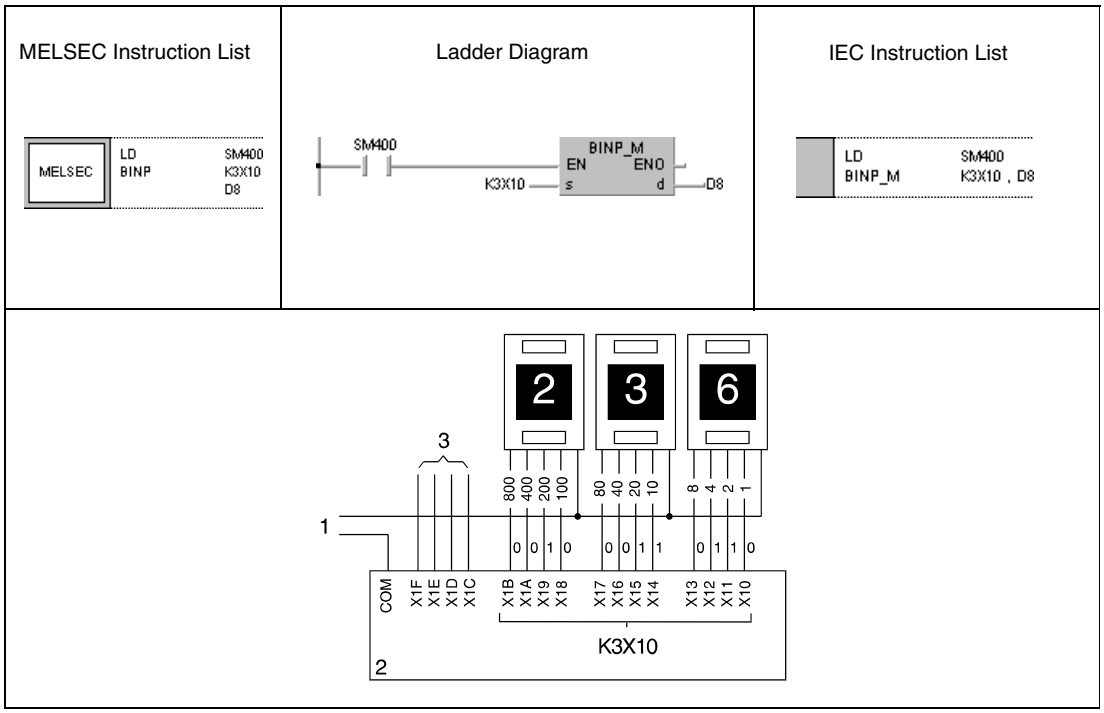
  This error can be suppressed by turning SM722 ON. However, the instruction is not executed regardless of the status of SM722 if the specified value in s is out of range.

  For the BINP/DBINP instruction, the next operation will not be performed until the command (execution condition) is turned from OFF to ON regardless of the presence/absence of an error.

Program Example 1

BINP

The following program converts BCD data in X10 through X1B into BIN data. The result is stored in D8.



[1] Input power supply

[2] Input module

[3] Available inputs

Program Example 2

DBINP

With leading edge from X8, the following program converts BCD data at X10 through X37 into BIN data. The result is stored in D0 through D1.

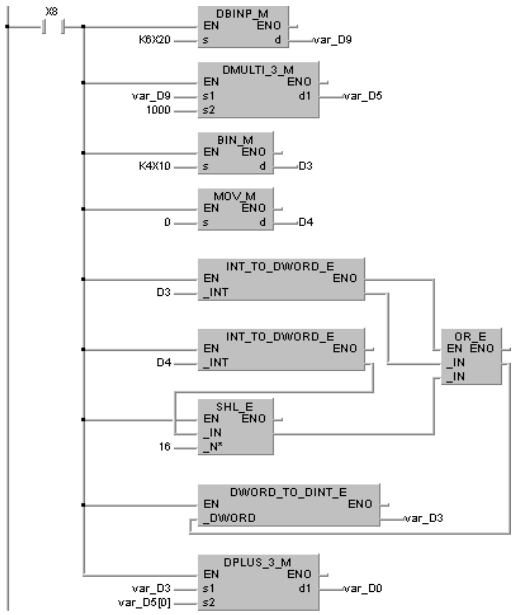
MELSEC Instruction List

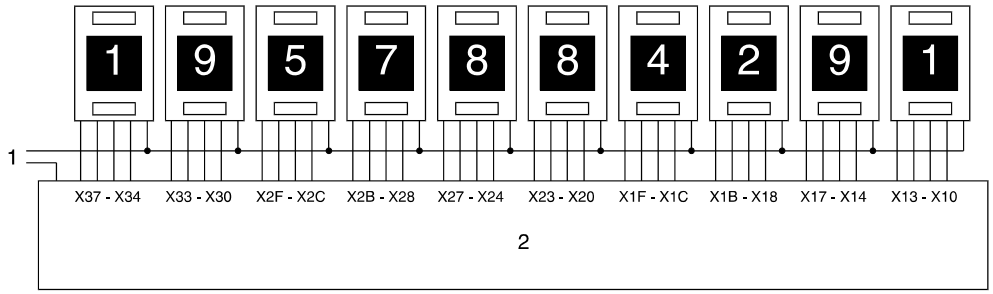
```
MELSEC
        LD      X8
        DBINP   K6X20
                D9
        D*      D9
                K1000
                D5
        BIN     K4X10
                D3
        MOV     K0
                D4
        D+      D3
                D5
                D0
```

IEC Instruction List

```
LD              X8
DBINP_M         K6X20 , Var_D9
DMULTI_3_M      var_D9 , K1000 , Var_D5
BIN_M           K4X10 , D3
MOV_M           K0 , D4
INT_TO_DWORD_E      D4 , var_D3
SHL_E           var_D3 , K16 , var_D3
INT_TO_DWORD_E      D3 , var_temp
LD              var_D3
OR              var_temp
ST              var_D3
LD              X8
DPLUS_3_M       var_D3 , var_D5[0] , var_D0
```

Ladder Diagram





[1] Input power supply

[2] Input module

NOTE

BCD data at X10 through X37 exceeding the relevant device range of 2147483647 cannot be processed by 32-bit devices! In this case the values in D0 and D1 become negative. For further datails see section 3.4 "Programming of variables" in the Programming Manual.

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.3.3 FLT, FLTP, DFLT, DFLTP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

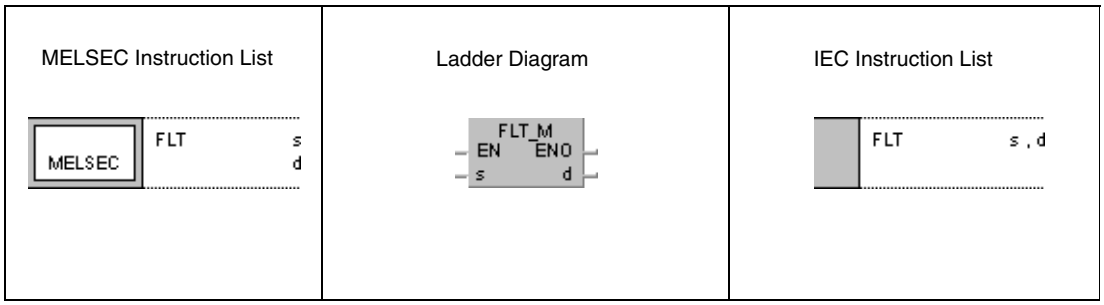
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

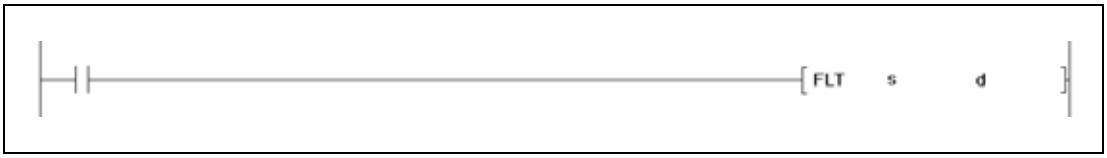
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□N□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Available only in multiple Universal model QCPU and LCPU

GX IEC Developer



GX Works2



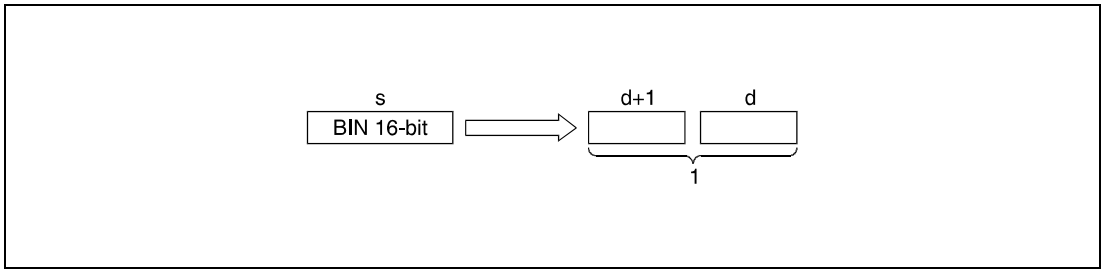
Variables

Set Data	Meaning	Data Type
s	BIN data, or first number of device storing BIN data.	BIN 16-/32-bit
d	First number of device storing floating point data.	Real number

Functions **Conversion from BIN 16-bit/32-bit data into floating point data (Single precision)**

FLT Conversion from BIN 16-bit data into floating point data

BIN 16-bit data in s is converted into 32-bit floating point data. The result is stored in d.

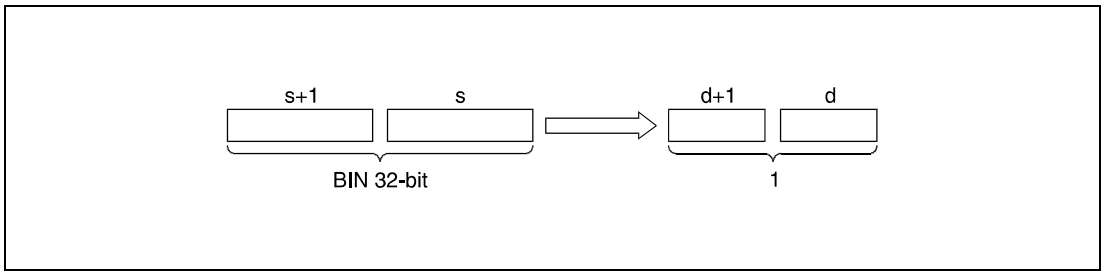


[1] 32-bit floating point data, data type real number

BIN 16-bit data designated by s has to range within -32768 and 32767.

DFLT Conversion from BIN 32-bit data into floating point data

BIN 32-bit data in s is converted into 32-bit floating point data. The result is stored in d.

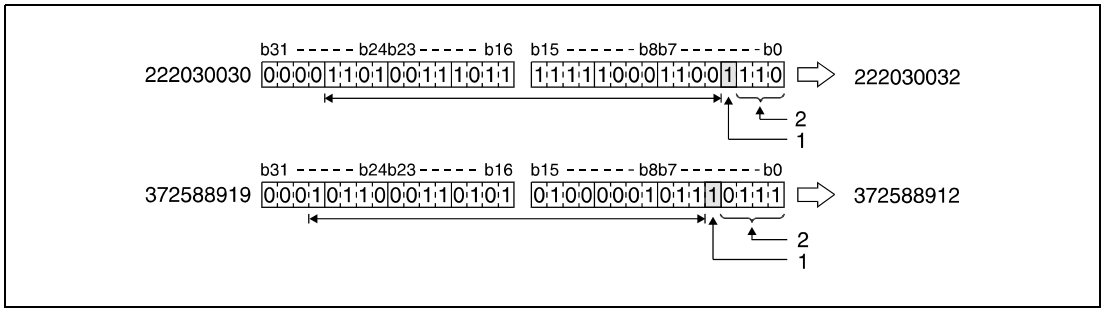


[1] 32-bit floating point data, data type real number

BIN 32-bit data designated by s and s+1 have to range within -2147483648 and 2147483647.

Due to the fact that floating point data (data type real number) is processed by simple 32-bit procedures, the number of significant bits is 24 for a binary display, or approx. 7 digits for a decimal display.

The result of the conversion is rounded off at the 25th bit. All higher bits are eliminated. For this reason, if the resulting integer exceeds a range of -16777216 to 16777215 (BIN 24-bit value), errors may occur in the conversion.



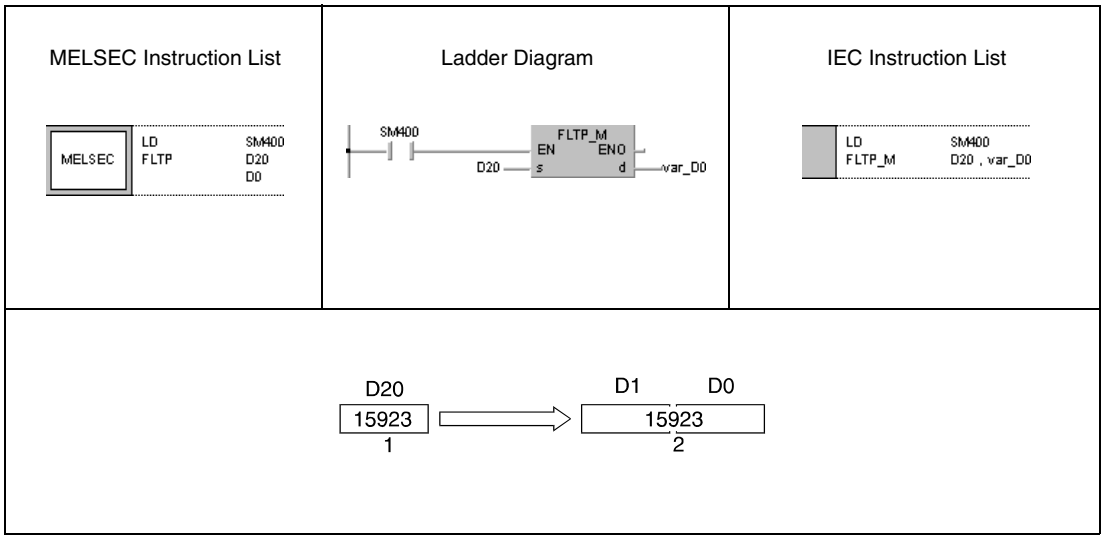
[1] Rounded off

[2] Eliminated

Program Example 1

FLTP

The following program converts BIN 16-bit data in D20 into 32-bit floating point data. The result is stored in D0 and D1.



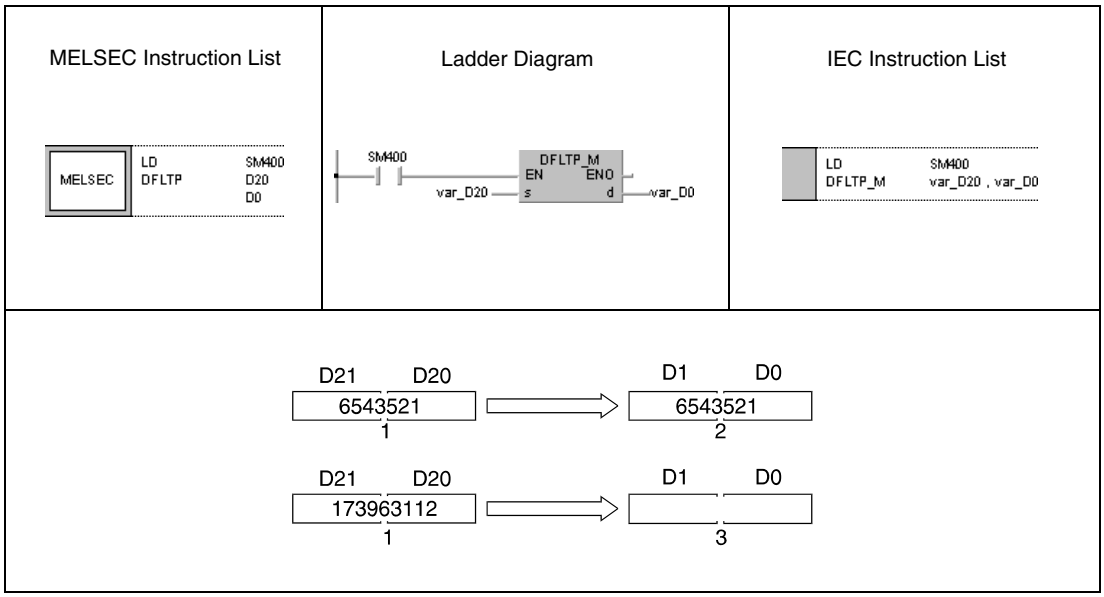
[1] BIN 16-bit data

[2] 32-bit floating point data, data type real number

Program Example 2

DFLTP

The following program converts BIN 32-bit data in D20 and D21 into 32-bit floating point data. The result is stored in D0 and D1.



[1] BIN 32-bit data

[2] 32-bit floating point data, data type real number

[3] Conversion error, because there are 7 significant digits

NOTE

These programs will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.3.4 FLTD, FLTPD, DFLTD, DFLTPD

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

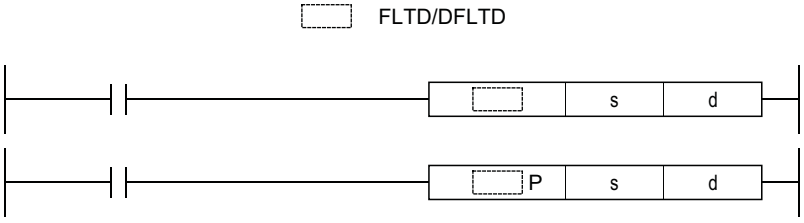
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	●	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

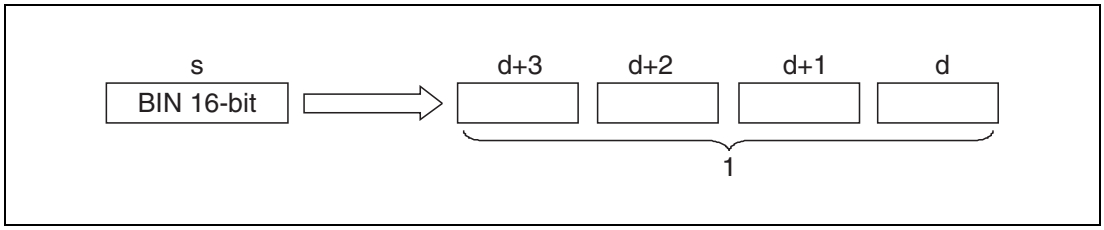
Set Data	Meaning	Data Type
s	BIN data, or first number of device storing BIN data.	BIN 16-/32-bit
d	First number of device storing floating point data.	Real number

Functions

Conversion from BIN 16-bit/32-bit data into floating point data (Double precision)

FLTD Conversion from BIN 16-bit data into floating point data

BIN 16-bit data in s is converted into 64-bit floating point data. The result is stored in d.

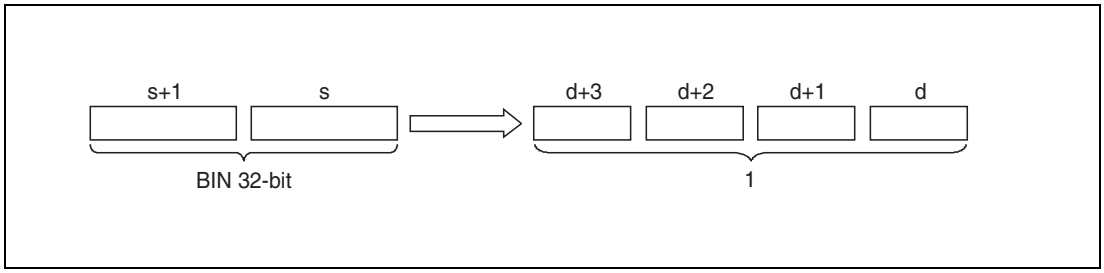


[1] 64-bit floating point data, data type real number

BIN 16-bit data designated by s has to range within -32768 and 32767.

DFLTD Conversion from BIN 32-bit data into floating point data

BIN 32-bit data in s is converted into 64-bit floating point data. The result is stored in d.

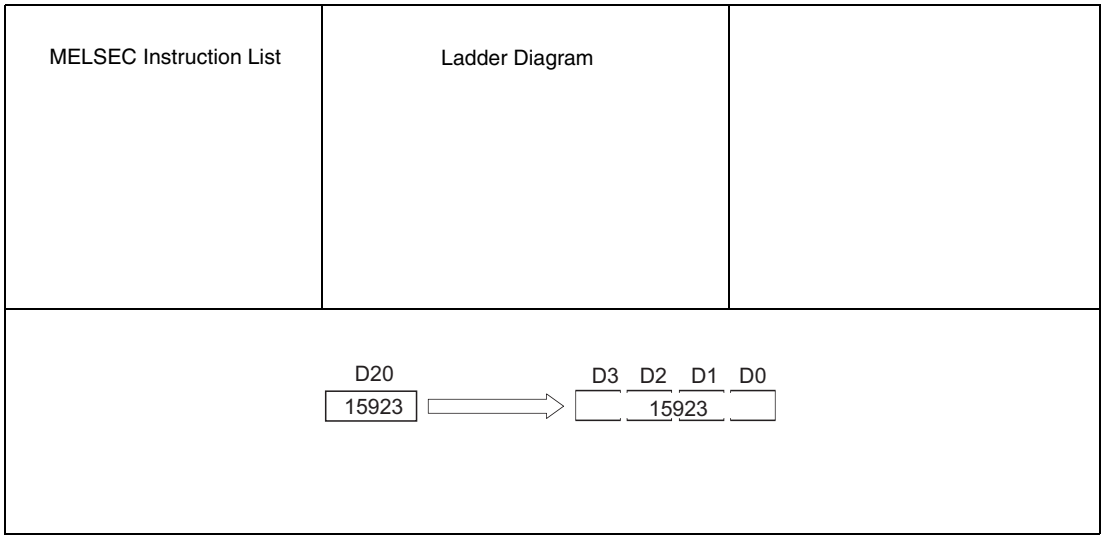


[1] 64-bit floating point data, data type real number

Program Example 1

FLTDP

The following program converts BIN 16-bit data in D20 into 64-bit floating point data. The result is stored in D0 to D3.



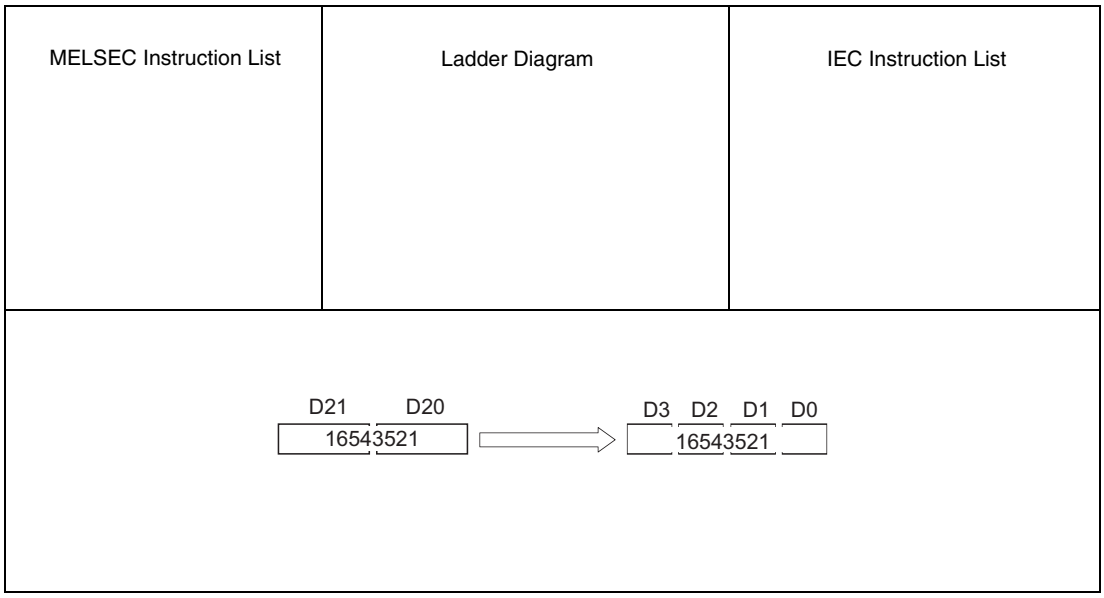
[1] BIN 16-bit data

[2] 64-bit floating point data, data type real number

Program Example 2

DFLTDP

The following program converts BIN 32-bit data in D20 and D21 into 64-bit floating point data. The result is stored in D0 to D3.



[1] BIN 32-bit data

[2] 64-bit floating point data, data type real number

NOTE *These programs will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.3.5 INT, INTP, DINT, DINTP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

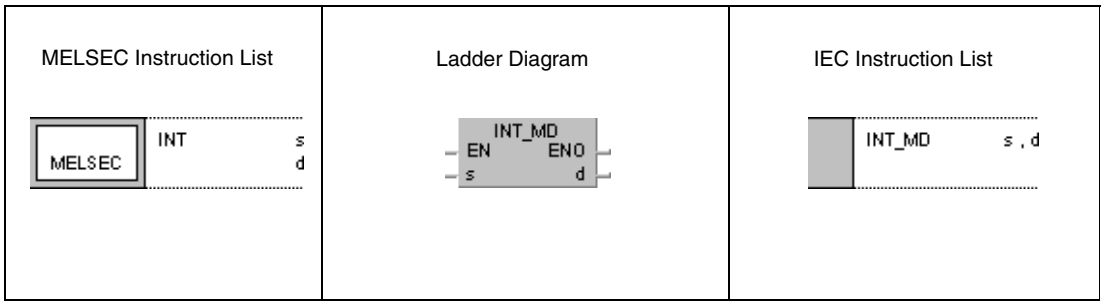
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

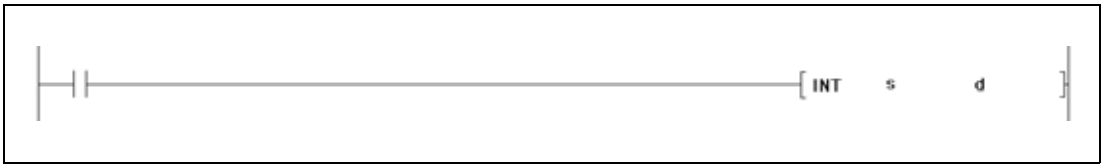
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	●	●	●	●	●	●	●	—	—

[1] Available only in multiple Universal model QCPU and LCPU

GX IEC Developer



GX Works2



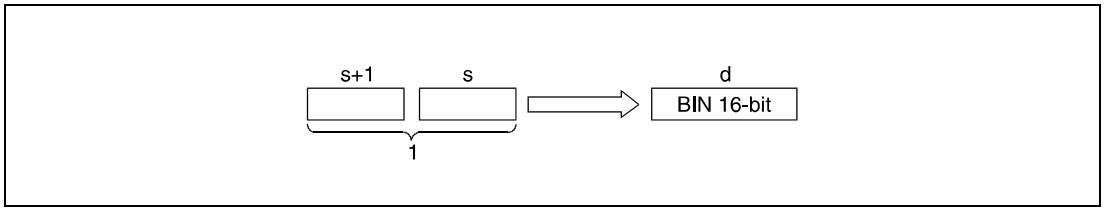
Variables

Set Data	Meaning	Data Type
s	Floating point data, or first number of device storing floating point data.	Real number
d	First number of device storing BIN data.	BIN 16-/32-bit

Functions **Conversion from floating point data into BIN 16-bit/32-bit data (Single precision)**

INT Conversion from 32-bit floating point data into BIN 16-bit data

32-bit floating point data in s is converted into BIN 16-bit data. The result is stored in d.



[1] 32-bit floating point data, data type real number

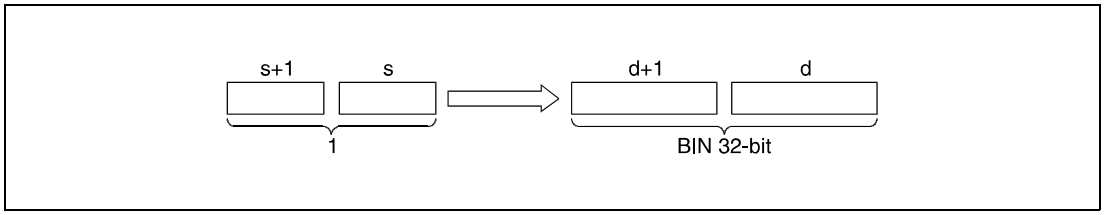
Floating point data in s and s+1 have to range within -32768 and 32767.

The converted integer value is stored as BIN 16-bit data.

The converted integer value is rounded off at the first digit after the decimal point.

DINT Conversion from 32-bit floating point data into BIN 32-bit data

32-bit floating point data in s is converted to BIN 32-bit data. The result is stored in d.



[1] 32-bit floating point data, data type real number

Floating point data in s and s+1 have to range within -2147483648 and 2147483647.

The converted integer value is stored as BIN 32-bit data.

The converted integer value is rounded off at the first digit after the decimal point.

Operation Errors

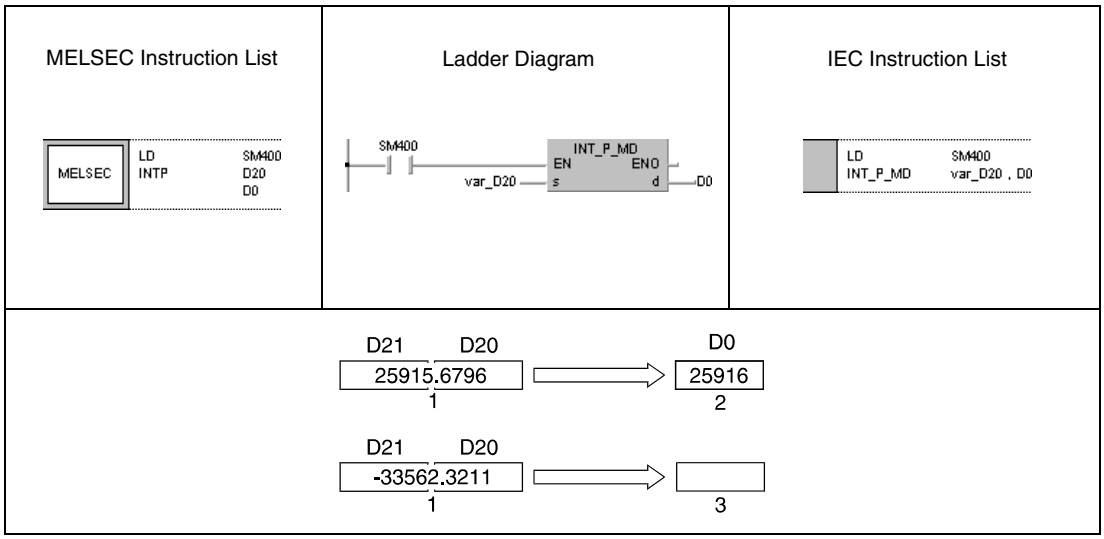
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The contents of the designated device or the result of the addition are not zero and not within the following range:
  $\pm 2^{-126} \leq$ (Contents of designated device) $< \pm 2^{128}$
  (For the Universal model QCPU, LCPU) (Error code 4140)
- The value of the specified device is –0, unnormalized number, nonnumeric, and $\pm\infty$.
  (For the Universal model QCPU, LCPU) (Error code 4140)
- Performing an INT instruction, floating point data designated by s exceeds the relevant device range of -32768 to 32767.
- Performing a DINT instruction, floating point data designated by s exceeds the relevant device range of -2147483648 to 2147483647.

Program Example 1

INTP

The following program converts 32-bit floating point data in D20 and D21 into BIN 16-bit data. The result is stored in D0.



[1] 32-bit floating point data, data type real number

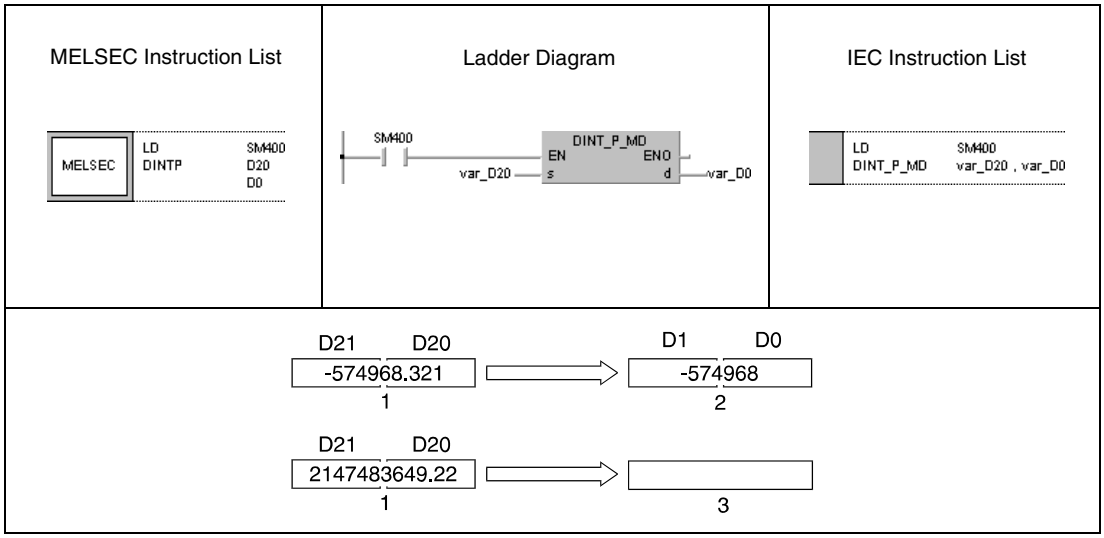
[2] BIN 16-bit data

[3] No result. Value exceeds relevant device range of INT instruction. Error code is returned.

Program Example 2

DINTP

The following program converts 32-bit floating point data in D20 and D21 into BIN 32-bit data. The result is stored in D0 and D1.



[1] 32-bit floating point data, data type real number

[2] BIN 32-bit data

[3] No result. Value exceeds relevant device range of DINT instruction. Error code is returned.

NOTE *These programs will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.3.6 INTD, INTPD, DINTD, DINTPD

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

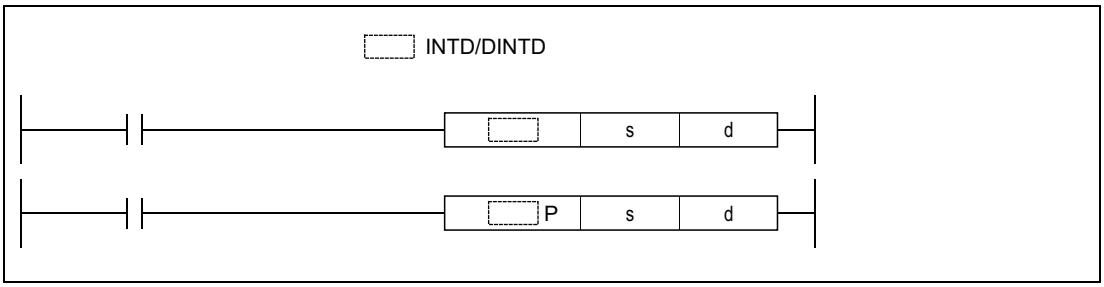
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	●	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

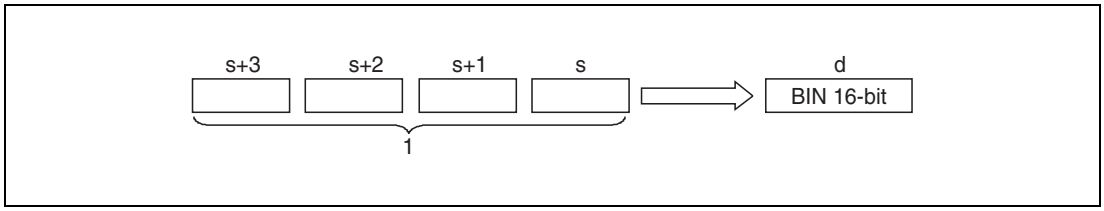
Set Data	Meaning	Data Type
s	Floating point data, or first number of device storing floating point data.	Real number
d	First number of device storing BIN data.	BIN 16-/32-bit

Functions

Conversion from floating point data into BIN data (Double precision)

INTD Conversion from 64-bit floating point data into BIN 16-bit data

64-bit floating point data in s is converted into BIN 16-bit data. The result is stored in d.



[1] 64-bit floating point data, data type real number

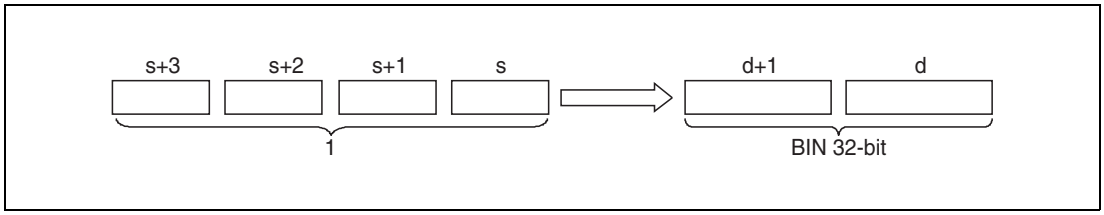
Floating point data in s+3, s+2, s+1 and s have to range within -32768 and 32767.

The converted integer value is stored as BIN 16-bit data.

The converted integer value is rounded off at the first digit after the decimal point.

DINTD Conversion from 64-bit floating point data into BIN 32-bit data

64-bit floating point data in s is converted to BIN 32-bit data. The result is stored in d.



[1] 64-bit floating point data, data type real number

Floating point data in s+3, s+2, s+1 and s have to range within -2147483648 and 2147483647.

The converted integer value is stored as BIN 32-bit data.

The converted integer value is rounded off at the first digit after the decimal point.

Operation Errors

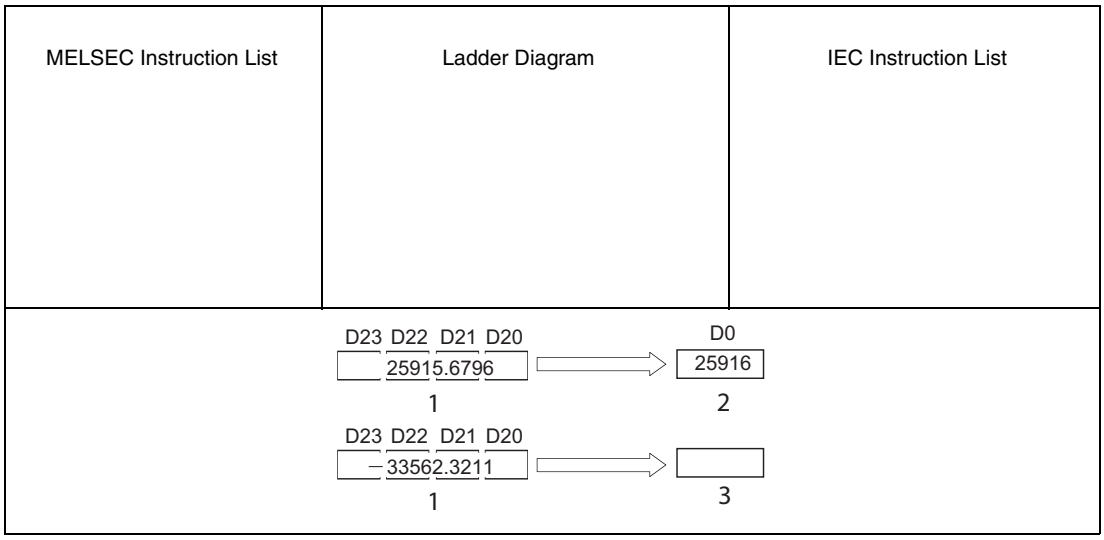
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the designated device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Contents of designated device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0. (Error code 4140)
- Performing an INTD instruction, floating point data designated by s exceeds the relevant device range of -32768 to 32767. (Error code 4100)
- Performing a DINTD instruction, floating point data designated by s exceeds the relevant device range of -2147483648 to 2147483647. (Error code 4100)

Program Example 1

INTDP

The following program converts 64-bit floating point data in D20 to D23 into BIN 16-bit data. The result is stored in D0.



[1] 64-bit floating point data, data type real number

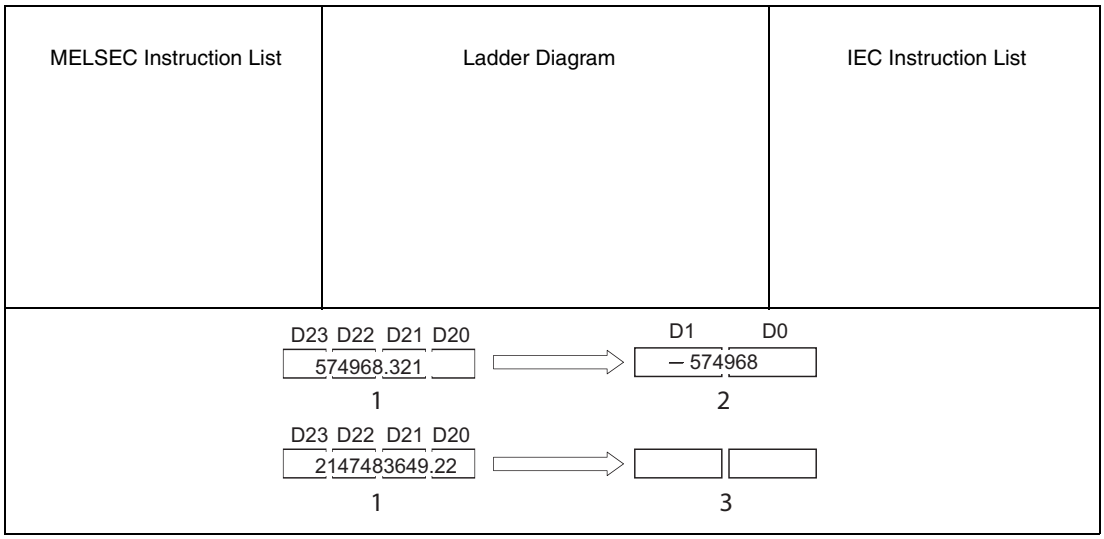
[2] BIN 16-bit data

[3] No result. Value exceeds relevant device range of INTD instruction. Error code is returned.

Program Example 2

DINTDP

The following program converts 64-bit floating point data in D20 to D23 into BIN 32-bit data. The result is stored in D0 and D1.



[1] 64-bit floating point data, data type real number

[2] BIN 32-bit data

[3] No result. Value exceeds relevant device range of DINTD instruction. Error code is returned.

NOTE

These programs will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.3.7 DBL, DBLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

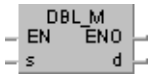
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

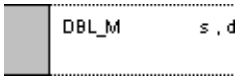
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

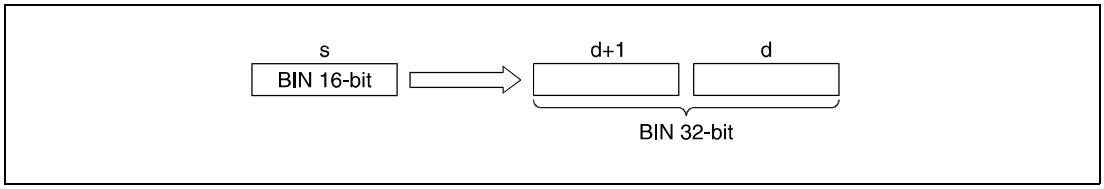
Set Data	Meaning	Data Type
s	First number of device storing data to be converted.	BIN 16-bit
d	First number of device storing converted data.	BIN 32-bit

Functions

Conversion from BIN 16-bit data into BIN 32-bit data

DBL Conversion from BIN 16-bit data into BIN 32-bit data

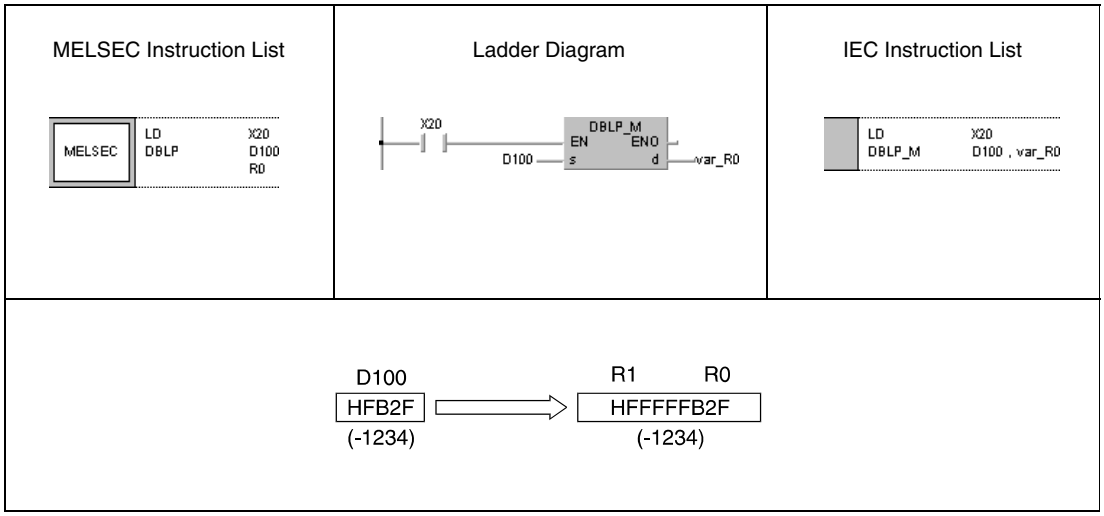
BIN 16-bit data in s is converted into BIN 32-bit data with sign. The result is stored in d.



Program Example

DBLP

With leading edge from X20, the following program converts BIN 16-bit data in D100 into BIN 32-bit data. The result ist stored in R0 and R1.



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.3.8 WORD, WORDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

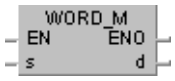
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

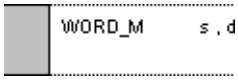
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

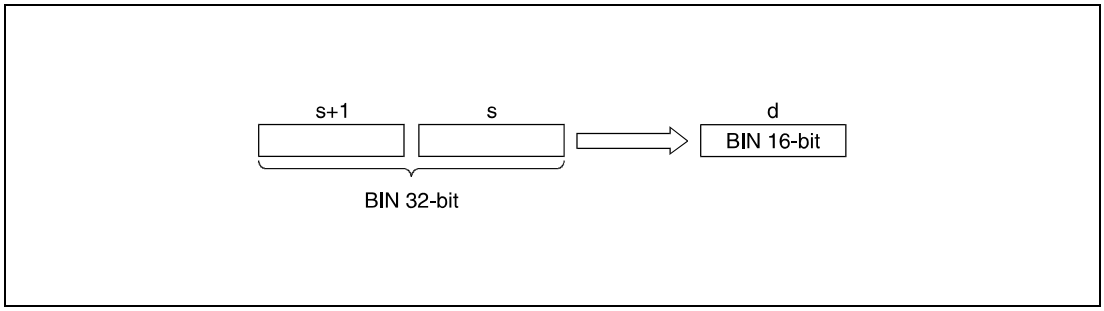
Set Data	Meaning	Data Type
s	First number of device storing data to be converted.	BIN 32-bit
d	First number of device storing converted data.	BIN 16-bit

Functions

Conversion from BIN 32-bit data into BIN 16-bit data

WORD Conversion from BIN 32-bit data into BIN 16-bit data

BIN 32-bit data in s is converted into BIN 16-bit data. The result is stored in d.



Operation Errors

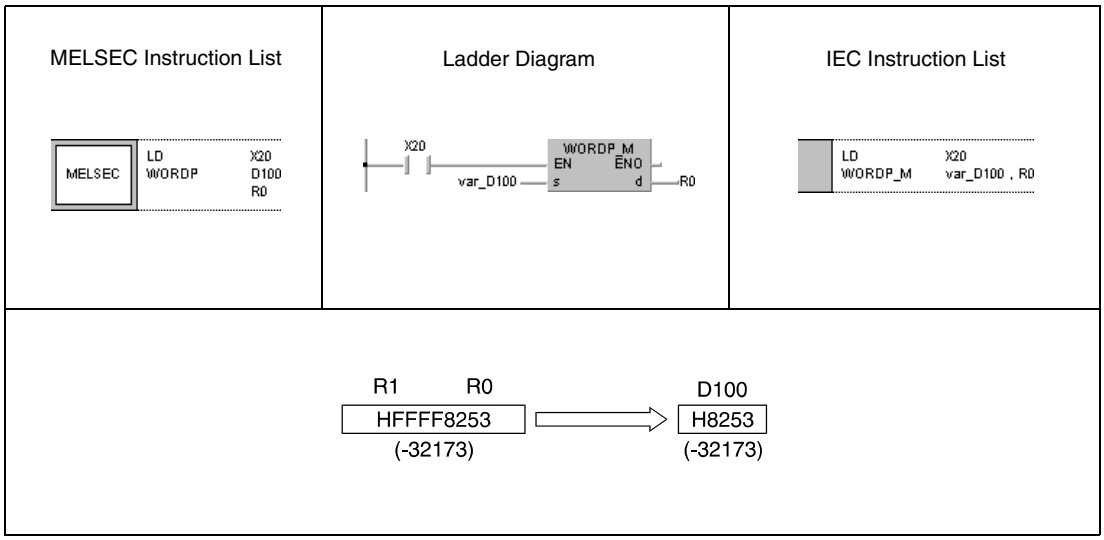
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The BIN data designated by s and s+1 exceed the relevant device range of -32768 to 32767. (Error code 4100)

Program Example

WORDP

With leading edge from X20, the following program converts BIN 32-bit data in D100 and D101 into BIN 16-bit data. The result is stored in R0.



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.3.9 GRY, GRYP, DGRY, DGRYP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

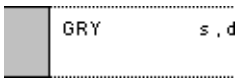
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer

MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2

Variables

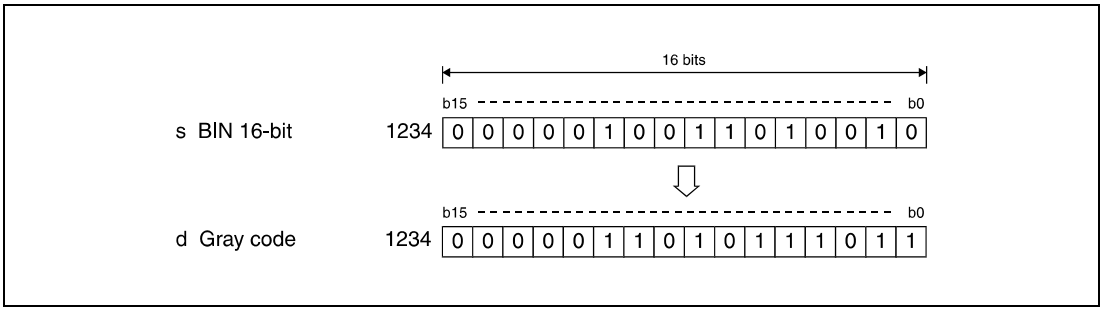
Set Data	Meaning	Data Type
s	BIN data, or first number of device storing BIN data.	BIN 16-/32-bit
d	First number of device storing converted Gray code data.	Gray code data 16-/32-bit

Functions

Conversion from BIN data into Gray code data

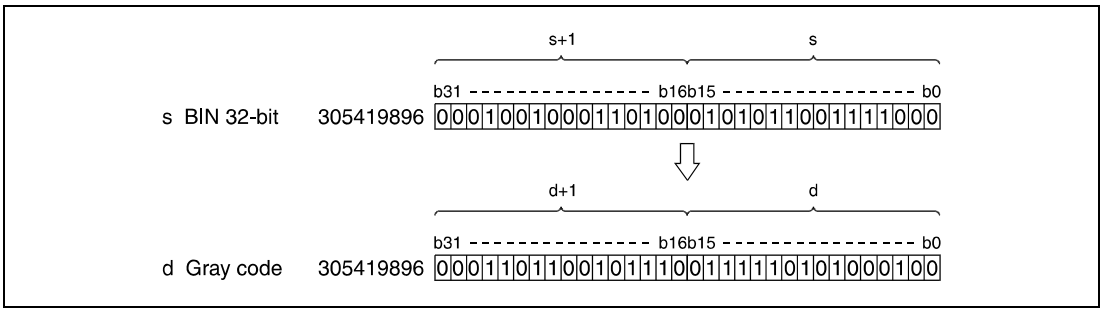
GRY Conversion from BIN 16-bit data into Gray code data

BIN 16-bit data in s is converted into Gray code data. The result is stored in d.



DGRY Conversion from BIN 32-bit data into Gray code data

BIN 32-bit data in s is converted into Gray code data. The result is stored in d.



Operation Errors

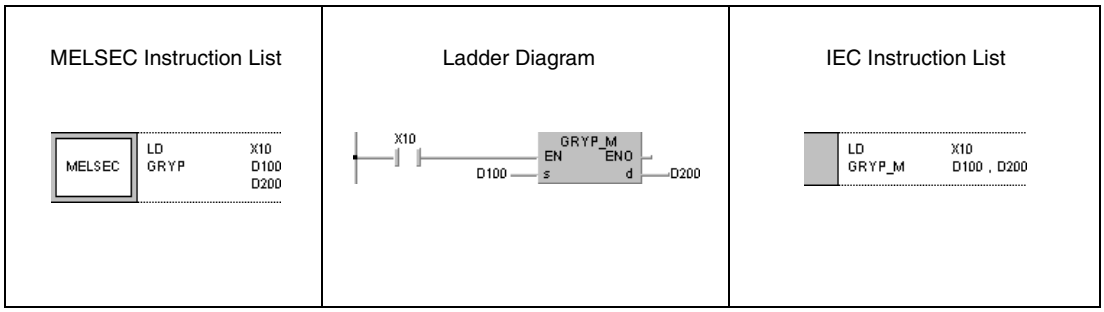
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- Data in s is negative. (Error code 4100)

Program Example 1

GRYP

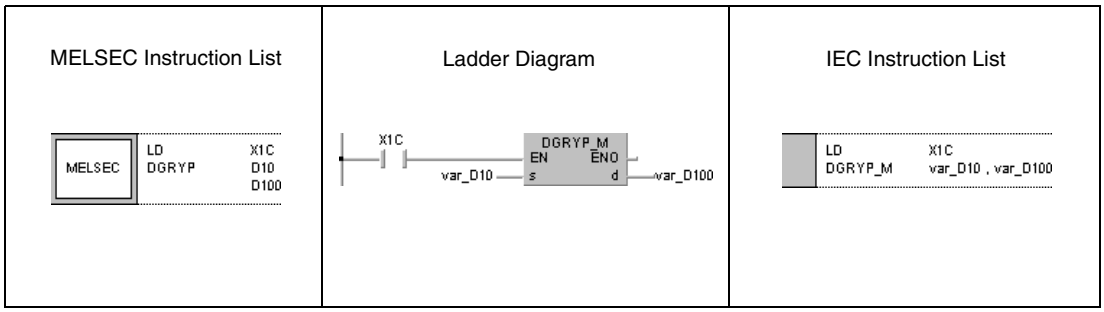
With leading edge from X10, the following program converts BIN 16-bit data in D100 into Gray code data. The result is stored in D200.



Program Example 2

DGRYP

With leading edge from X1C, the following program converts BIN 32-bit data in D10 and D11 into Gray code data. The result is stored in D100 and D101.



NOTE *The program example 2 will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.3.10 GBIN, GBINP, DGBIN, DGBINP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

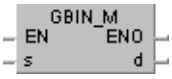
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

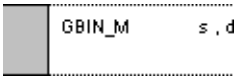
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

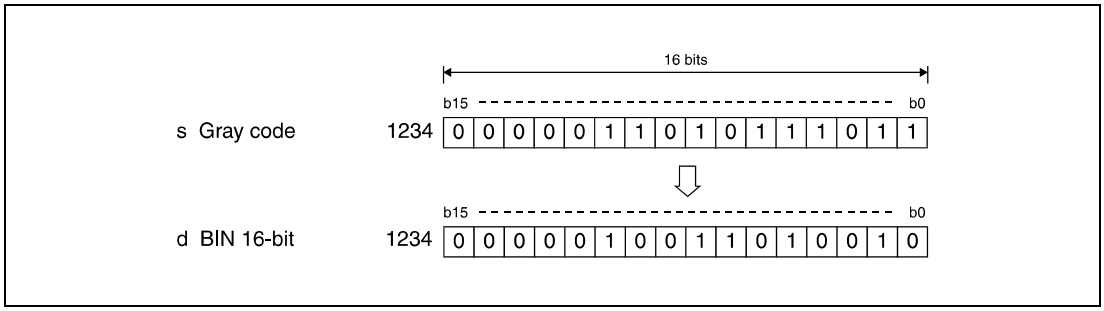
Variables

Set Data	Meaning	Data Type
s	Gray code data, or first number of device storing Gray code data.	Gray code data 16-/32-bit
d	First number of device storing converted BIN data.	BIN 16-/32-bit

Functions **Conversion from Gray code data into BIN data**

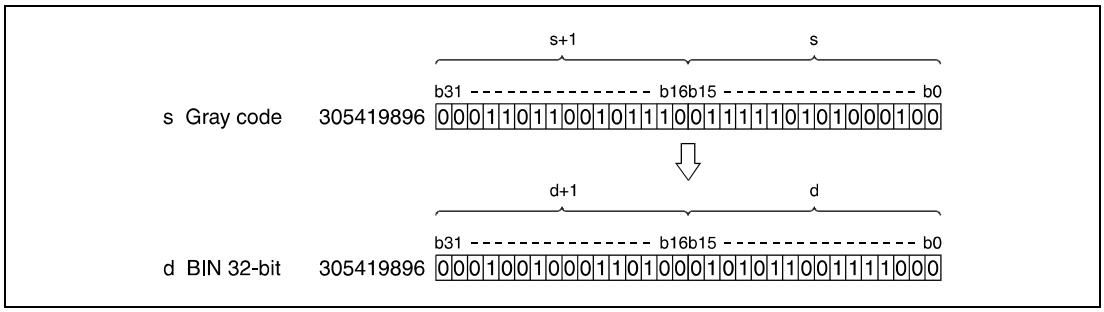
GBIN Conversion from Gray code data into BIN 16-bit data

Gray code data in s is converted into BIN 16-bit data. The result is stored in d.



DGBIN Conversion from Gray code data into BIN 32-bit data

Gray code data in s is converted into BIN 32-bit data. The result is stored in d.



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

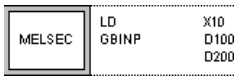
- Performing a GBIN instruction, data in s exceeds the relevant device range of 0 to 32767. (Error code 4100)
- Performing a DGBIN instruction, data in s exceeds the relevant device range of 0 to 2147483647. (Error code 4100)

Program Example 1

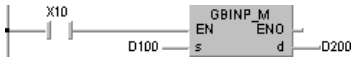
GBINP

With leading edge from X10, the following program converts Gray code data in D100 into BIN 16-bit data. The result is stored in D200.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

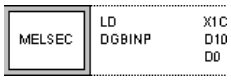
```
LD    X10
GBINP_M  D100 , D200
```

Program Example 2

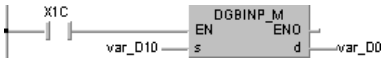
DGBINP

With leading edge from X1C, the following program converts Gray code data in D10 and D11 into BIN 32-bit data. The result is stored in D0 and D1.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD    X1C
DGBINP_M  var_D10 , var_D0
```

NOTE *The program example 2 will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.3.11 NEG, NEGP, DNEG, DNEGP

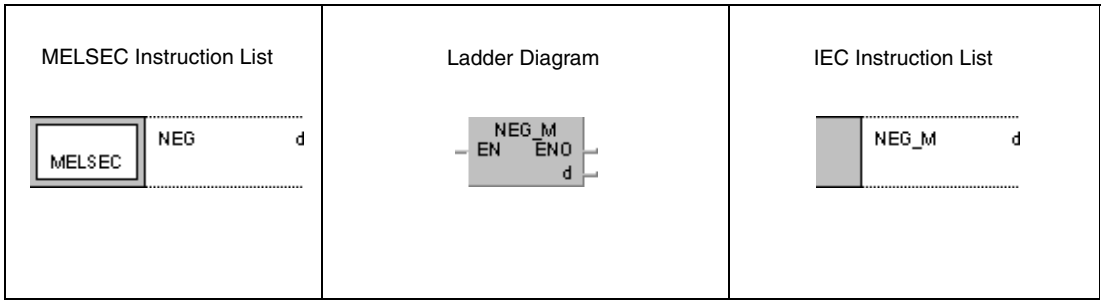
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

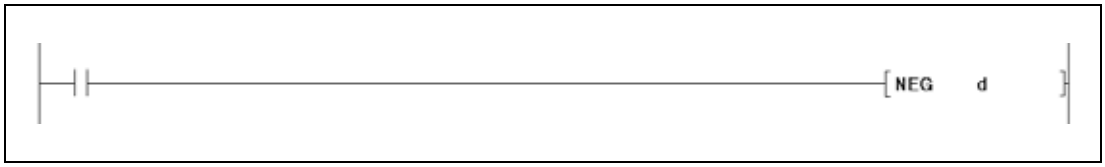
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	●	●	●	●	●	●	●	—	—

GX IEC Developer



GX Works2



Variables

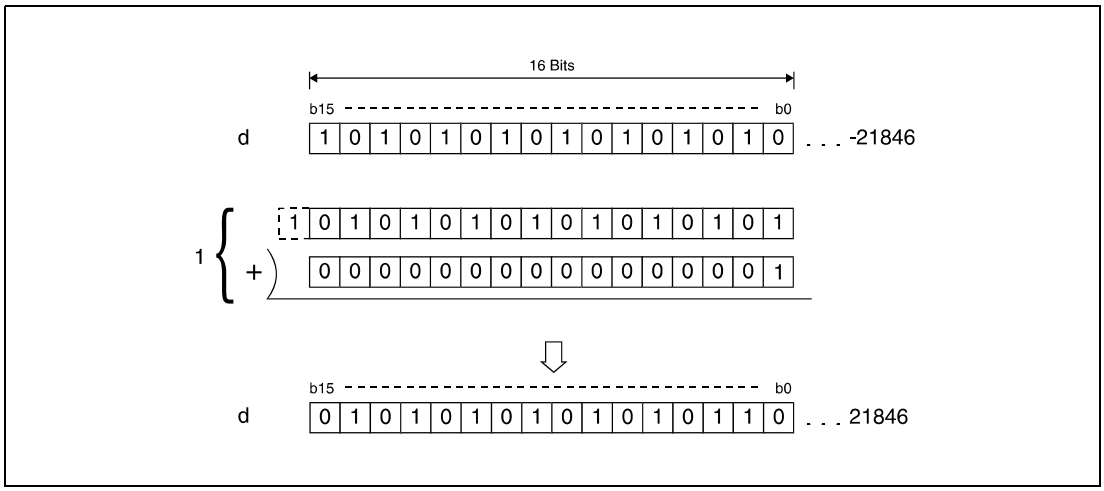
Set Data	Meaning	Data Type
d	First number of device storing data for the sign reversal.	BIN 16-/32-bit

Functions

Complement of 2 of BIN 16- and 32-bit data (sign reversal)

NEG Negation of BIN 16-bit data

The NEG instruction (complement of 2) reverses the sign of BIN 16-bit data. BIN 16-bit data in d is inverted first and then the value "1" is added. The result is stored in d.

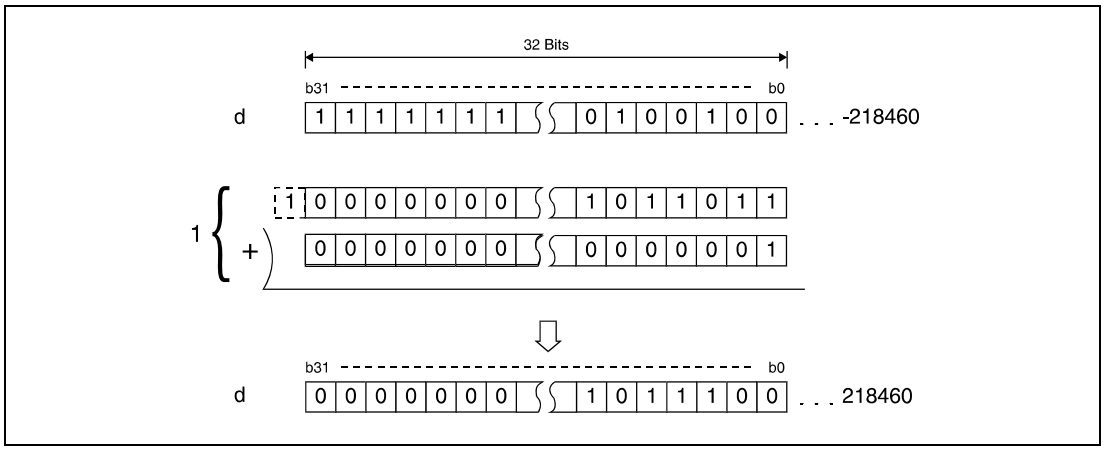


[1] Inversion with following addition

The function of this instruction is to change a negative sign into a positive one, or to change a positive sign into a negative one.

DNEG Negation of BIN 32-bit data

The DNEG instruction (complement of 2) reverses the sign of BIN 32-bit data. BIN 32-bit data in d is inverted first and then the value "1" is added. The result is stored in d.



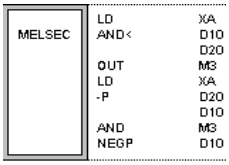
[1] Inversion with following addition

Program Example

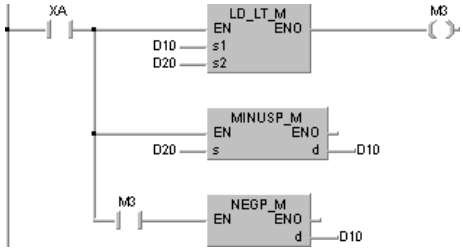
NEGP

With leading edge from XA, the following program subtracts data in D10 from data in D20. M3 is set, if D10 is less than D20. If M3 is set, the result in D10 is the absolute value (complement of 2) and becomes positive.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD          XA
AND_LT_M    D10 , D20
ST          M3
LD          XA
MINUSP_M    D20 , D10
AND         M3
NEGP_M      D10
```

6.3.12 ENEG, ENEGP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

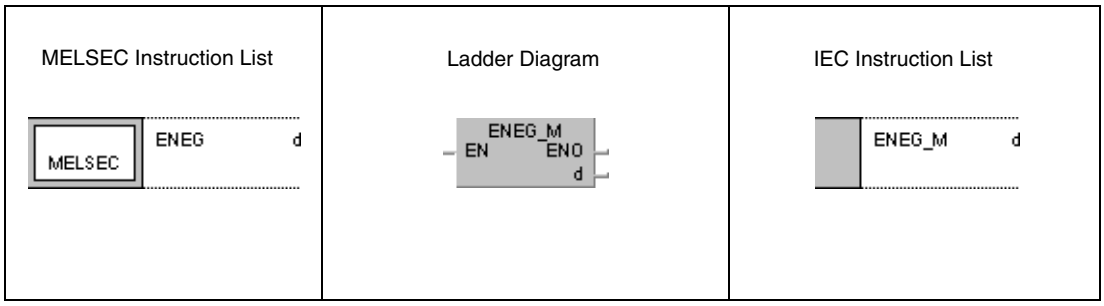
[1] Basic model QCPU: The serial number (upper five digits) is "04122" or higher.

Devices

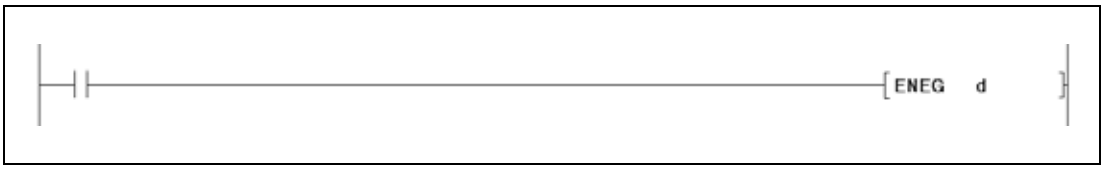
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□N□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	—	●	●	—	●	●	●[1)]	—	—

[1] Available only in multiple Universal model QCPU and LCPU

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
d	First number of device storing floating point data for the sign reversal.	Real number

Functions

Sign reversal for floating point data (Single precision)

ENEG Negation of 32-bit floating point data

These instructions negate 32-bit floating point data in d. The result is stored in d.

The function of these instructions is to change a negative sign into a positive one, or a positive sign into a negative one.

Operation Errors

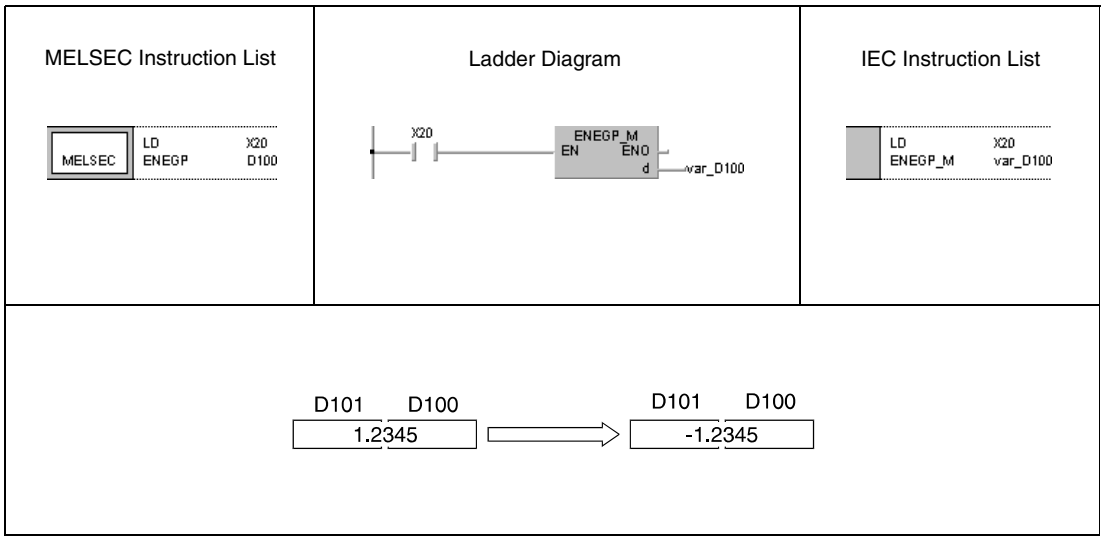
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The contents of the designated device or the result of the addition are not zero and not within the following range:
  $\pm 2^{-126} \leq$ (Contents of designated device) $< \pm 2^{128}$
  (For the Universal model QCPU, LCPU) (Error code 4140)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU) (Error code 4140)

Program Example

ENEGP

With leading edge from X20, the following program negates floating point data in D100 and D101. The result is stored in D100 and D101.



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.3.13 EDNEG, EDNEGP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

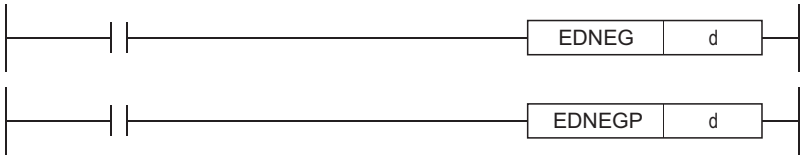
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
d	First number of device storing floating point data for the sign reversal.	Real number

Functions

Sign reversal for floating point data (Double precision)

EDNEG Negation of 64-bit floating point data

These instructions negate 64-bit floating point data in d. The result is stored in d.

The function of these instructions is to change a negative sign into a positive one, or a positive sign into a negative one.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

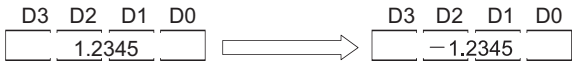
- The contents of the designated device or the result of the addition are not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Contents of designated device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)

Program Example

EDNEGP

With leading edge from X20, the following program negates 64-bit floating point data in D0 to D3. The result is stored in D0 to D3.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.3.14 BKBCD, BKBCDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

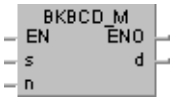
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

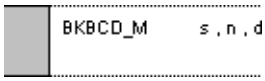
GX IEC Developer

MELSEC Instruction List

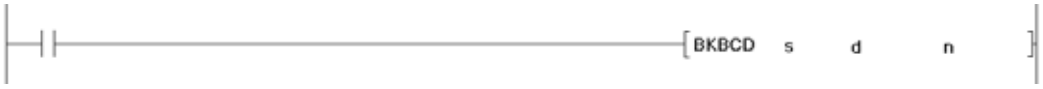
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	First number of device storing BIN data to be converted.	BIN 16-bit
d	First number of device storing converted BCD data.	BCD 4-digit
n	Number of data blocks to be converted.	BIN 16-bit

Functions

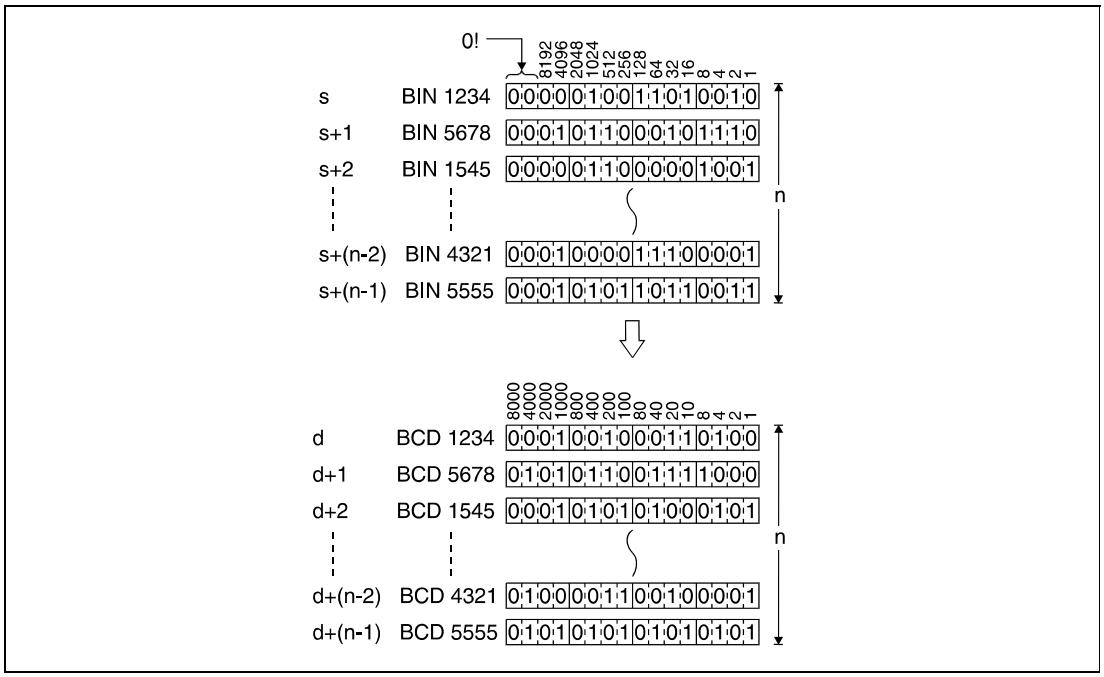
Conversion from BIN block data into BCD block data

BKBCD Conversion from BIN 16-bit block data into BCD 4-digit block data

This instruction converts each nth BIN 16-bit block in s into the nth BCD 4-digit block. Converted data is stored in d.

BIN 16-bit block data in s has to range within 0 and 9999.

The most significant two bits of the BIN 16-bit data blocks in s must be reset (0).



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of data blocks determined by n exceeds the storage device numbers designated by s and d. (Error code 4101)
- BIN block data in s exceeds the relevant device range of 0 to 9999. (Error code 4100)
- The storage device numbers designated by s and d overlap. (Error code 4101)

For details on index qualification refer to section 3.6 of this manual.

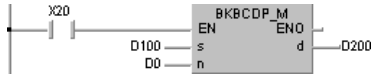
Program Example

BKBCDP

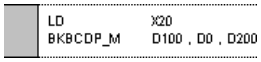
With leading edge from X20, the following program converts BIN 16-bit block data in D100 into BCD 4-digit block data. Converted data is stored in D200. The number of data blocks (3) converted is stored in D0.

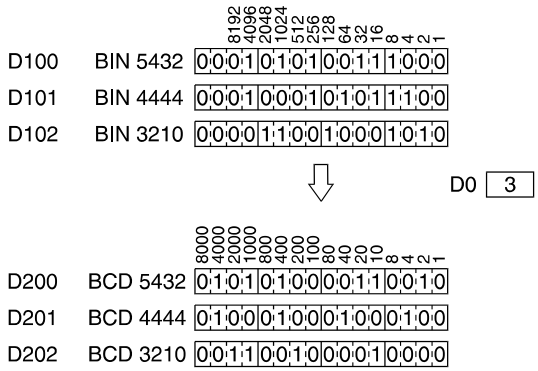
MELSEC Instruction List

Ladder Diagram



IEC Instruction List





6.3.15 BKBIN, BKBINP

CPU

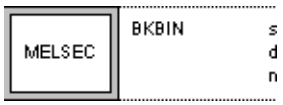
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

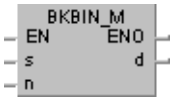
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

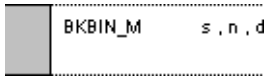
MELSEC Instruction List



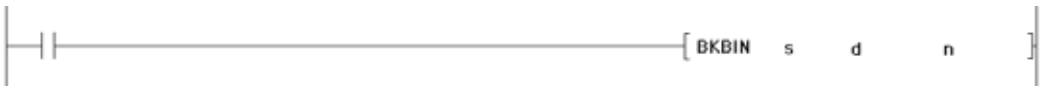
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	First number of device storing BCD data to be converted.	BCD 4-digit
d	First number of device storing converted BIN data.	BIN 16-bit
n	Number of data blocks to be converted.	BIN 16-bit

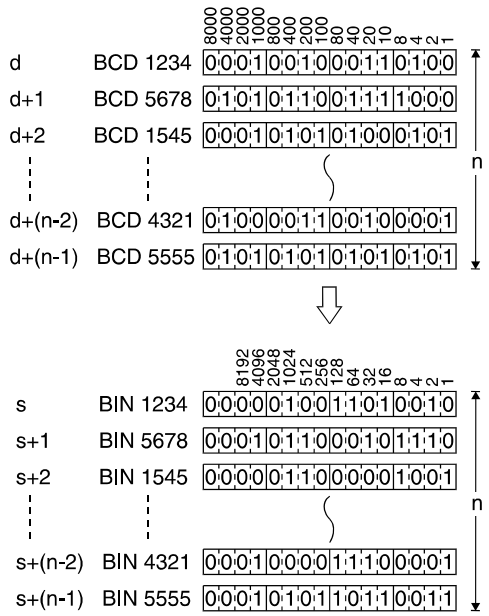
Functions

Conversion from BCD block data into BIN block data

BKBIN, BKBINP Conversion from BCD 4-digit block data into BIN 16-bit block data

This instruction converts each nth BCD 4-digit block in s into the nth BIN 16-bit block. Converted data is stored in d.

BIN 16-bit block data in s has to range within 0 to 9999.



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of data blocks determined by n exceeds the storage device numbers designated by s and d. (Error code 4101)
- BCD block data in s exceeds the relevant device range of 0 to 9999. (Error code 4100)
- The storage device numbers designated by s and d overlap. (Error code 4101)

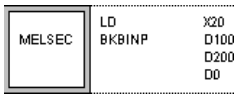
For details on index qualification refer to section 3.6 of this manual.

Program Example

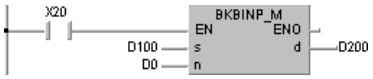
BKBINP

With leading edge from X20, the following program converts BCD 4-digit block data in D100 into BIN 16-bit block data. Converted data is stored in D200. The number of data blocks (3) converted is stored in D0.

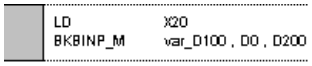
MELSEC Instruction List

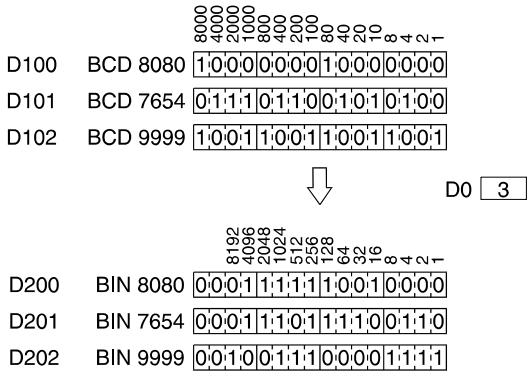


Ladder Diagram



IEC Instruction List





NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.3.16 ECON, ECONP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

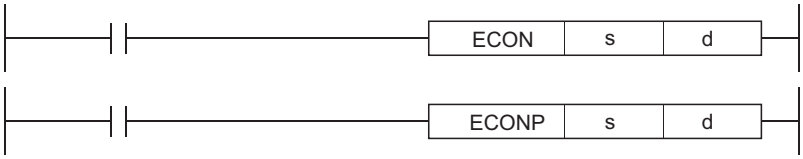
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	●	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

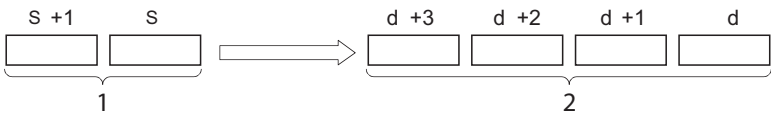
Set Data	Meaning	Data Type
s	Conversion source data, or head number of the device where conversion source data is stored	(Real number (single precision)
d	Head number of the device where the converted data is stored	(Real number (double precision))

Functions

Conversion from Single precision to Double precision

ECON Conversion from 32-bit into 64-bit floating point real number

This instruction converts 32-bit floating-point real number specified for s into 64-bit floating-point real number, and stores the conversion result to the device specified for d.



[1] 32-bit floating-point real number

[2] 64-bit floating-point real number

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

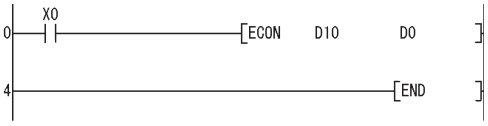
- The value of the designated device is not zero and not within the following range:
  $\pm 2^{-126} \leq$ (Value of designated device) $< \pm 2^{128}$
  (Error code 4140)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (Error code 4140)

Program Example

ECON

With leading edge from X0, the following program converts 32-bit floating-point real number of the devices D10 to D11, into 64-bit floating-point real number. Converted data is stored to the devices D0 to D3.

Ladder Diagram



6.3.17 EDCON, EDCONP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

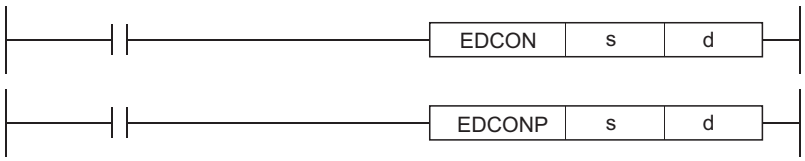
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	●	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

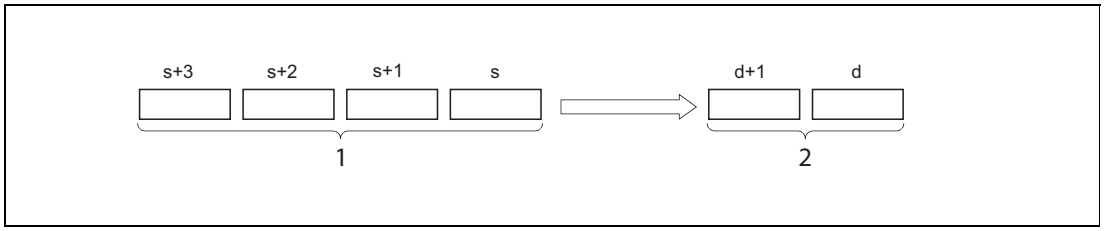
Set Data	Meaning	Data Type
s	Conversion source data, or head number of the device where conversion source data is stored	Real number (double precision)
d	Head number of the device where the converted data is stored	Real number (single precision))

Functions

Conversion from Double precision to Single precision

EDCON Conversion from 64-bit into 32-bit floating point real number

This instruction converts 64-bit floating-point real number specified for s into 32-bit floating-point real number, and stores the conversion result to the device specified for d.



[1] 64-bit floating-point real number

[2] 32-bit floating-point real number

Operation Errors

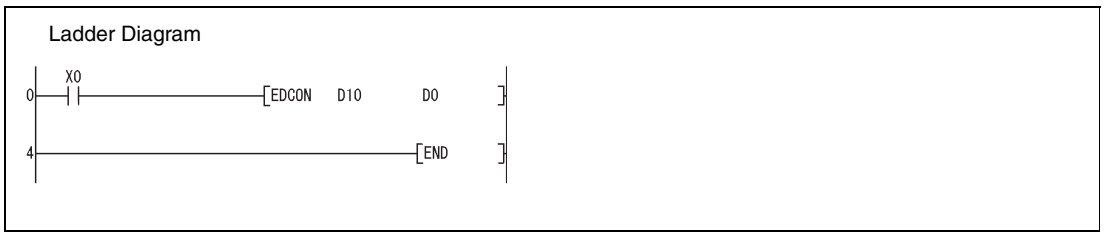
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the designated device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of designated device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The conversion result is not within the following range:
  $-2^{128} \leq$ (Conversion result) $\leq 2^{128}$
  (Error code 4141)

Program Example

EDCON

With leading edge from X0, the following program converts 64-bit floating-point real number of the devices D10 to D13, into 32-bit floating-point real number. Converted data is stored to the devices D0 and D1.



6.4 Data transfer instructions

These instructions transfer, invert, or exchange data. Refer to the following table for an overview of the instructions.

NOTE *Transferred data remain stored until they are replaced. Therefore, data even remain stored if the input condition of the transfer instruction is reset.*

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
BIN Data Transfer (16-/32-bit)	MOV	MOV_M
	MOVP	MOVP_M
	DMOV	DMOV_M
	DMOVP	DMOVP_M
Transfer of Floating Point Data (16-/32-bit)	EMOV	EMOV_M
	EMOVP	EMOVP_M
	EDMOV	EDMOV_M
	EDMOVP	EDMOVP_M
Transfer of Character String Data	$MOV	STRING_MOV_M
	$MOVP	STRING_MOVP_M
Inverted BIN Data Transfer (16-/32-bit)	CML	CML_M
	CMLP	CMLP_M
	DCML	DCML_M
	DCMLP	DCMLP_M
Block Data Transfer	BMOV	BMOV_M
	BMOVP	BMOVP_M
Block Transfer of identical Data (16-/32-bit)	FMOV	FMOV_M
	FMOVP	FMOVP_M
	DFMOV	DFMOV_M
	DFMOVP	DFMOVP_M
BIN Data Exchange (16-/32-bit)	XCH	XCH_M
	XCHP	XCHP_M
	DXCH	DXCH_M
	DXCHP	DXCHP_M
BIN Data Exchange (16-bit blocks)	BXCH	BXCH_M
	BXCHP	BXCHP_M
Byte Exchange (upper and lower byte)	SWAP	SWAP_MD
	SWAPP	SWAP_P_MD

NOTE *Within the IEC editors please use the IEC commands.*

6.4.1 MOV, MOVP, DMOV, DMOVP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

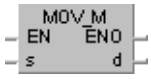
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

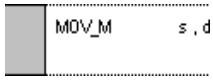
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

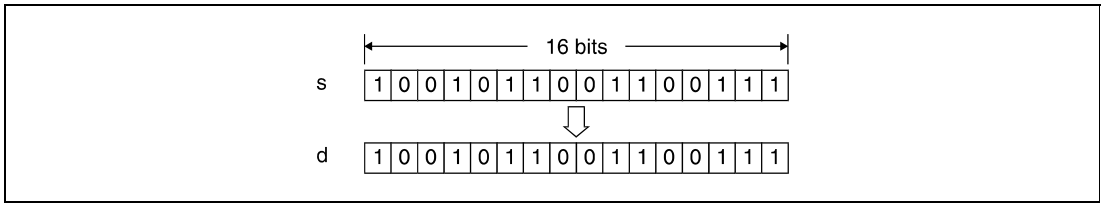
Set Data	Meaning	Data Type
s	Source data, or first number of device storing data to be transferred.	BIN 16-/32-bit
d	First number of destination device to store transferred data.	

Functions

BIN 16-bit/32-bit data transfer

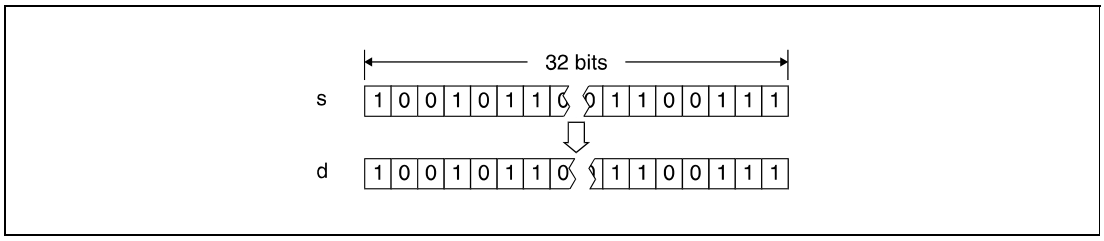
MOV BIN 16-bit data transfer

The MOV instruction transfers BIN 16-bit data in s to the device designated by d.



DMOV BIN 32-bit data transfer

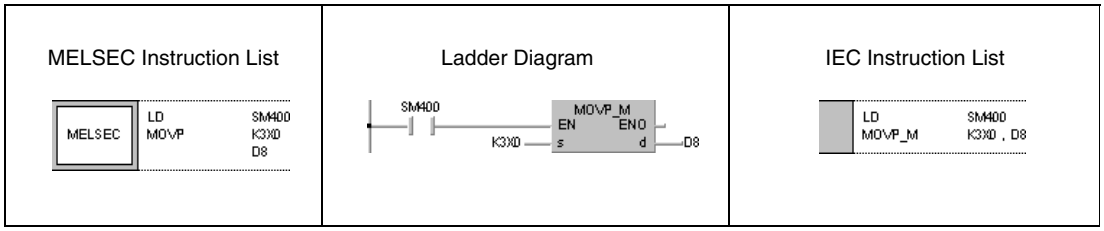
The DMOV instruction transfers BIN 32-bit data in s to the device designated by d.



Program Example 1

MOVP

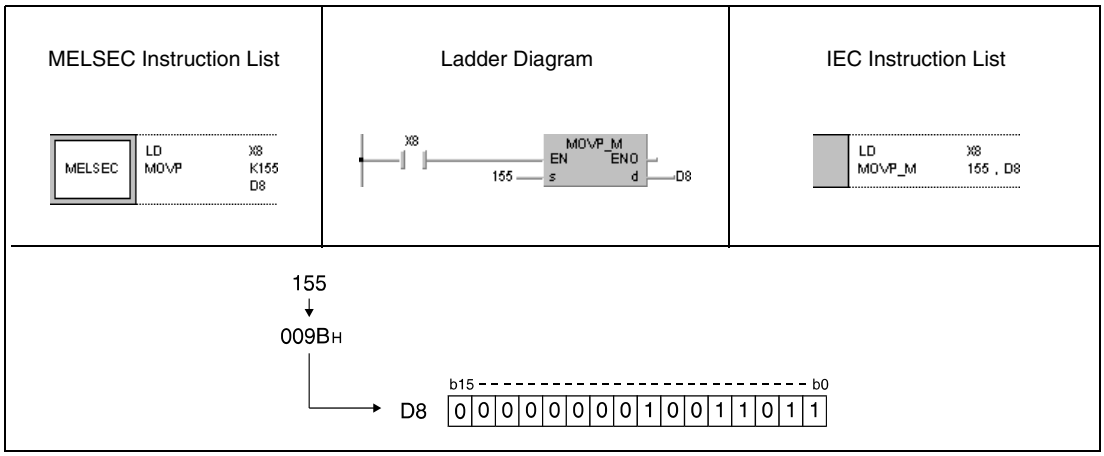
The following program transfers data at X0 through XB to D8.



Program Example 2

MOVP

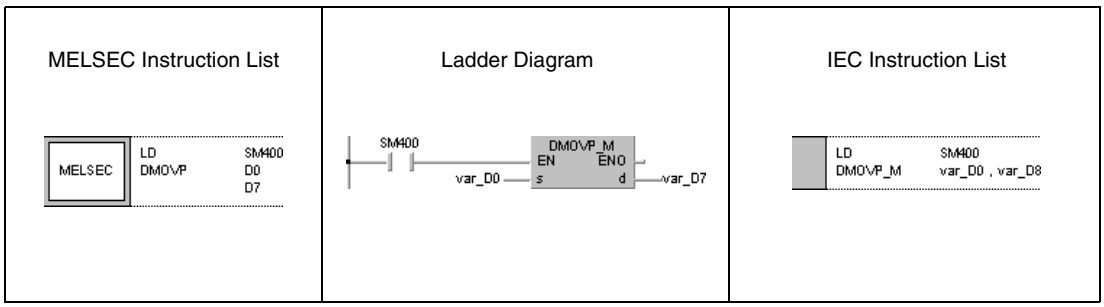
With leading edge from X8, the following program transfers the constant 155 as BIN value to D8.



Program Example 3

DMOVP

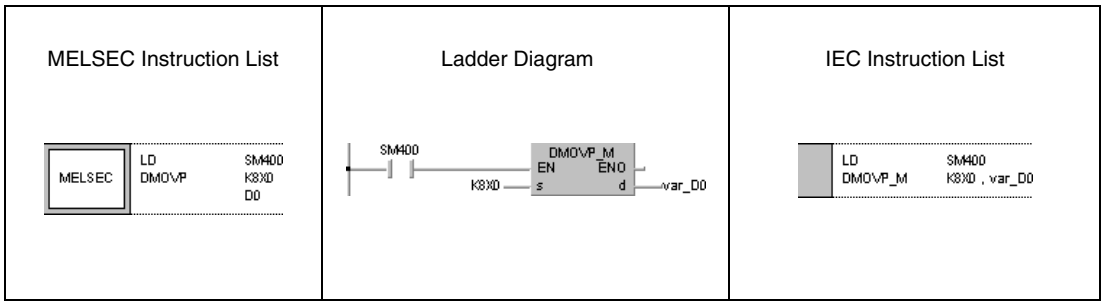
The following program transfers data in D0 and D1 to D7 and D8.



Program Example 4

DMOVP

The following program transfers data at X0 through X1F to D0 and D1.



NOTE

The program examples 3 and 4 will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.4.2 EMOV, EMOVP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

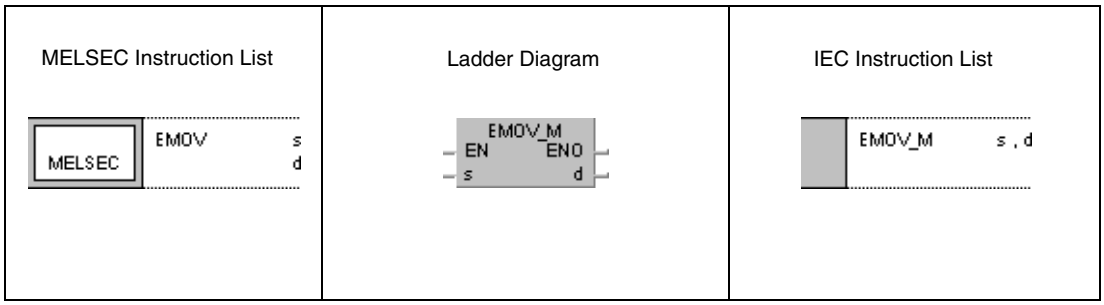
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

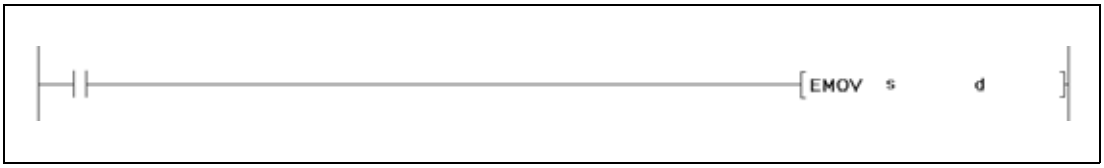
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Available only in multiple Universal model QCPU, LCPU

GX IEC Developer



GX Works2



Variables

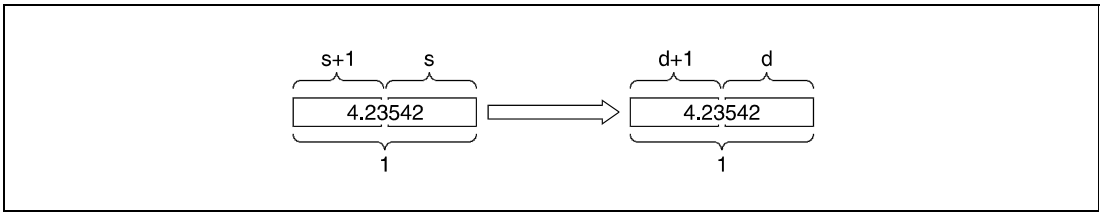
Set Data	Meaning	Data Type
s	Floating point data, or first number of device storing data to be transferred.	Real number
d	First number of device storing transferred floating point data.	

Functions

Floating point data transfer (Single precision)

EMOV 32-bit floating point data transfer

The EMOV instruction transfers 32-bit floating point data in s to the device designated by d.

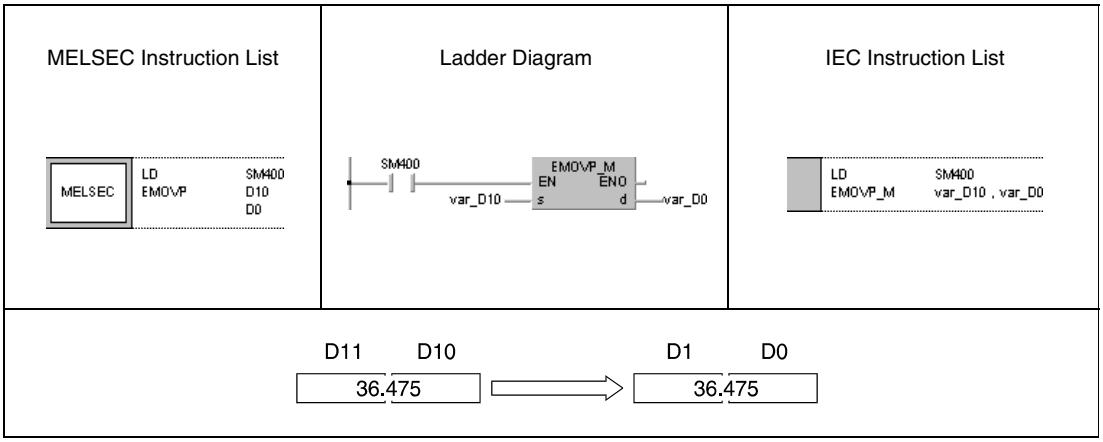


[1] 32-bit floating point number, data type real number

Program Example 1

EMOVP

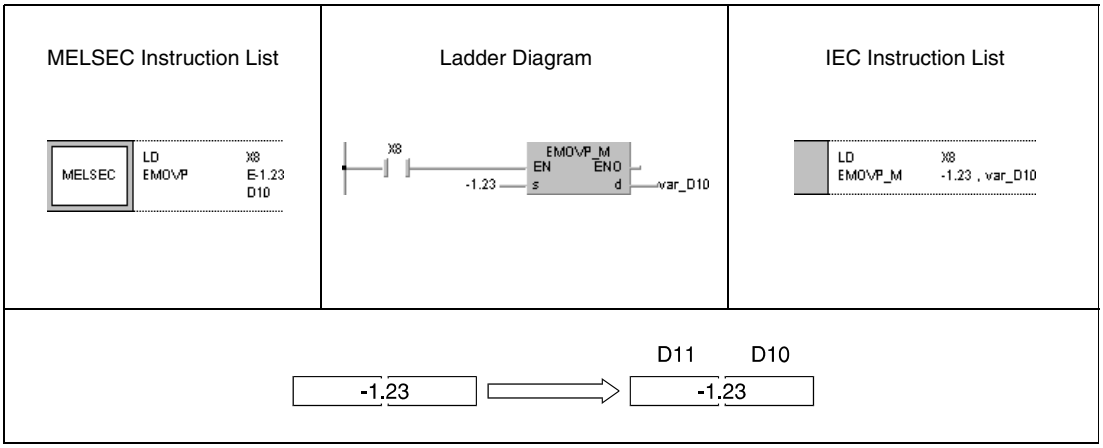
The following program transfers 32-bit floating point data in D10 and D11 to D0 and D1.



Program Example 2

EMOVP

With leading edge from X8, the following program transfers the real number –1.23 to D10 and D11.



NOTE

These programs will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.4.3 EDMOV, EDMOVP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

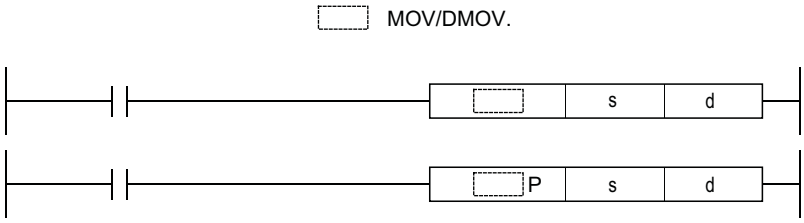
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

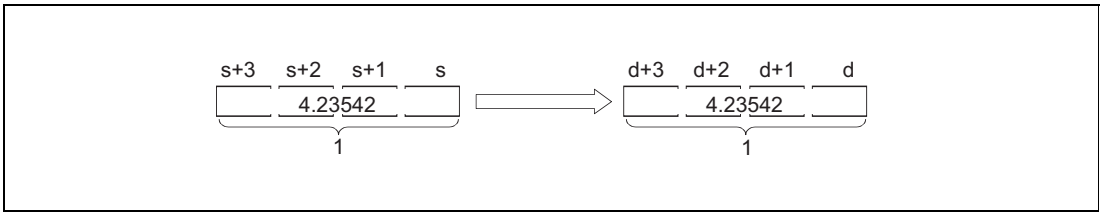
Set Data	Meaning	Data Type
s	Floating point data, or first number of device storing data to be transferred.	Real number
d	First number of device storing transferred floating point data.	

Functions

Floating point data transfer (Double precision)

EDMOV 64-bit floating point data transfer

The EDMOV instruction transfers 64-bit floating point data in s to the device designated by d.

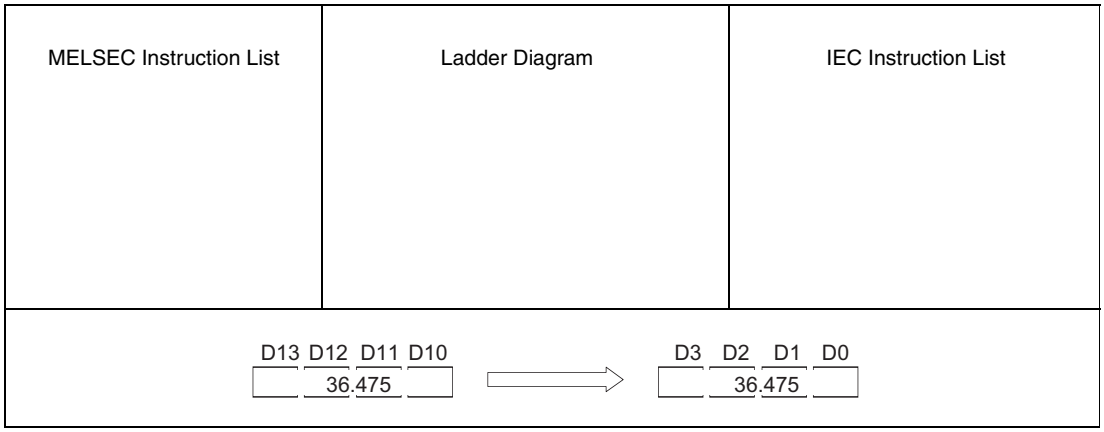


[1] 64-bit floating point number, data type real number

Program Example 1

EDMOVP

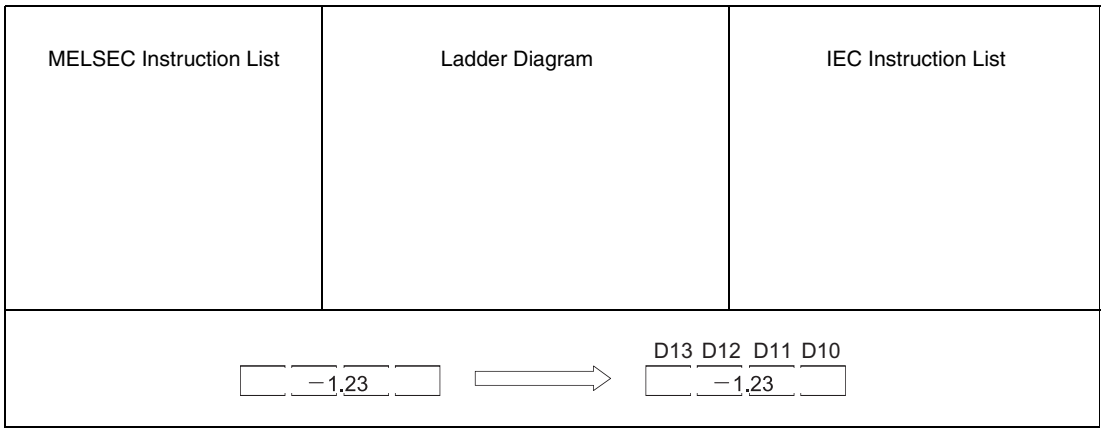
The following program transfers 64-bit floating point data in D10 to D13 to D0 to D3.



Program Example 2

EDMOVP

With leading edge from X8, the following program transfers the real number –1.23 to D10 to D13.



NOTE

These programs will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.4.4 $MOV, $MOVP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

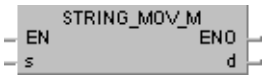
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

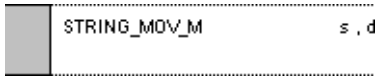
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

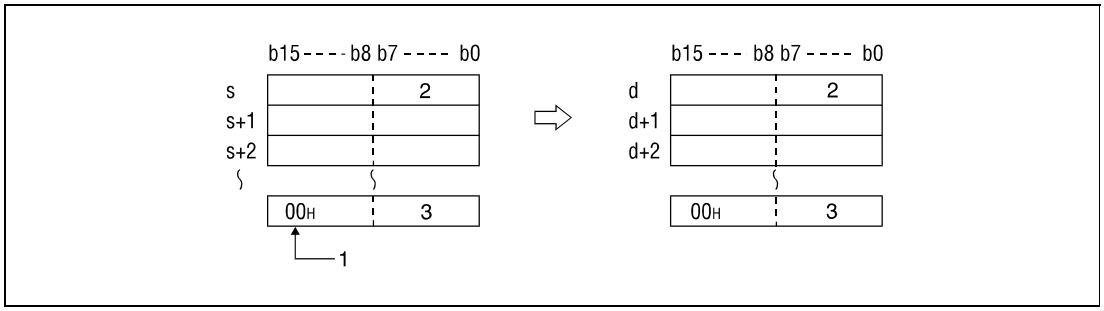
Variables

Set Data	Meaning	Data Type
s	Character string data, or first number of device storing data to be transferred.	Character string
d	First number of device storing transferred character string data.	

Functions **Character string data transfer**

$MOV Character string data transfer

The $MOV instruction transfers character string data in s to d. The instruction transfers character string data from the first number of device designated by s up to the number of device storing the code "00H" (end of string) in one operation.

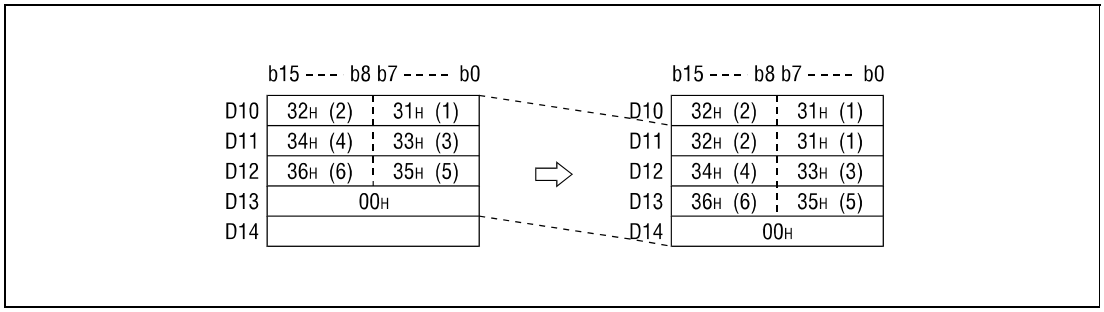


[1] Indicates end of character string

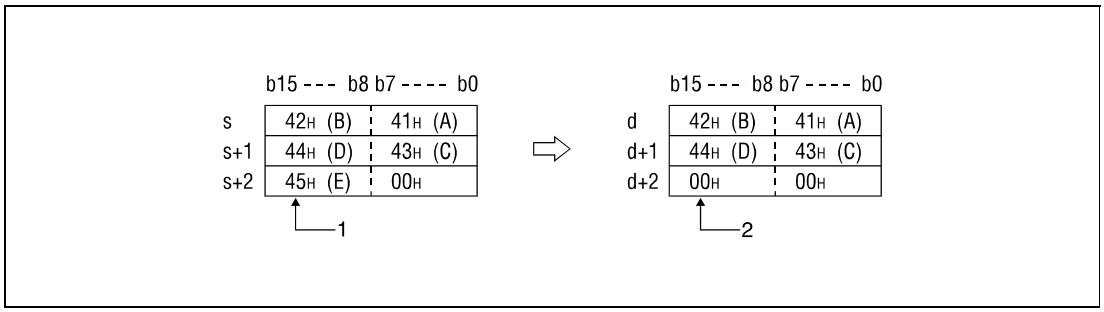
[2] 1st character

[3] nth character

The $MOV instruction is even performed without error messages, if the range of devices storing character string data to be transferred (s through s+n) overlaps with the range of devices storing transferred data (d through d+n). The $MOV instruction performs as follows, if character string data in D10 through D13 is transferred to D11 through D14:



If the code "00H" is stored at lower bytes of s+n, the characters following at the higher bytes are omitted. In d+n, the transferred code "00H" will be stored at both, the higher bytes and the lower bytes:



[1] Character is not transferred.

[2] "00H" is stored automatically.

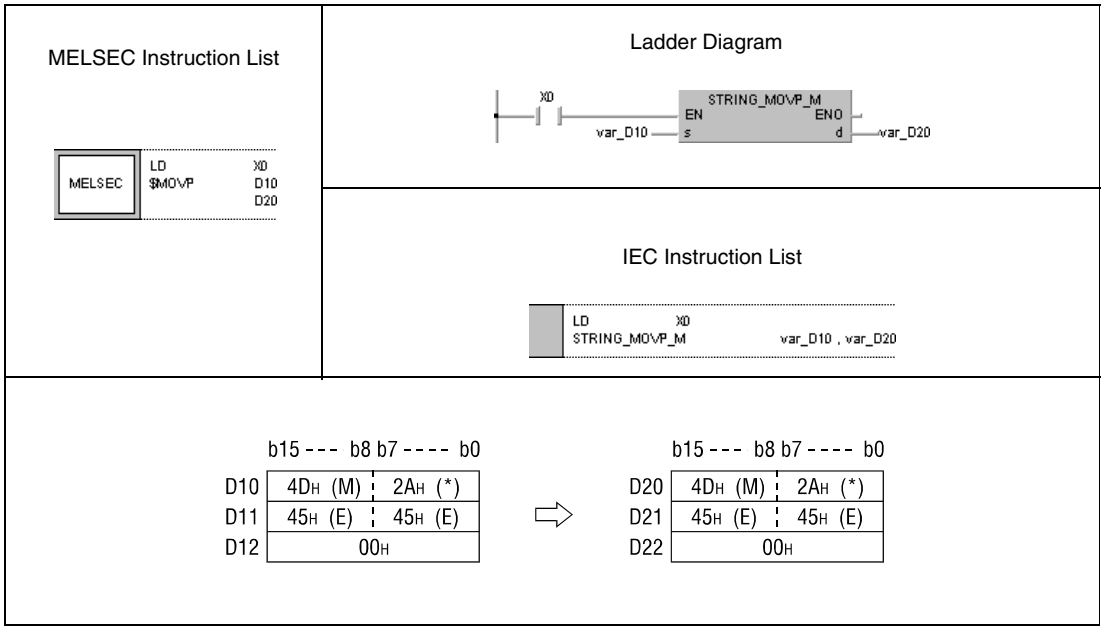
Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The code "00H" does not exist in character string data designated by s. (Error code 4101)
- Character string data in s cannot be transferred completely to d. (Error code 4101)
- The character string of s exceeds 16383 characters. (Error code 4101)

Program Example

With leading edge from X0, the following program transfers character string data at D10 through D12 to D20 through D22.



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.4.5 CML, CMLP, DCML, DCMLP

CPU

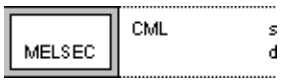
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

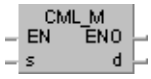
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer

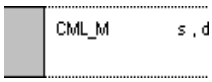
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

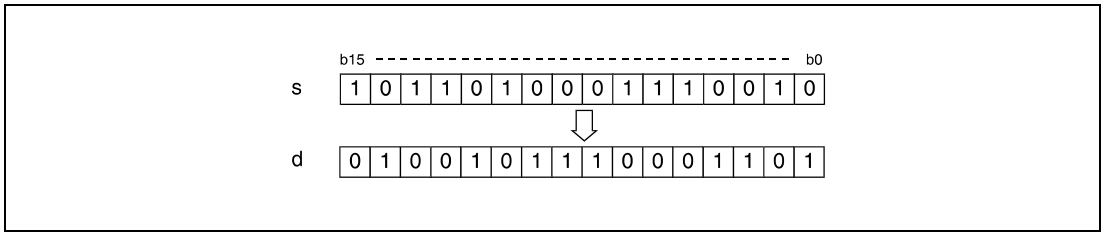
Set Data	Meaning	Data Type
s	BIN data, or first number of device storing data to be inverted.	BIN 16-/32-bit
d	First number of device storing inverted data.	

Functions

BIN 16-bit/32-bit data inversion

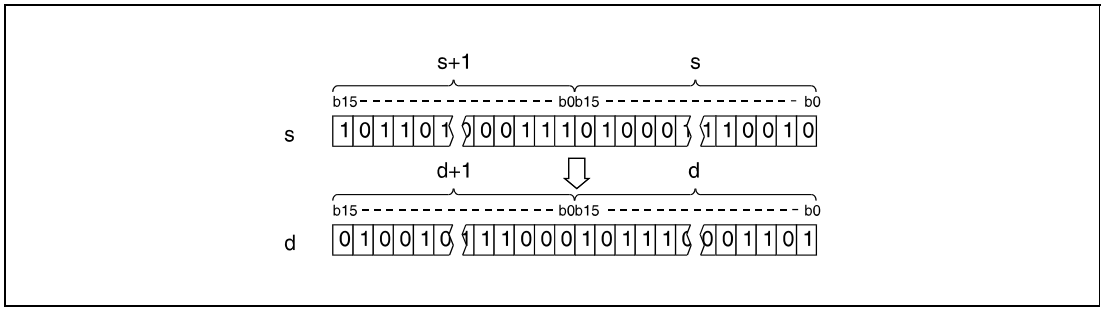
CML BIN 16-bit data inversion

BIN 16-bit data in s is inverted bit by bit. The result is stored in d.



DCML BIN 32-bit data inversion

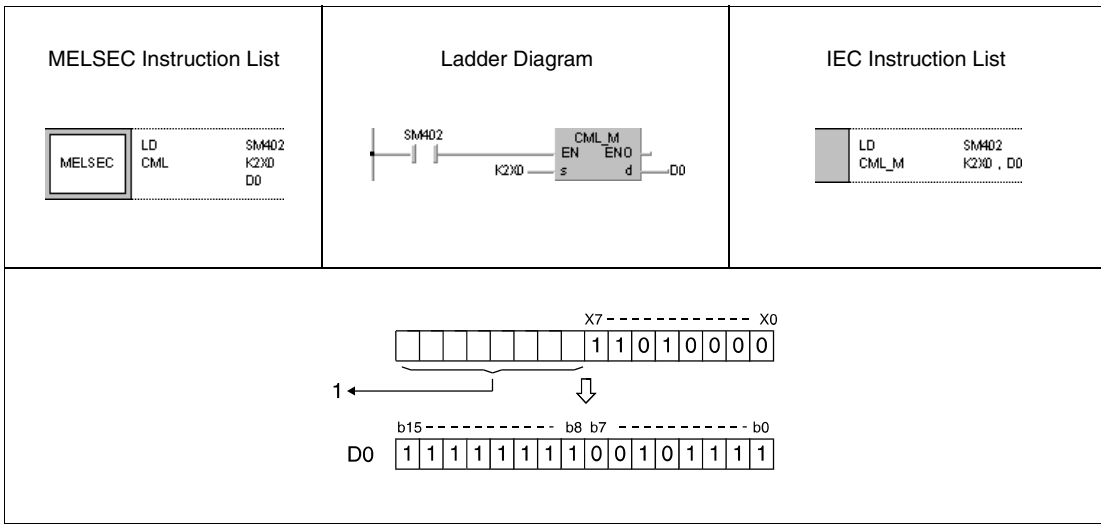
BIN 32-bit data in s is inverted bit by bit. The result is stored in d.



Program Example 1

CML

While SM402 is set, the following program transfers data at X0 through X7 inverted to D0.



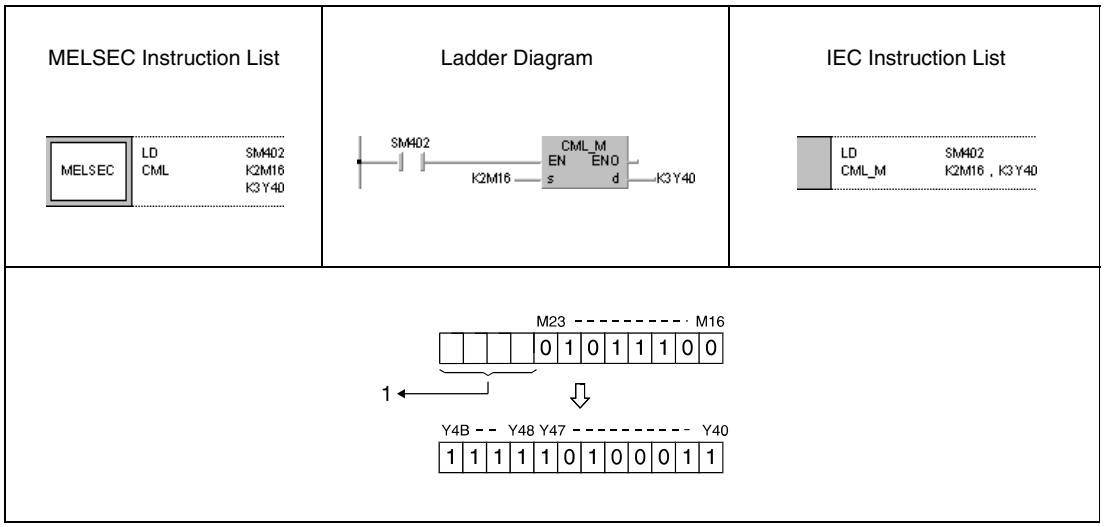
[1] Undesignated bits are read as 0.

In this example the number of bits in s is smaller than the number of bits in d.

Program Example 2

CML

While SM402 is set, the following program transfers data in M16 through M23 inverted to K3 Y40 (Y40 through Y4F). Y48 through Y4B are all set (1), because they were read as 0.



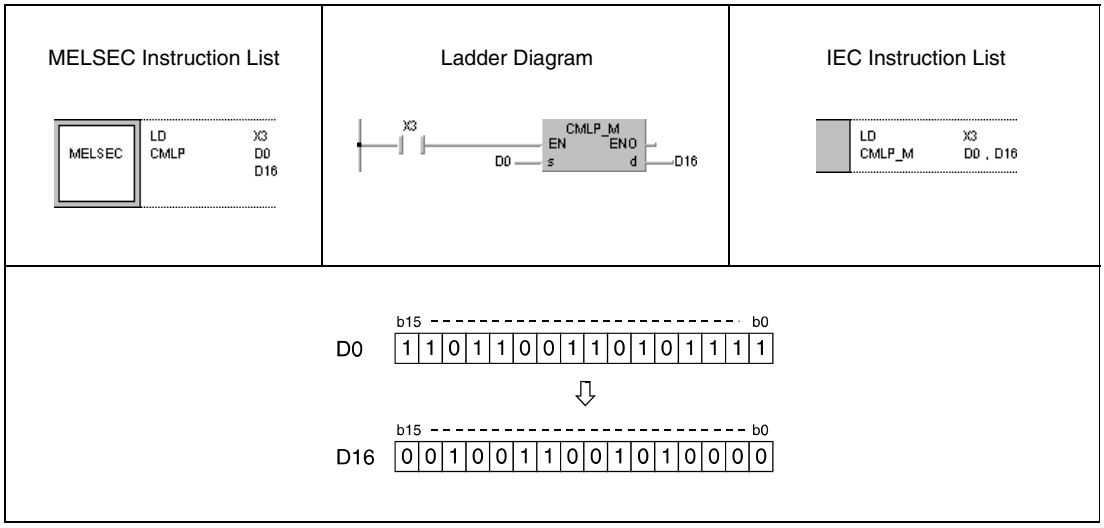
[1] Undesignated bits are read as 0.

In this example the number of bits in s is smaller than the number of bits in d.

Program Example 3

CMLP

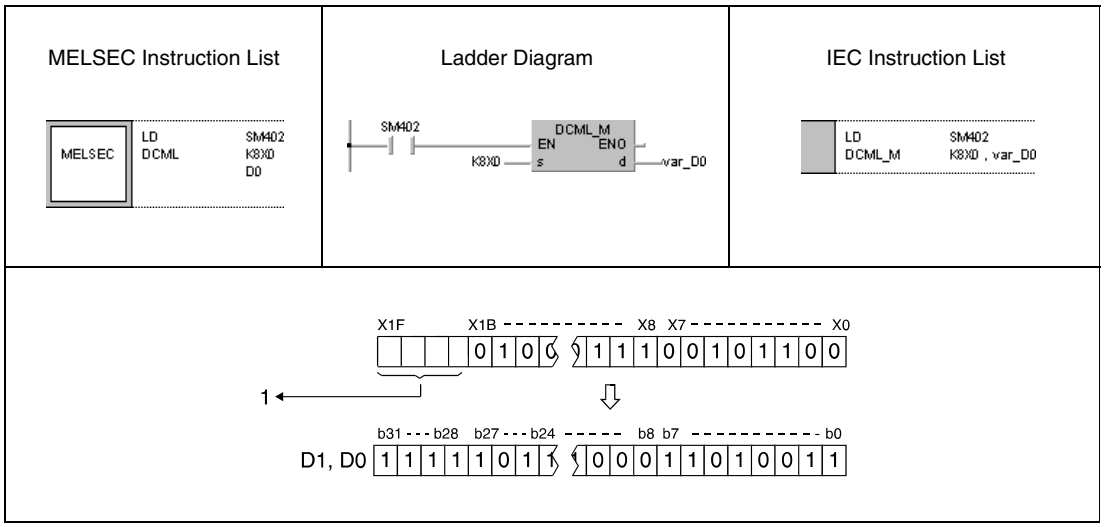
With leading edge from X3, the following program transfers data in D0 inverted to D16.



Program Example 4

DCML

While SM402 is set, the following program transfers data at X0 through X1F inverted to D0 and D1.



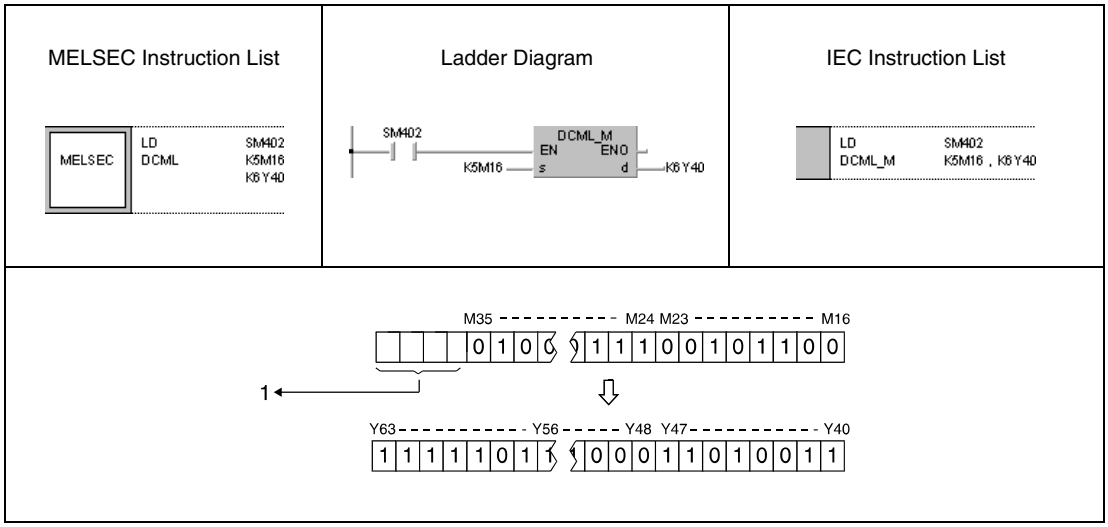
[1] Undesignated bits are read as 0.

In this example the number of bits in s is smaller than the number of bits in d.

Program Example 5

DCML

While SM402 is set, the following program transfers data in M16 through M35 inverted to Y40 and Y57.



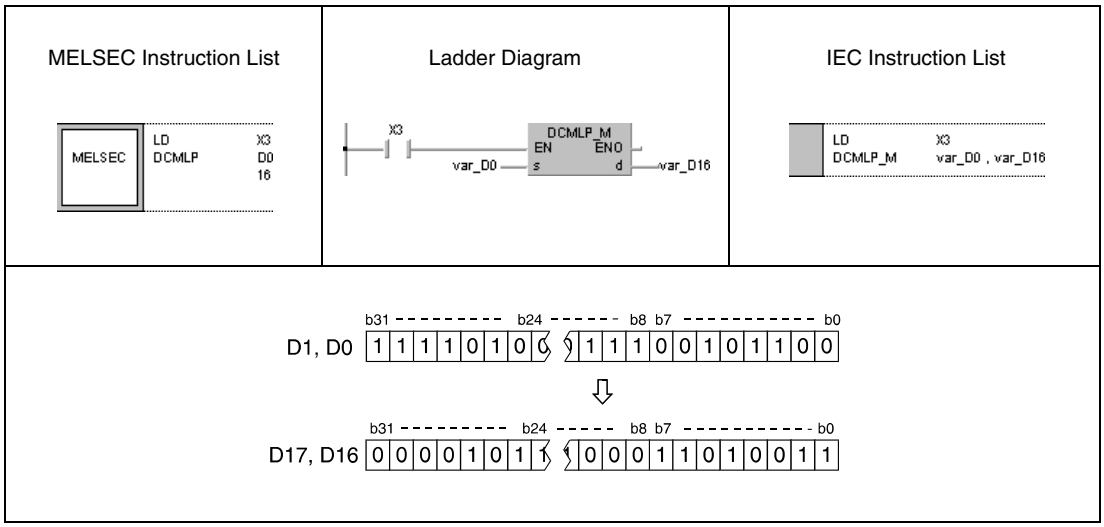
[1] Undesignated bits are read as 0.

In this example the number of bits in s is smaller than the number of bits in d.

Program Example 6

DCMLP

With leading edge from X3, the following program transfers data in D0 and D1 inverted to D16 and D17.



NOTE *The program examples 4 and 6 will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.4.6 BMOV, BMOVP

CPU

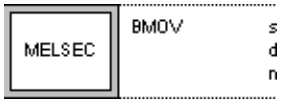
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	—	—	—
d	●	●	●	●	●	●	—	—	—
n	●	●	●	●	●	●	●	●	—

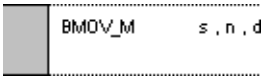
GX IEC Developer

MELSEC Instruction List

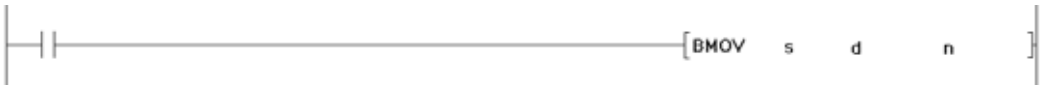


Ladder Diagram

IEC Instruction List



GX Works2



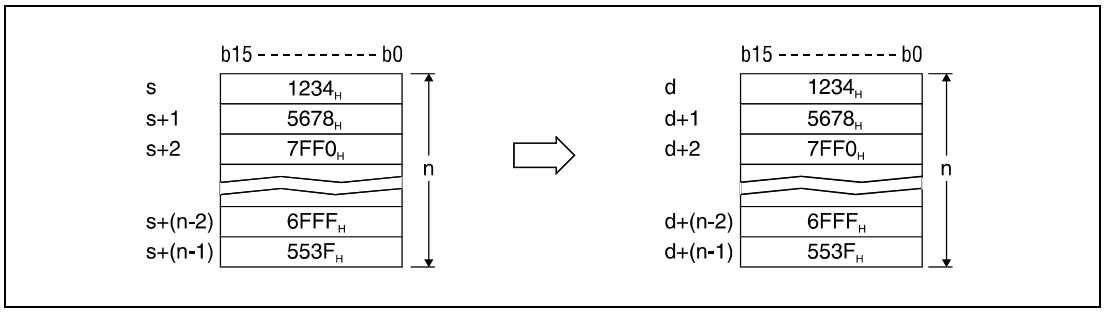
Variables

Set Data	Meaning	Data Type
s	First number of device storing data to be transferred.	BIN 16-bit
d	First number of device storing transferred data.	
n	Number of data blocks to be transferred.	

Functions **BIN block data transfer**

BMOV BIN 16-bit block data transfer

The BMOV instruction transfers successive data blocks in a batch. The first number of device storing block data is designated by s. The number of successive data blocks to be transferred is determined by n. The data are transferred to the device designated by d onwards.



A transfer can even be performed without operation errors, if the source and the destination devices overlap. Transfer to the smaller device number begins from s. Transfer to the larger device number begins from s+(n-1).

However, as shown in the example below, when transferring data from R to ZR, or from ZR to R, the range to be transferred (source) and the range of destination must not overlap. Transfer from R to R, or from ZR to ZR can be performed without any problem.

– ZR transfer range
(specified head No. of ZR) to
(specified head No. of ZR + the number of transfers -1)

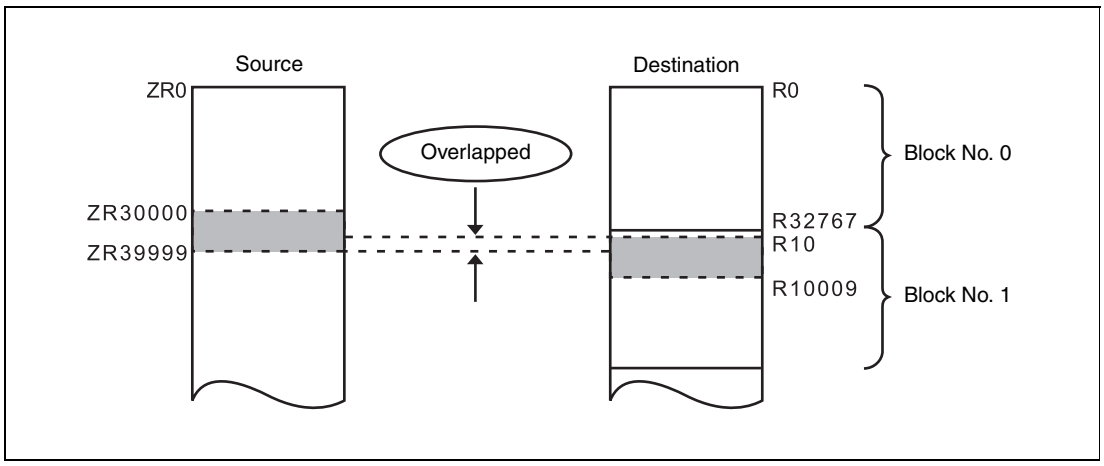
– R transfer range
((specified head No. of R + file register block No. 32768) to
(specified head No. of R + file register block No. 32768 + the number of transfers -1))

Example Transfer ranges of ZR and R overlap when transferring 10000 blocks of data from ZR30000 to R10 (block no. 1 of destination).
Die Übertragungsbereiche von ZR und R überlappen sich, wenn 10000 Datenblöcke von ZR30000 nach R10 (Block-Nr. 1 des Datenziels) übertragen werden.

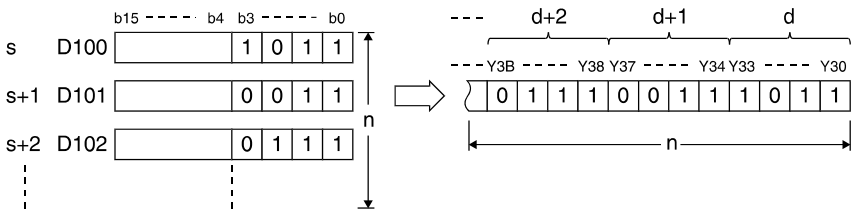
– ZR transfer range: (30000) to (30000 + 10000 -1) = (30000) to (39999)

– R transfer range: (10 + (1 x 32768)) to (10 + (1 x 32768) + 10000 -1)
= (32778) to (42777)

Therefore the range 32778 to 39999 overlaps and the data is not transferred correctly.



If s is a word device and d is a bit device, the number of bits designated by digit designation for the bit device will be the object bits for the word device. If K1Y30 is designated by d, the object bits for the word device s are the lower 4 bits.



If s and d are bit devices, the number of bits in s and d must equal.

When using a link direct device and an intelligent function module device for s and d, only either of s or d can be used.

Whether to check a device range during execution of the BMOV instruction can be selected with the device range check inhibit flag (SM237) (only when the conditions for subset processing are established). While SM237 is ON, whether s to s + (n) -1 and d to d + (n) - 1 are within the device range or not are not checked.

NOTES

SM237 can be used only for the Universal model QCPU whose first 5 digits of serial number is 10012 or higher and for LCPU.

While SM237 is on, do not make the following access.

- *The indexing target exceeds the device range.*
- *The value obtained from "d to d+(n)- 1" is over the boundaries of the device ranges.*
- *Accessing the file register with file register not set.*
- *Accessing the area where the multiple CPU high speed transmission area device is not available (only for the QCPU).*

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

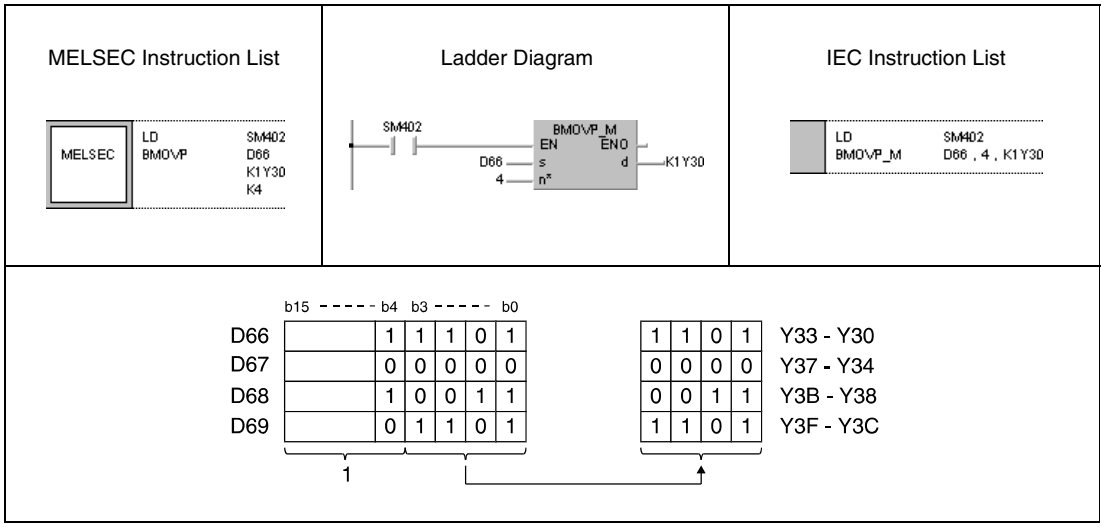
- The number of data blocks determined by n exceeds the storage device numbers designated by s and d.
  (Error code 4101)

Program Example 1

BMOVP

With leading edge from SM402, the following program transfers the lower 4 bits of data (b0 through b3) in D66 through D69 to the outputs Y30 through Y3F. The number of blocks (4) to be transferred is determined by the constant K4.

The bit patterns show the structure of bits before and after the transfer.



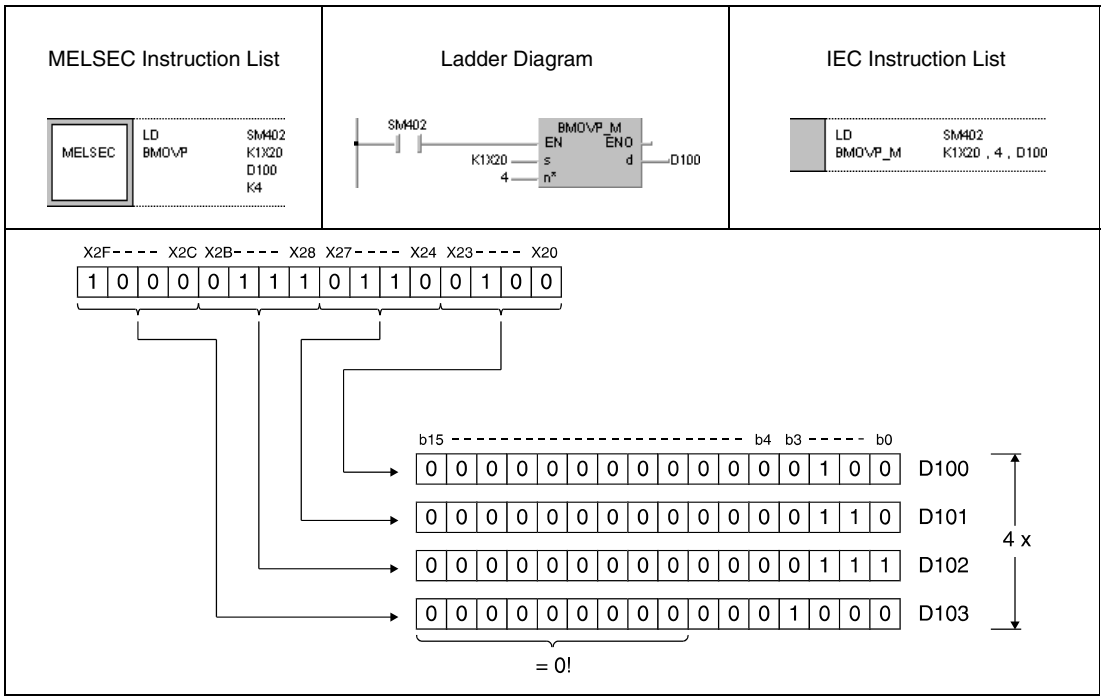
[1] These bits are ignored.

Program Example 2

BMOVP

With leading edge from SM402, the following program transfers data at X20 through X2F to D100 through 103. The number of blocks (4) to be transferred is determined by the constant K4.

The bit patterns show the structure of bits before and after the transfer.



6.4.7 FMOV, FMOVP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

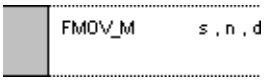
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

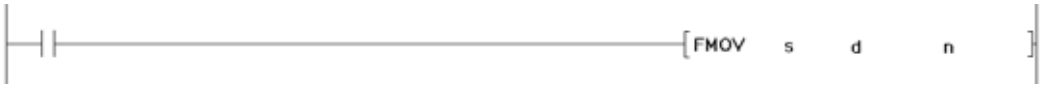
MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2



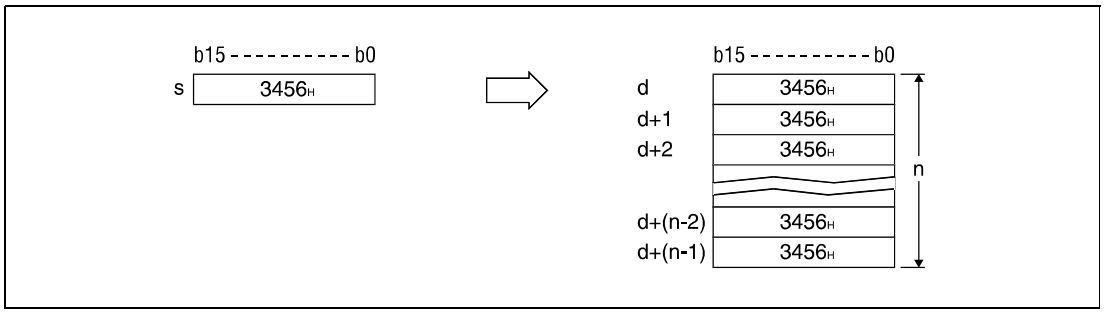
Variables

Set Data	Meaning	Data Type
s	First number of device storing data to be transferred.	BIN 16-bit
d	First number of device storing transferred data.	
n	Number of data blocks to be transferred.	

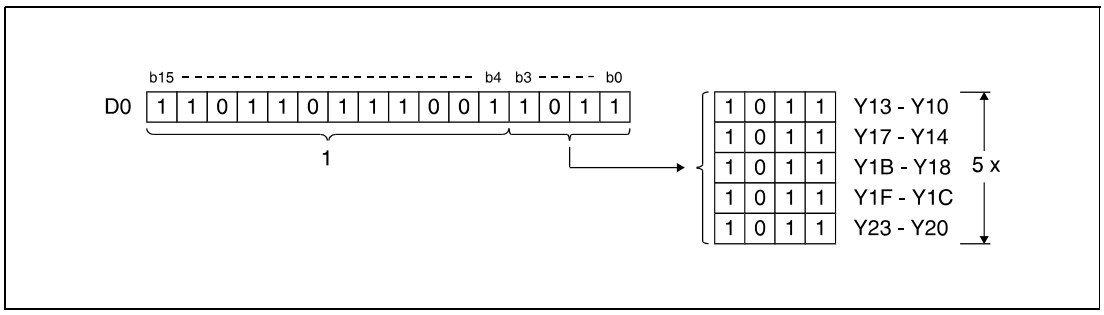
Functions **Identical BIN block data transfer**

FMOV Identical BIN 16-bit block data transfer

The FMOV instruction transfers 16-bit block data in s to d through d+(n-1). Each device of the data block from d through d+(n-1) stores the value from s.



If s is a word device and d is a bit device, the number of bits designated by digit designation for the bit device will be the object bits for the word device.



[1] These bits are ignored.

If s and d are bit devices, the number of bits in s and d must equal.

Whether to check a device range during execution of the FMOV instruction can be selected with the device range check inhibit flag (SM237) (only when the conditions for subset processing are established). While SM237 is ON, whether d to d + (n) - 1 is within the device range or not is not checked.

NOTES

SM237 can be used only for the Universal model QCPU whose first 5 digits of serial number is 10012 or higher and for LCPU.

While SM237 is on, do not make the following access.

- *The indexing target exceeds the device range.*
- *The value obtained from "d to d+(n–1)" is over the boundaries of the device ranges.*
- *Accessing the file register with file register not set.*
- *Accessing the area where the multiple CPU high speed transmission area device is not available (only for the QCPU).*

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

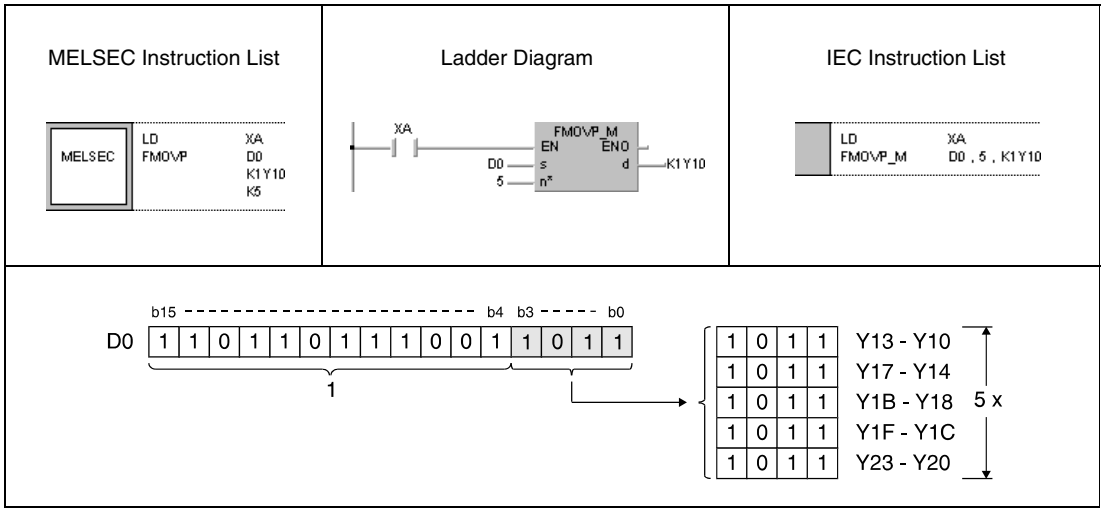
- The number of data blocks determined by n exceeds the storage device numbers designated by d. (Error code 4101)

Program Example 1

FMOVP

With leading edge from XA, the following program transfers the lower 4 bits of data (b0 through b3) in D0 to the outputs Y10 through Y23. The number of blocks (5) is determined by the constant K5.

The bit patterns show the structure of bits before and after the transfer.



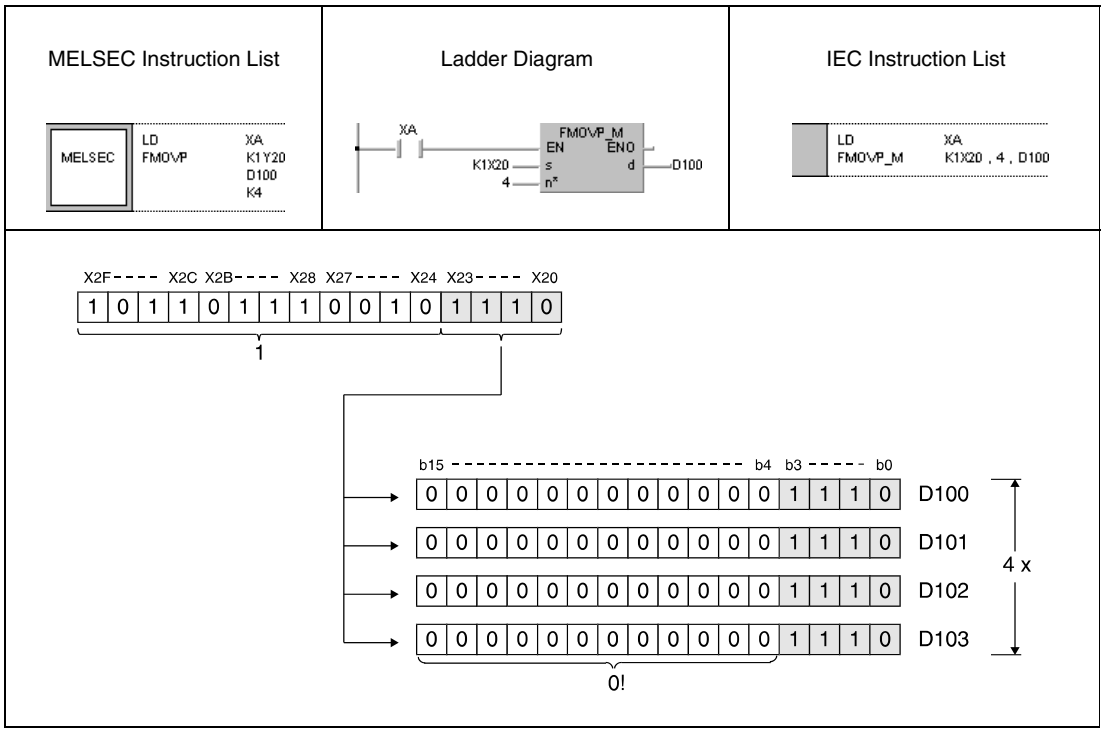
[1] These bits are ignored.

Program Example 2

FMOVP

With leading edge from XA, the following program transfers data at X20 through X23 to D100 through D103. The number of blocks (4) to be transferred is determined by the constant K4.

The bit patterns show the structure of bits before and after the transfer.



[1] These bits are ignored.

6.4.8 DFMOV, DFMOVP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

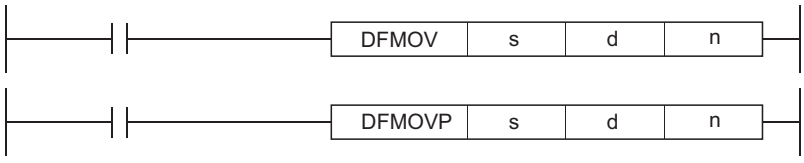
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

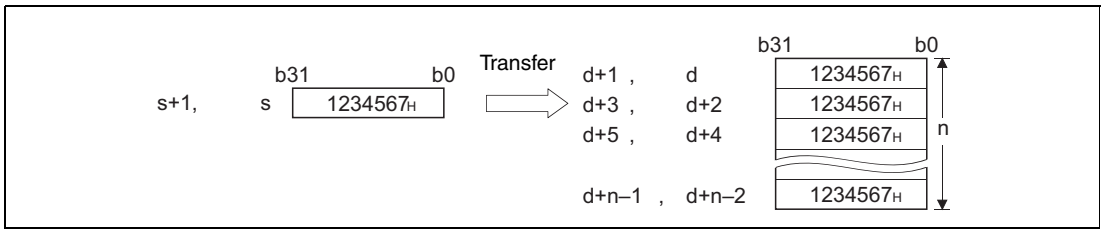
Set Data	Meaning	Data Type
s	First number of device storing data to be transferred.	BIN 32-bit
d	First number of device storing transferred data.	
n	Number of data blocks to be transferred.	

Functions

Identical BIN 32-bit block data transfer

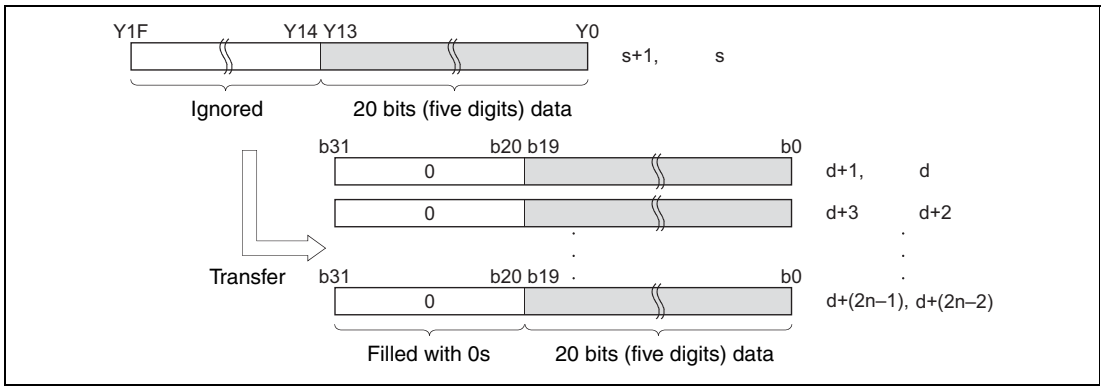
DFMOV Identical BIN 32-bit block data transfer

The DFMOV instruction transfers 32-bit data in s to d through d+(n-2). Each device of the data block from d through d+(n-2) stores the value from s.



If s is a word device and d is a bit device, the number of bits designated by digit designation for the bit device will be the object bits for the word device.

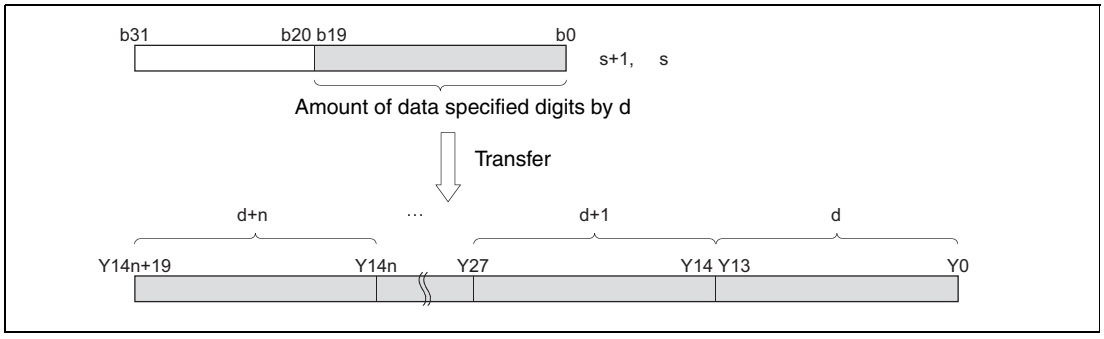
If K5Y0 is specified by s, the lower 20 bits (five digits) of the word device specified by s will be the object.



If d specifies data of a device with digit specification, the amount of data stored in the device specified by d will be transferred.

If K5Y0 is specified by d, the lower 20 bits of the word device specified by s will be the object.

If both s and d specify data of a device with digit specification, the amount of data specified by d will be transferred regardless of the number of digits.



If the value specified by n is 0, the instruction will be not processed.

Whether to check a device range during the execution of the DFMOV instruction can be selected with the device range check inhibit flag (SM237). (Only when the conditions of the subset processing are established).

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

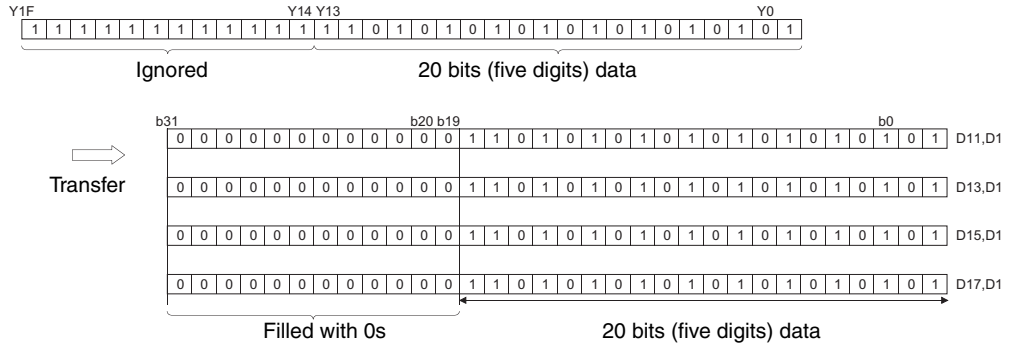
- The value specified by n is negative.
  (Error code 4100)
- The number of data blocks determined by n exceeds the storage device numbers designated by d.
  (Error code 4101)

Program Example

DFMOVP

With leading edge from M0, the following program transfers the value of data (Y0 to Y13 (20 bits) into D10 to D17.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



6.4.9 XCH, XCHP, DXCH, DXCHP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

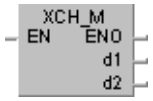
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d1	●	●	●	●	●	●	●	—	—
d2	●	●	●	●	●	●	●	—	—

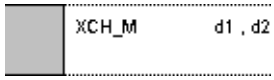
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

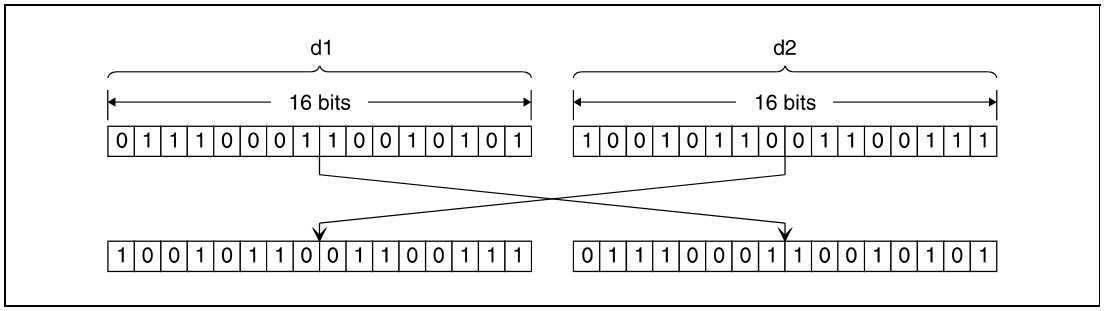
Variables

Set Data	Meaning	Data Type
d1	First number of device storing data to be exchanged.	BIN 16-/32-bit
d2		

Functions **BIN data exchange**

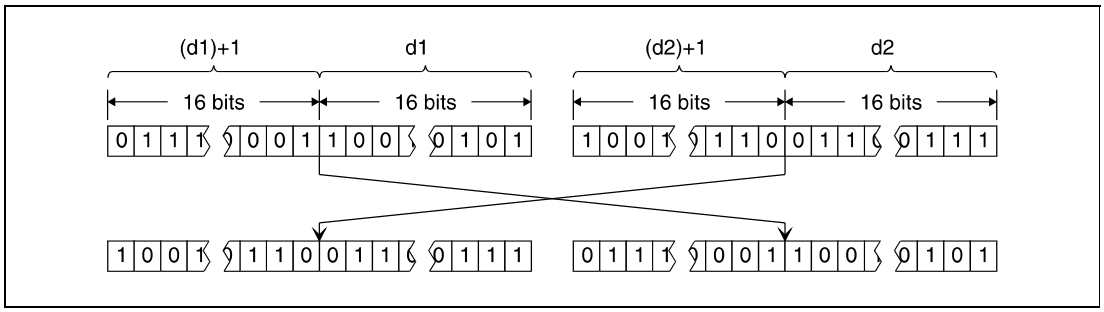
XCH BIN 16-bit data exchange

The XCH instruction e**xch**anges BIN 16-bit data in d1 and BIN 16-bit data in d2.



DXCH BIN 32-bit data exchange

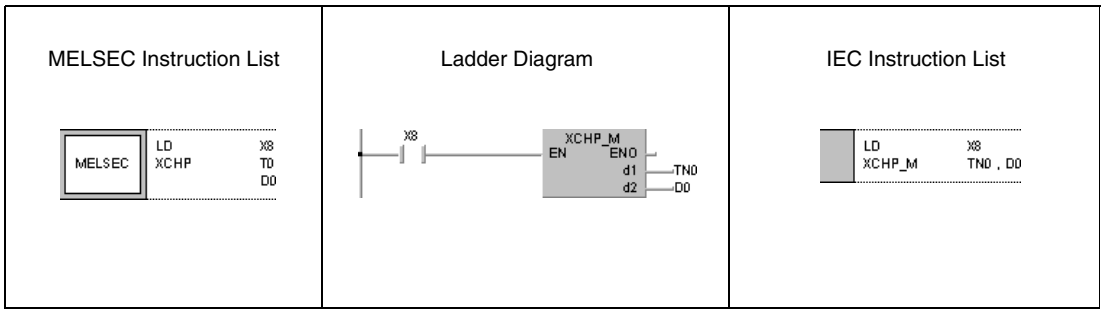
The DXCH instruction exchanges BIN 32-bit data in (d1)+1, d1 and BIN 32-bit data in(d2)+1, d2.



Program Example 1

XCHP

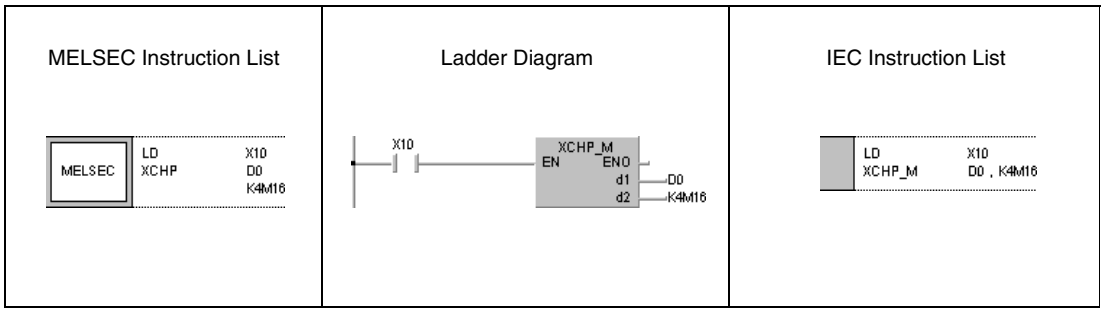
With leading edge from X8, the following program exchanges data in D0 and the actual value in T0.



Program Example 2

XCHP

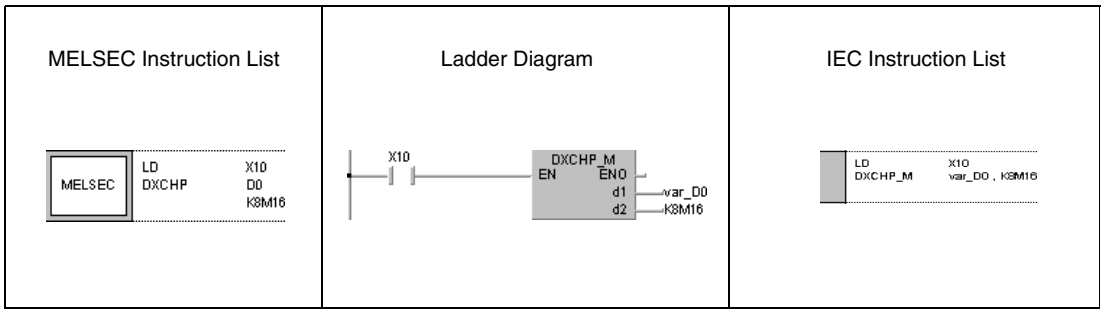
With leading edge from X10, the following program exchanges data in D0 and data in M16 through M31.



Program Example 3

DXCHP

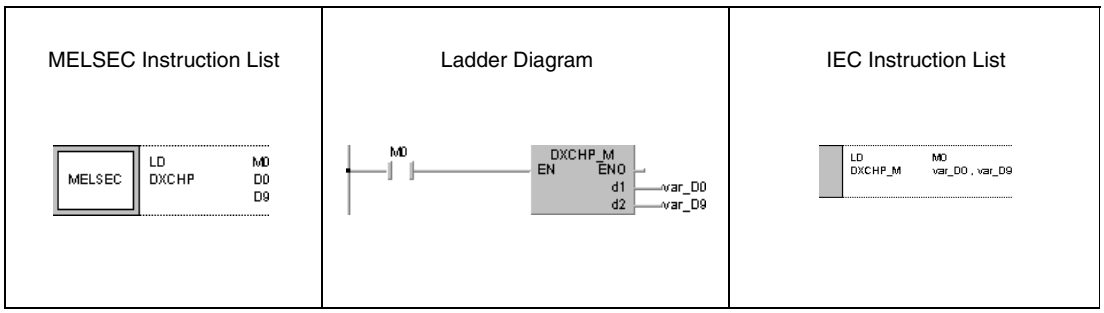
With leading edge from X10, the following program exchanges data in D0 and D1 and data in M16 through M47.



Program Example 4

DXCHP

With leading edge from M0, the following program exchanges data in D0 and D1 and data in D9 and D10.



NOTE

The program examples 3 and 4 will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.4.10 BXCH, BXCHP

CPU

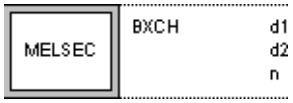
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

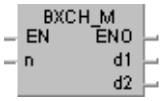
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d1	—	●	●	—	—	—	—	—	—
d2	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

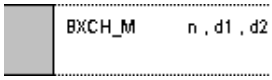
MELSEC Instruction List



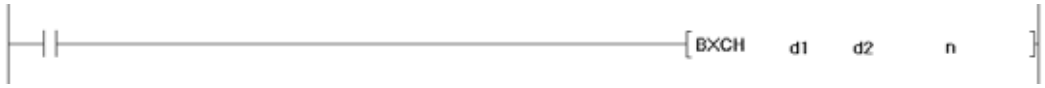
Ladder Diagram



IEC Instruction List



GX Works2



Variables

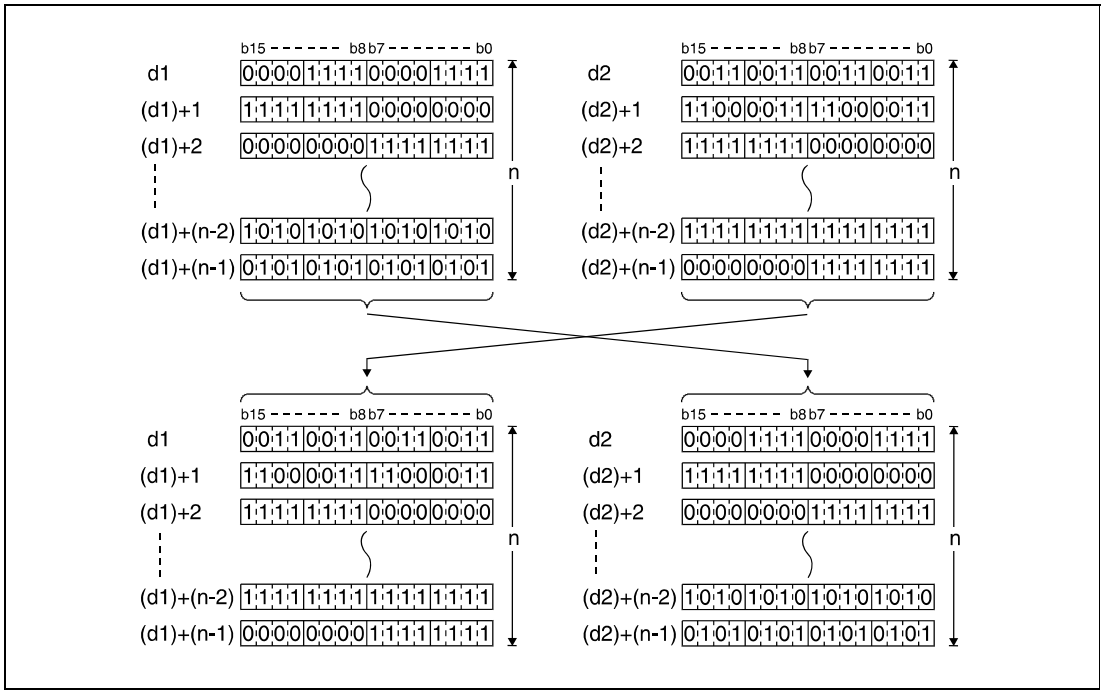
Set Data	Meaning	Data Type
d1	First number of device storing data to be exchanged	BIN 16-bit
d2		
n	Number of exchanges	

Functions

BIN block data exchange

BXCH BIN 16-bit block data exchange

The BXCH instruction exchanges BIN 16-bit block data in d1 and BIN 16-bit block data in d2.



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

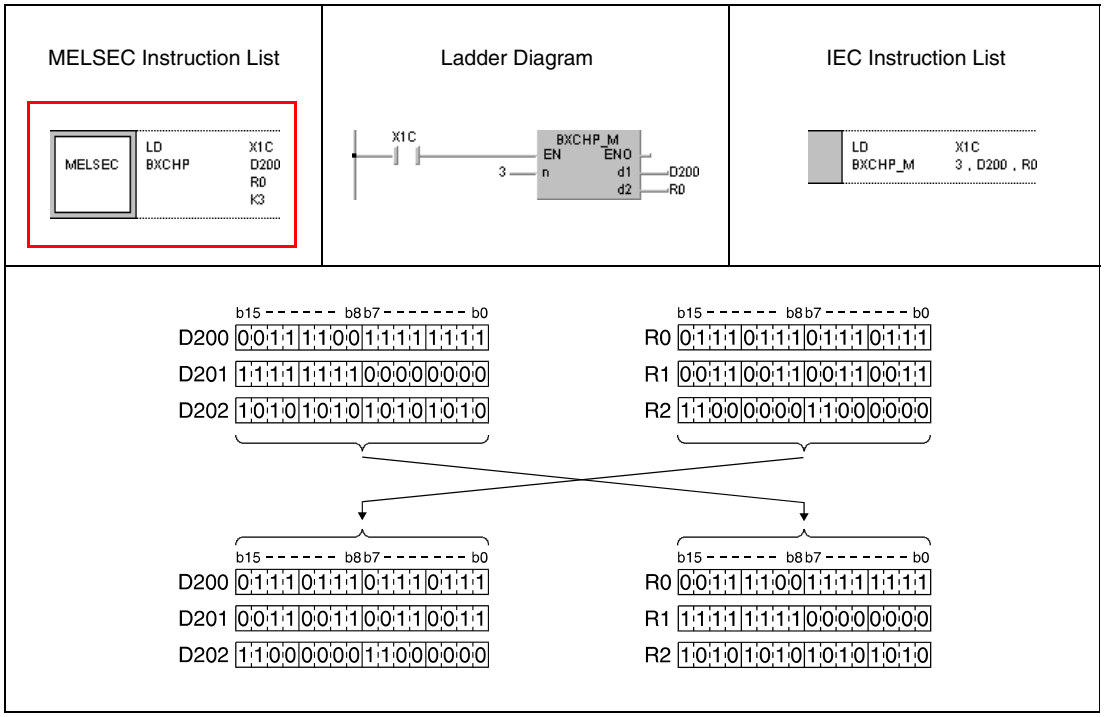
- The number of data blocks determined by n exceeds the storage device numbers designated by d1 and d2. (Error code 4101)
- The storage device numbers designated by d1and d2 overlap. (Error code 4101)

Program Example

BXCHP

With leading edge from X1C, the following program exchanges data blocks beginning from D200 and data blocks beginning from R0. The number of blocks (3) to be exchanged is determined by the constant K3.

The bit patterns show the structure of bits before and after the transfer.



6.4.11 SWAP, SWAPP

CPU

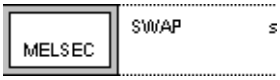
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

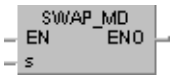
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	—	—

GX IEC Developer

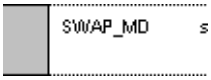
MELSEC Instruction List



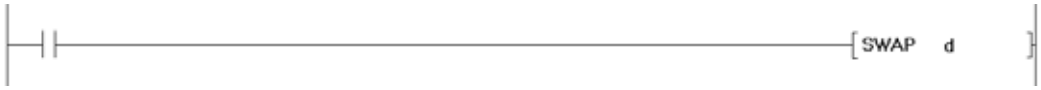
Ladder Diagram



IEC Instruction List



GX Works2



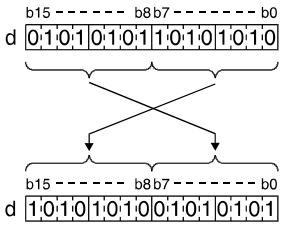
Variables

Set Data	Meaning	Data Type
s	First number of device storing data to be swapped.	BIN 16-bit

Functions **Upper and lower byte exchanges**

SWAP Upper and lower byte exchanges

The swap instruction exchanges the upper and lower 8 bits (upper and lower byte) of BIN 16-bit data in s.



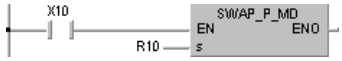
Program Example

SWAPP

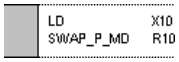
With leading edge from X10, the following program exchanges the upper and lower 8 bits in R10.

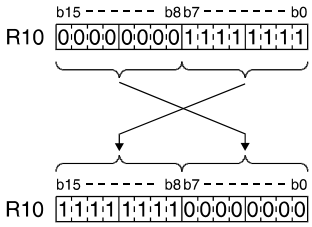
MELSEC Instruction List

Ladder Diagram



IEC Instruction List





6.5 Program branch instructions

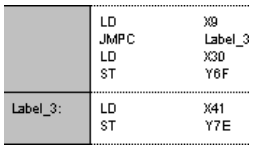
Program branch instructions include a jump destination.

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Conditional Jump	CJ	CJ_M
Conditional Jump from next Scan	SCJ	SCJ_M
Jump	JMP	JMP_M
Jump to End of Program	GOEND	GOEND_M

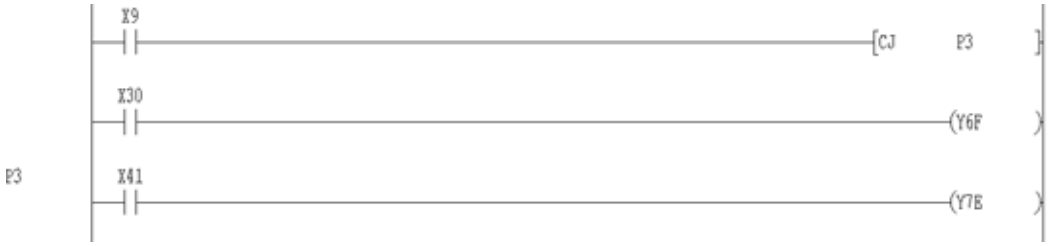
A jump destination is designated by a pointer P (GX Works2) or a label (GX IEC Developer).

For details on programming a label in GX IEC Developer see the Programming Manual for the GX IEC Developer.

GX IEC Developer



GX Works2



6.5.1 CJ, SCJ, JMP

CPU

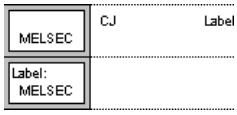
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

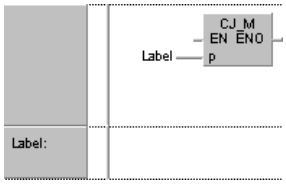
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				P
p	—	—	—	—	—	—	—	—	●

GX IEC Developer

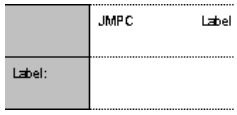
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
p	Jump destination	Pointer/Label

Functions **Jump instructions**

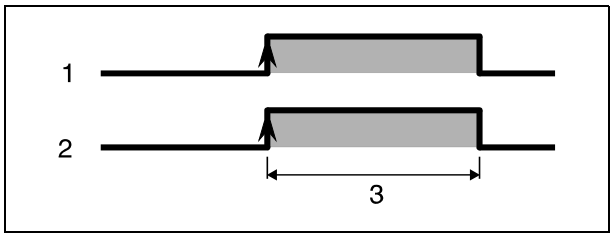
A jump instruction consists of a jump command CJ, SCJ, or JMP (**C**onditional **J**ump, **J**u**MP**) and a pointer (or label) P, designating the jump destination.

The pointer (label) number has to range within P(Label)0 and P(Label)4095. A jump destination P(Label)xx can be freely placed in a program.

CJ Conditional jump

Executes the program specified by the pointer number within the same program file, when the execution command is ON.

When the execution command is OFF, the program at the next step is executed.

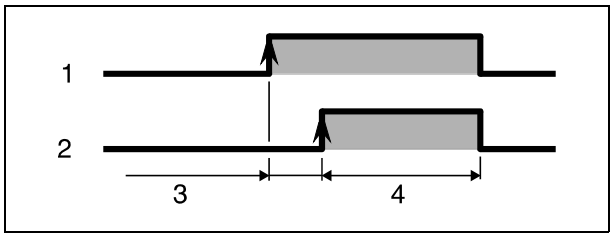


[1] Input condition
[2] CJ instruction
[3] Executed each scan

SCJ Conditional jump from next program scan

Executes the program specified by the pointer number within the same program file starting with the scan immediately after turning from OFF to ON of the execution command.

When the execution command is OFF or turned from ON to OFF, the program at the next step is executed.



[1] Input condition
[2] SCJ instruction
[3] One scan
[4] Executed each scan

JMP Jump instruction

The jump instruction executes the part of a program designated by the jump destination within the same program file without any input condition (unconditional jump).

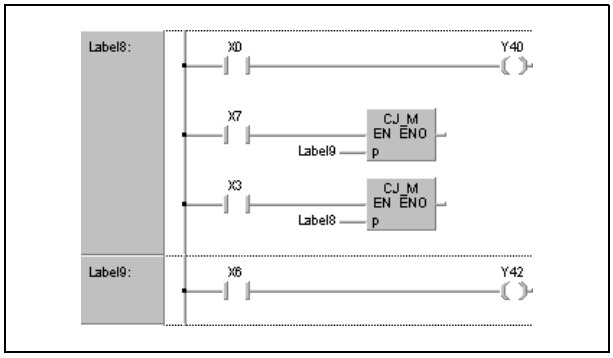
NOTE

If a set timer is skipped by a CJ, SCJ, or JMC instruction it will nevertheless keep its timing accurately.

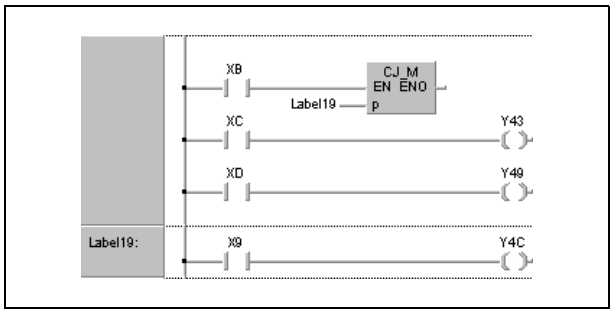
If an OUT instruction is skipped by a jump instruction, the condition of the output remains unchanged.

Executing a jump instruction shortens the scan time of a program in relation to the skipped program steps (see tables in appendices).

The CJ, SCJ, and JMP instruction can even jump back to a lower jump destination. However, a program must exit the program loop before the watchdog timer times out (the following program example exits the loop, when X7 is set).

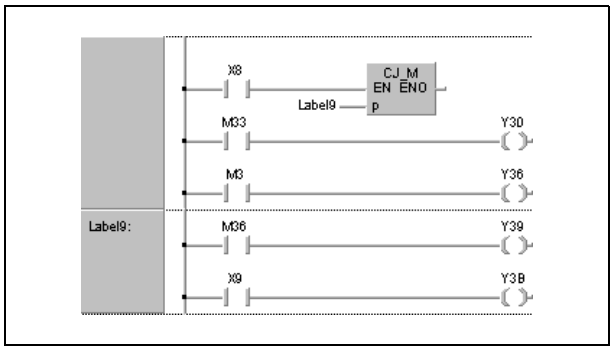


The condition of a device skipped by a jump instruction remains unchanged. This is illustrated by the following program example:



After XB is set, this program jumps to the jump destination Label19. The conditions of the outputs Y43 and Y49 even remain unchanged, if XC or XD are set or reset.

The jump destination (e.g. Label9) occupies one program step.



The CJ, SCJ, or JMP instruction only jumps to destinations within one single program.

If a jump destination is located within the skip range during a skip operation (operation skipping parts of a program), program execution proceeds from the first available address following the jump destination.

Operation Errors

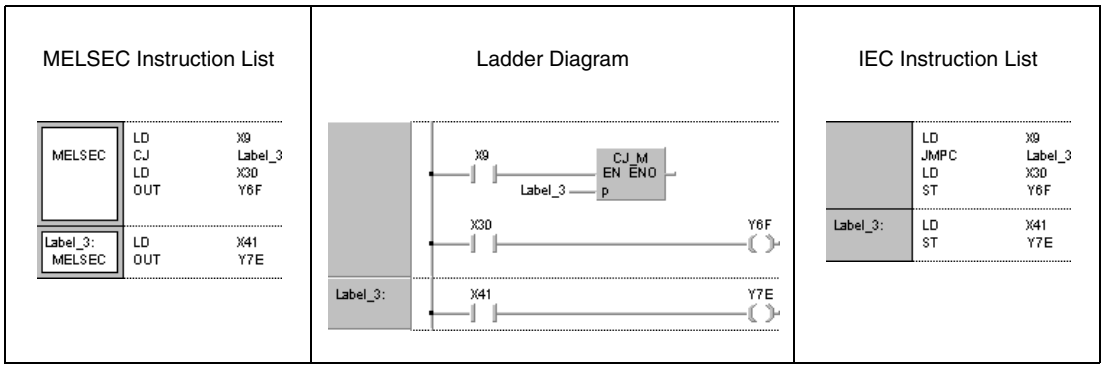
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- A common pointer has been designated. (Error code 4210)
- The jump destination of the jump instruction is not defined in a program (jump destination or pointer is missing). (Error code 4210)
- The jump destination is located after an END instruction. (Error code 4210)

Program Example 1

CJ

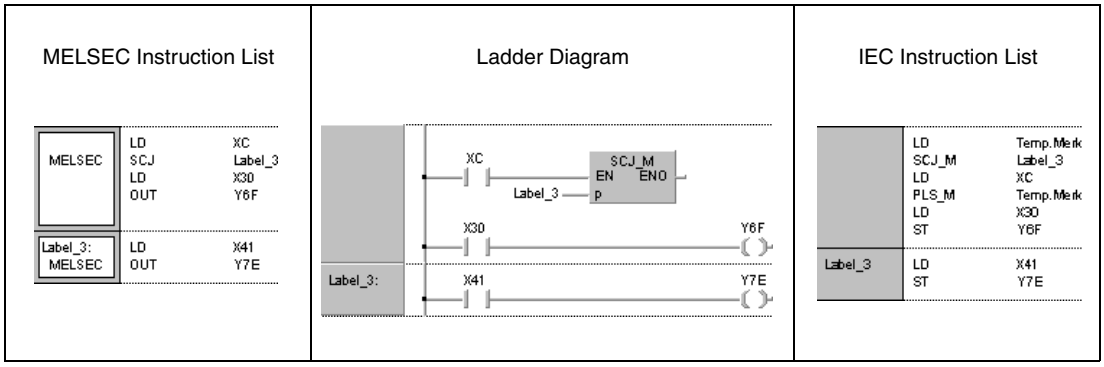
The following program jumps to the destination Label_3 when X9 is set.



Program Example 2

SCJ

The following program jumps to the destination Label_3 from the next scan when XC is set.



6.5.2 GOEND

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

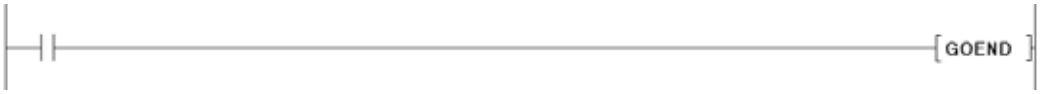
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

GX Works2



Variables

Set Data	Meaning	Data Type
—	—	—

Functions

GOEND Jump to the end of a program

The jump destination of the GOEND instruction is the FEND or END instruction of the program.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- A GOEND instruction was executed after a CALL or ECALL instruction and before a RET instruction. (Error code 4211)
- A GOEND instruction was executed after a FOR instruction and before a NEXT instruction. (Error code 4200)
- A GOEND instruction was executed during an interrupt program but before an IRET instruction. (Error code 4221)
- A GOEND instruction was executed between a CHKCIR and a CHKEND instruction. (Error code 4230)
- A GOEND instruction was executed between an IX and an IXEND instruction. (Error code 4231)

Program Example

GOEND

The following program jumps to the END instruction when data in D0 is negative.

MELSEC Instruction List

Ladder Diagram

IEC Instruction List

```
LD      TRUE
LD_LT_M D0 , K0
GOEND_M
```

6.6 Program execution control instructions

Program execution control instructions invoke interrupt routines. The interrupts can be enabled or disabled individually or via bit patterns.

The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Interrupt disabled	DI	DI_M
Interrupt enabled	EI	EI_M
Bit pattern of execution conditions of interrupt programs	IMASK	IMASK_M
End of interrupt program	IRET	IRET_M

6.6.1 DI, EI, IMASK

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

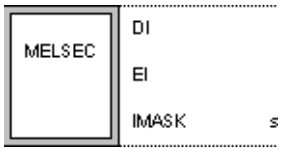
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●[1)]	●[1)]	—	●[1)]	●[1)]	—	—	—
—	—	—	—	—	—	—	—	—	—

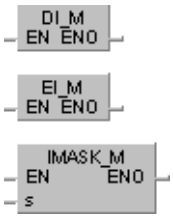
[1] IMASK instruction only

GX IEC Developer

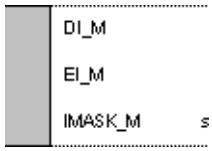
MELSEC Instruction List



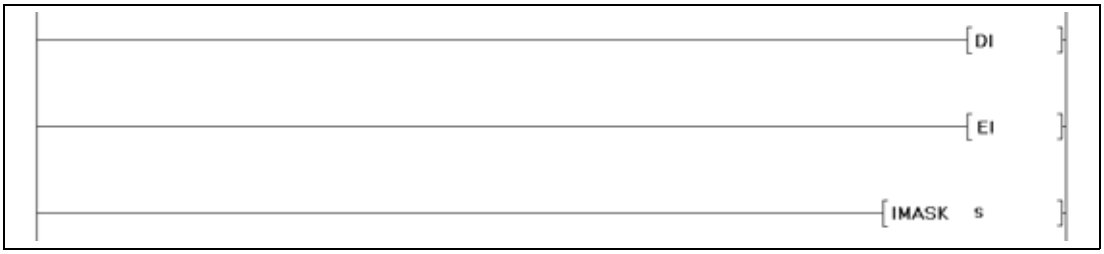
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	Bit pattern storing execution conditions of interrupts or first number of device storing bit pattern.	BIN 16-bit

Functions **Interrupt instructions**

An interrupt program is an inserted part of program (designated by an interrupt address Ixx) that can be invoked by an external interrupt signal. The interrupt program is executed depending on the EI/DI instruction.

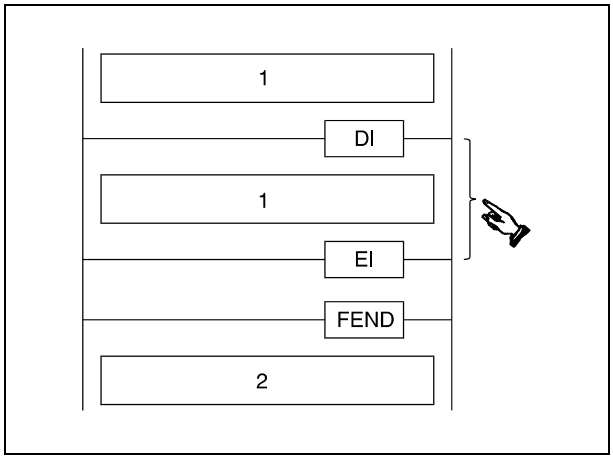
DI Disable interrupt

The DI instruction disables the execution of an interrupt program until an EI instruction is executed. The DI state is actice when power is turned ON or when the CPU module is reset.

EI Enable interrupt

The EI instruction enables invoking an interrupt program designated by an interrupt address Ixx, or enables the execution of an IMASK instruction.

Even though an interrupt condition might be generated between the DI and EI instructions, the interrupt program is suspended until the entire cycle from DI to EI has been processed. The following diagram illustrates such an execution:



[1] Sequence program

[2] Interrupt program

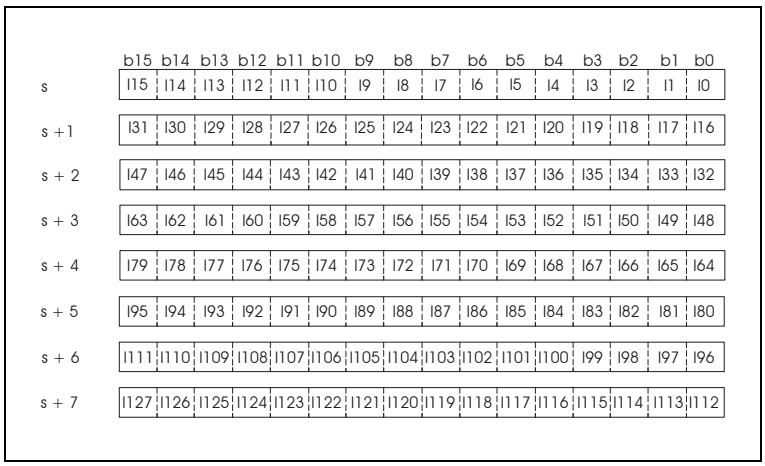
NOTE *The GX IEC Developer inserts the FEND instruction automatically. The event Ixx has to be allocated to a task.*

IMASK Bit pattern of execution conditions of interrupt programs

In the bit pattern designated by s a particular interrupt address is allocated to each bit. The condition of each bit determines whether the allocated interrupt can be executed. If the bit is reset (0), the interrupt program cannot be executed. If the bit is set (1), the interrupt program will be executed.

System Q CPU (Basic Model QCPU)

The allocation of bits in s through s+7 to the corresponding interrupt addresses is shown below:



When the power supply of the CPU is switched on or when the CPU has been reset, the execution of interrupt programs I0 through I31, I48 to I127 is enabled.

The bit patterns designated by s through s+2 are stored in the special registers SD715 through SD717. The bit patterns designated by s+3 through s+7 are stored in the special registers SD781 through SD785.

The bit patterns are designated as s through s+7 successively although the special registers are separated (SD715 through SD717 and SD781 through SD785).

System Q CPU (other than Basic Model QCPU) and L-series CPU

The allocation of bits in s through s+15 to the corresponding interrupt addresses is shown below:

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
s	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
s +1	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
s + 2	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
s + 3	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48
s + 4	I79	I78	I77	I76	I75	I74	I73	I72	I71	I70	I69	I68	I67	I66	I65	I64
s + 5	I95	I94	I93	I92	I91	I90	I89	I88	I87	I86	I85	I84	I83	I82	I81	I80
s + 6	I111	I110	I109	I108	I107	I106	I105	I104	I103	I102	I101	I100	I99	I98	I97	I96
s + 7	I127	I126	I125	I124	I123	I122	I121	I120	I119	I118	I117	I116	I115	I114	I113	I112
s + 8	I143	I142	I141	I140	I139	I138	I137	I136	I135	I134	I133	I132	I131	I130	I129	I128
s + 9	I159	I158	I157	I156	I155	I154	I153	I152	I151	I150	I149	I148	I147	I146	I145	I144
s +10	I175	I174	I173	I172	I171	I170	I169	I168	I167	I166	I165	I164	I163	I162	I161	I160
s + 11	I191	I190	I189	I188	I187	I186	I185	I184	I183	I182	I181	I130	I129	I128	I127	I126
s + 12	I207	I206	I205	I204	I203	I202	I201	I200	I199	I198	I197	I196	I195	I194	I193	I192
s + 13	I223	I222	I221	I220	I219	I218	I217	I216	I215	I214	I213	I212	I211	I210	I209	I208
s + 14	I239	I238	I237	I236	I235	I234	I233	I232	I231	I230	I229	I228	I227	I226	I225	I224
s + 15	I255	I254	I253	I252	I251	I250	I249	I248	I247	I246	I245	I244	I243	I242	I241	I240

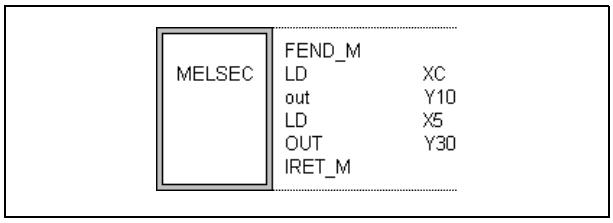
When the power supply of the CPU is switched on or when the CPU has been reset with the RUN/STOP switch, the execution of interrupt programs are as follows:

- High Performance model QCPU, Process CPU, and Redundant CPU
  Execution of interrupt programs I0 to I31 and I48 to I255 is enabled, and execution of interrupt programs I32 to I47 is disabled.
- Universal model QCPU and LCPU
  Execution of interrupt programs I0 to I31 and I45 to I255 is enabled, and execution of interrupt programs I32 to I44 is disabled.

The bit patterns designated by s through s+2 are stored in the special registers SD715 through SD717. The bit patterns designated by s+3 through s+15 are stored in the special registers SD781 through SD793.

Although the special registers are separated (SD715 through SD717 and SD781 through SD793), the bit patterns are designated as s through s+15 successively.

NOTES *The interrupt address (interrupt pointer) designating the interrupt program occupies one program step.*



With the GX Works2 or with the GX IEC Developer in MELSEC mode the instructions FEND and IRET have to be programmed by the user.

Alternatively to the MELSEC editor the IEC editor can be used. The interrupt is allocated to a task and the FEND and IRET instructions are placed automatically by the compiler of the GX IEC Developer MEDOC (see program example).

For the information on interrupt conditions, link direct devices, refer to the QnUCPU User's Manual(Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manuall(Function Explanation, Program Fundamentals).

During the execution of an interrupt program the DI status is internally set, so that no other interrupt program can be executed simultaneously. Another interrupt program can only be invoked after setting an EI instruction.

If an EI or DI instruction is placed within an MC instruction, the EI or DI instruction is executed without regard to the MC instruction.

Program Example

EI, DI, IMASK (GX IEC Developer)

The following program enables the execution of an interrupt program, if X0 is set (1). If X0 is reset (0), the execution of the interrupt program is disabled.

The lower diagram shows the tasks to be programmed in the IEC mode. These tasks invoke the interrupt programs I1 and I2.

Interrupt_1 (I1) and Interrupt_2 (I2) are interrupt programs. The IRET instruction does not need to be programmed because it is placed automatically by the compiler of the GX IEC Developer.

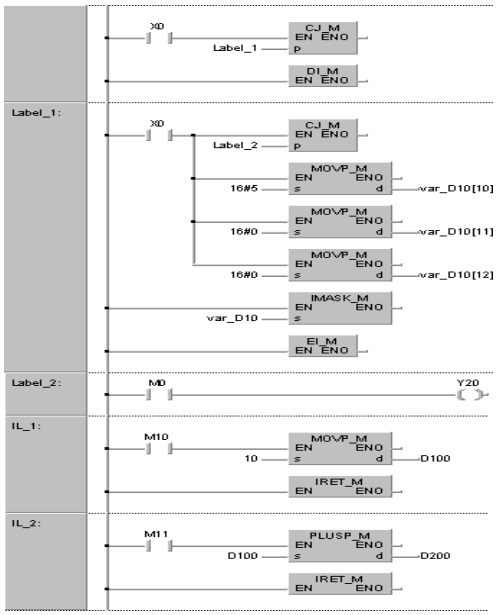
MELSEC Instruction List

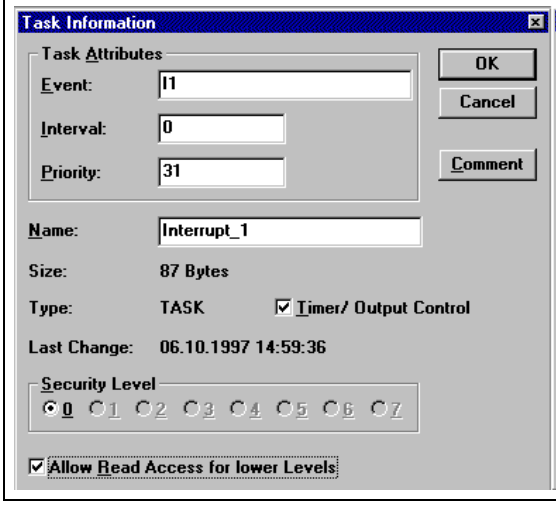
```
MELSEC    LD    X0
          CJ    Label_1
          DI
Label_1:  LD    X0
MELSEC    CJ    Label_2
          LD    X0
          MOVP  H5
                D10
          MOVP  H0
                D11
          MOVP  H0
                D12
          IMASK D10
          EI
Label_2:  LD    M0
MELSEC    OUT   Y20
I_1:      LD    M10
MELSEC    MOVP  K10
                D100
          IRET
I_2:      LD    M11
MELSEC    +P    D100
                D200
          IRET
```

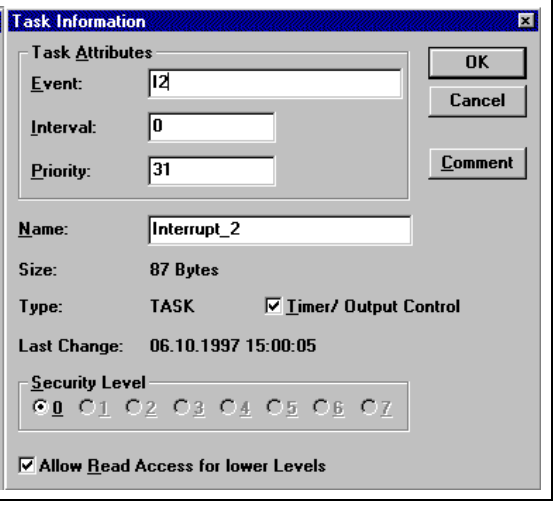
IEC Instruction List

```
          LD      X0
          JMPC    Label_1
          DI_M
Label_1:  LD      X0
          JMPC    Label_2
          LD      X0
          MOVP_M  16#6 , var_D10[10]
          MOVP_M  16#0 , var_D10[11]
          MOVP_M  16#0 , var_D10[12]
          IMASK_M var_D10
          EI_M
Label_2:  LD      M0
          ST      Y20
          LD      M10
          MOVP_M  10 , D100
          IRET_M
          LD      M11
          PLUSP_M D100 , D200
          IRET_M
```

Ladder Diagram







NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

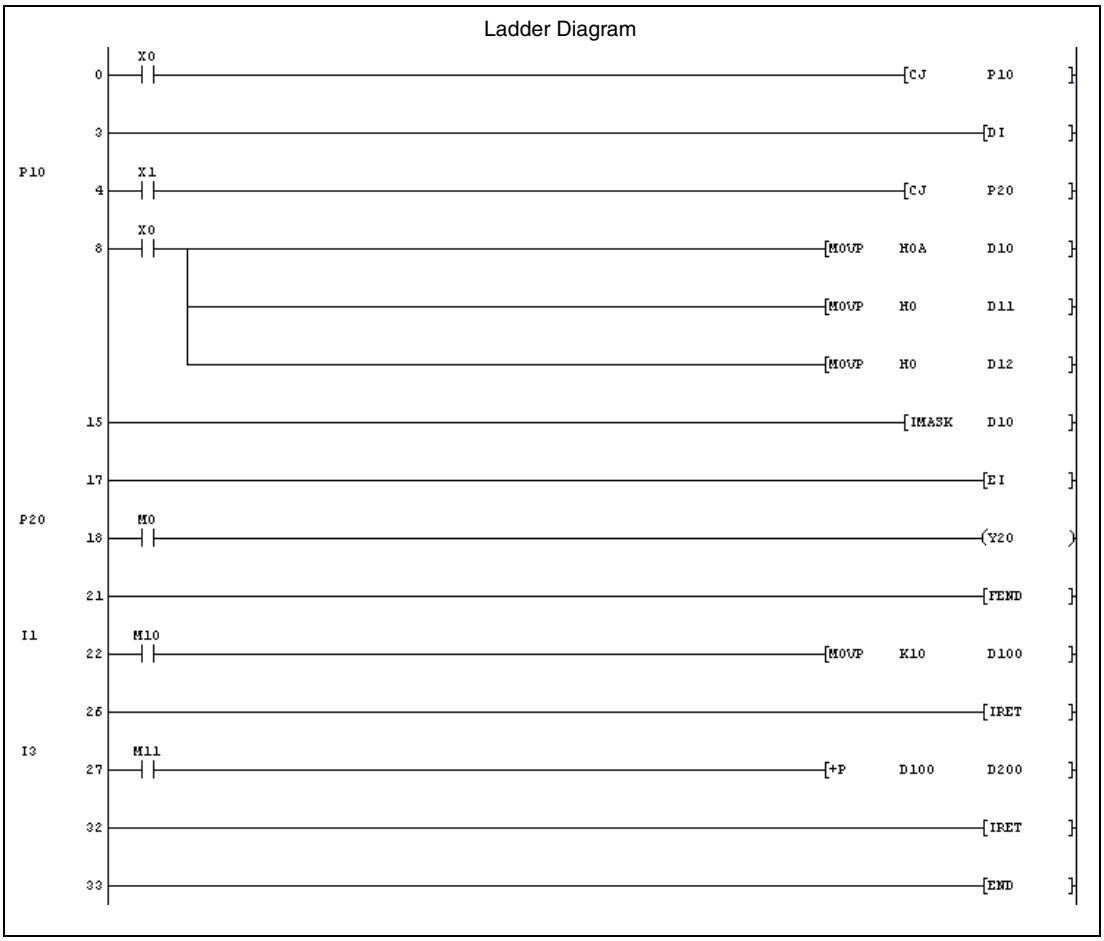
Program Example

EI, DI, IMASK (GX Works2)

In the following program, the execution of an interrupt program is enabled if X0 is set (1). When X0 is reset (0), the execution of the interrupt program is disabled.

I1 and I3 are interrupt programs.

Ladder Diagram



Instruction List

```
 0  LD      X0
 1  CJ      P10
 3  DI
 4  P10
 5  LD      X1
 6  CJ      P20
 8  LD      X0
 9  MOVP    H0A     D10
11  MOVP    H0      D11
13  MOVP    H0      D12
15  IMASK   D10
17  EI
18  P20
19  LD      M0
20  OUT     Y20
21  FEND
22  I1
23  LD      M10
24  MOVP    K10     D100
26  IRET
27  I3
28  LD      M11
29  +P      D100    D200
32  IRET
33  END
```

6.6.2 IRET

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

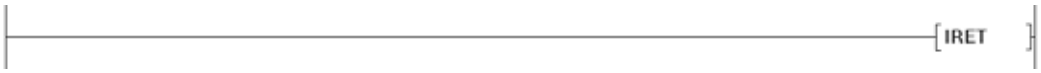
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

GX Works2



Variables

Set Data	Meaning	Data Type
—	—	—

NOTE *Within the IEC editors the IRET instruction is placed automatically in the program.*

Functions

Return from an interrupt program to the main program

IRET End of an interrupt program

The end of an interrupt program is indicated by an IRET instruction.

The main program is returned to after execution of the IRET instruction.

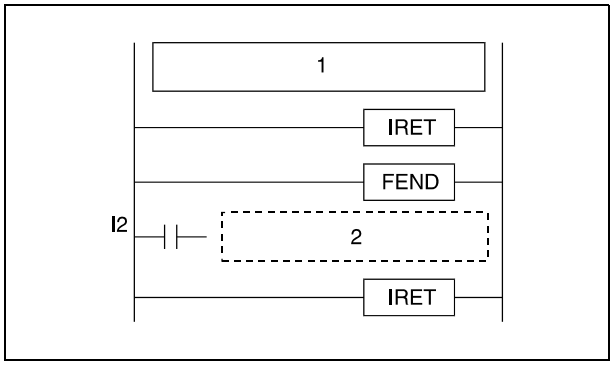
Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- There is no corresponding interrupt address for the interrupt call. (Error code 4220)
- If the IRET instruction is placed prior to an interrupt program, the CPU quits processing at that point. (Error code 4223)
- An END, FEND, GOEND, or STOP instruction was placed between an interrupt call and an IRET instruction.
- The IRET instruction was executed during the fixed scan execution type program. (For the Universal model QCPU, LCPU) (Error code 4223)

NOTE

The following example shows a programming error!



[1] Sequence program

[2] Interrupt program

Program Example

For the application of an IRET instruction in a program refer to the program examples for the EI, DI, and IMASK instructions (refer to section 6.6.1).

6.7 Link refresh instructions

Link refresh instructions refresh data at input/output interfaces. The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
I/O partial refresh	RFS	RFS_M
	RFSP	RFSP_M

6.7.1 RFS, RFSP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

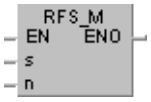
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●[1)]	—	—	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

[1] X and Y only

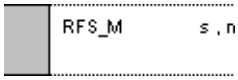
GX IEC Developer

MELSEC Instruction List

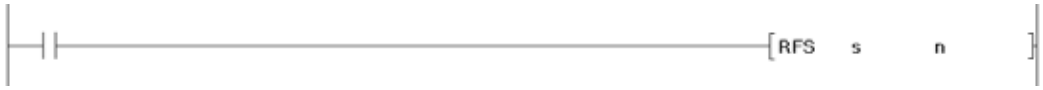
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	First number of I/O device to be refreshed	Bit
n	Number of I/O bits to be refreshed	BIN 16-bit

Functions

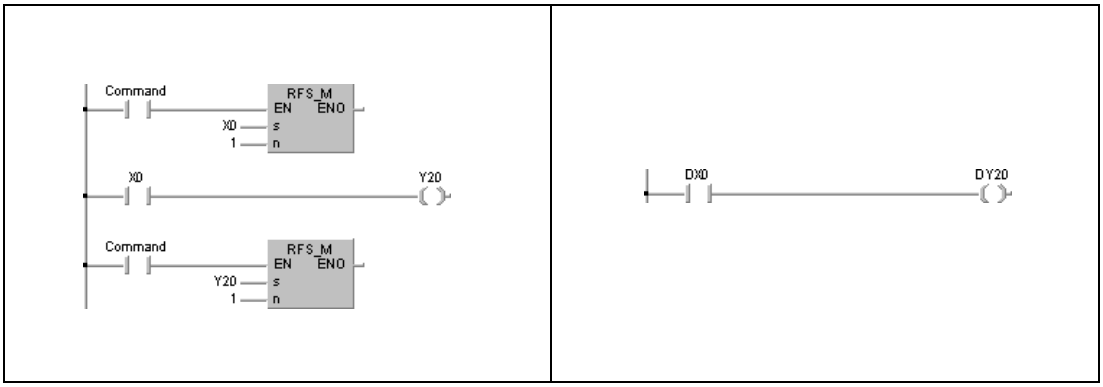
I/O partial refresh

RFS Refresh instruction

The RFS instruction refreshes the inputs and outputs of the designated range of I/O devices during one program scan. It reads data from an external source or writes data to an output module.

Data is read from an external source or written to an external output module in a batch after executing an END instruction. Therefore, a pulse signal cannot be output during one program scan. When the I/O refresh instruction is executed, the inputs (X) or outputs (Y) of the corresponding device numbers are refreshed forcibly midway through program execution. Thus, even pulse signals can be output.

If direct access inputs/outputs (DX/DY) are used, the inputs (X) and outputs (Y) are refreshed bit by bit.



The program example on the left refreshes the input X0 and the output Y20 via an RFS instruction.

The program example on the right performs the same functions via DX and DY without a refresh instruction.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of points determined by n exceeds the input/output device range. (Error code 4101)

Program Example

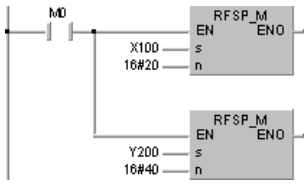
RFSP

With leading edge from M0, the following program refreshes the inputs X100 through X11F and the outputs Y200 through Y23F.

MELSEC Instruction List

```
MELSEC
LD    M0
RFSP  X100
      H20
RFSP  Y200
      H40
```

Ladder Diagram



IEC Instruction List

```
LD      M0
RFSP_M  X100, 16#20
RFSP_M  Y200, 16#40
```

6.8 Other convenient instructions

The instructions in the following table support programming of special timers and special counters, pulse counters and pulse outputs. Also included are instructions for positioning rotary tables and for building input matrices.

The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
1-Phase Input count-up/-down Counter	UDCNT1	UDCNT1_M
2-Phase Input count-up/-down Counter	UDCNT2	UDCNT2_M
Programmable (teaching) Timer	TTMR	TTMR_M
Special Function Timer	STMR	STMR_M
Positioning of Rotary Tables	ROTC	ROTC_M
Ramp Signal	RAMP	RAMP_M
Pulse Counter	SPD	SPD_M
Pulse Output with set Number of Outputs	PLSY	PLSY_M
Pulse Width Modulation	PWM	PWM_M
Building of Input Matrices	MTR	MTR_M

6.8.1 UDCNT1

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●[1)]	—	—	—	—	—	—	—	—
d	—	●[2,3)]	—	—	—	—	—	—	—
n	●[3)]	●[3)]	●[3)]	●	●	●	●	●	—

[1] X only

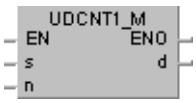
[2] C only

[3] Local devices and the file registers set for individual programs cannot be used

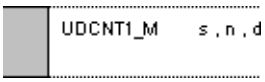
GX IEC Developer

MELSEC Instruction List

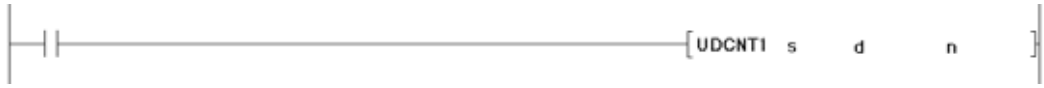
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	s+0: Input device number for count input (pulse signal, phase).	bit	Array [1..2] of BOOL
	s+1: Set count up or down 0 = count up 1 = count down		
d	Number of counter performing the UDCNT1 instruction.	BIN 16-bit (counter only)	ANY16
n	Setting	BIN 16-bit	ANY16

Functions

1-phase count-up/-down counter

UDCNT1 Counter instruction

When the input designated by s+0 (array_s [0]) changes from 0 to 1 the current count of the counter designated by d is updated. Consequently, only leading edges are counted.

The counting direction is determined by the status of the input designated by s+1 (array_s [1]):

– If the input condition is 0, the pulses of the input designated by s+0 (Array_s [0]) are added to the current count value.

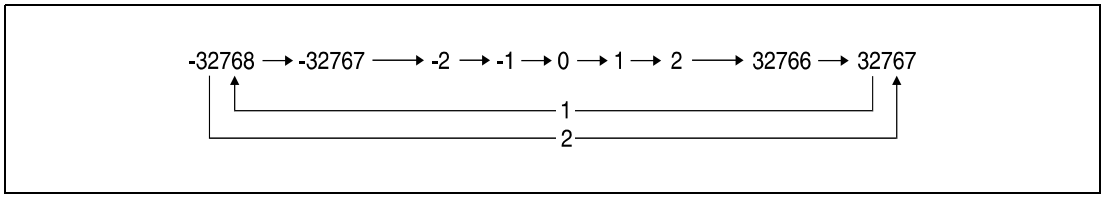
– If the input condition is 1, the pulses of the input designated by s+0 (Array_s [0]) are subtracted from the current count value.

The count processing performs as follows:

– When counting up, the counter contact designated by d is set (1), if the current count value is identical to the setting value in n. The counting process continues while the counter contact is set (see program example).

– When counting down, the counter contact is reset (0), if the current count value is identical to n-1 (see program example).

– The counter designated by d is a ring counter. If the count reads 32767 and is increased by 1, the counter jumps to -32768. If the count reads -32768 and is decreased by 1, the counter jumps to 32767. The following diagram illustrates ring counting:



[1] Counting up

[2] Counting down

The UDCNT1 instruction is started when the execution condition is set and stopped when the execution condition is reset. If the counter is started once again, it counts on from the value before it was stopped.

An RST instruction resets the counter designated by d and the according counter contact.

NOTE

The counting process of a UDCNT1 instruction is performed during a CPU interrupt (1 ms). For this reason only pulses with set/reset times over 1 ms can be counted accurately.

The setting value cannot be changed during the counting process (in this case the input designated by s+0 (Array_s [0]) is set). In order to change the setting, the input designated by s+0 (Array_s [0]) has to be reset.

Counters designated by a UDCNT1 instruction cannot be used by other instructions at the same time. In this case they would not return an accurate count.

The UDCNT1 instruction can be used as many as 6 times within all the programs being executed. The seventh and the subsequent UDCNT1 instructions are not processed.

Operation Errors

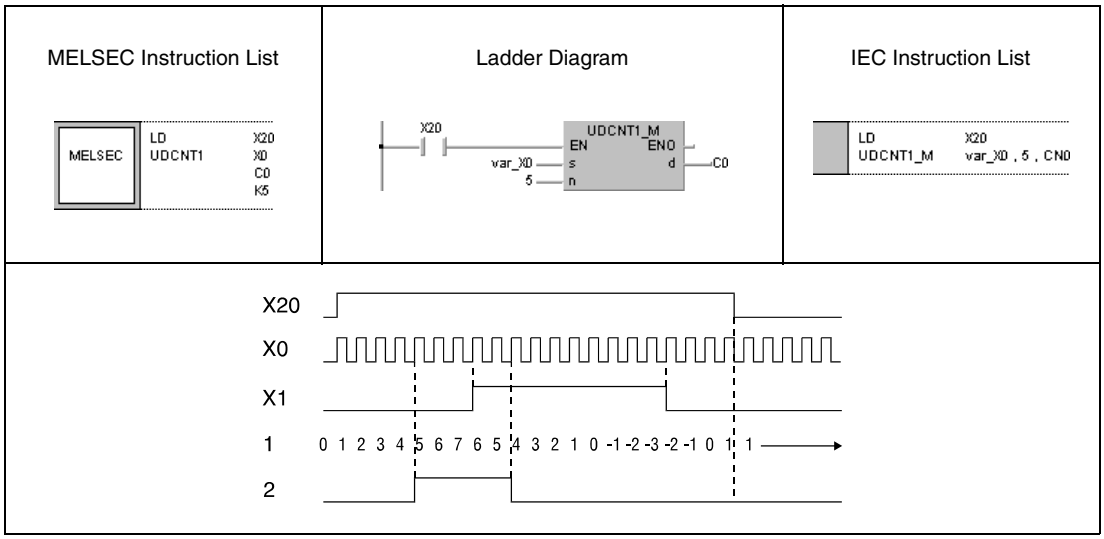
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by s exceeds the range of the corresponding device. (Error code 4101)

Program Example

UDCNT1

If X20 is set, the following program designates counter C0 (up/down counter) to count the number of leading edges from X0.



[1] Count

[2] Counter contact of counter C0

NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.8.2 UDCNT2

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●[1)]	—	—	—	—	—	—	—	—
d	—	●[2,3)]	—	—	—	—	—	—	—
n	●[3)]	●[3)]	●[3)]	●	●	●	●	●	—

[1] X only

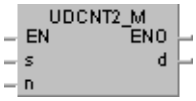
[2] C only

[3] Local devices and the file registers set for individual programs cannot be used

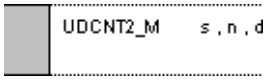
GX IEC Developer

MELSEC Instruction List

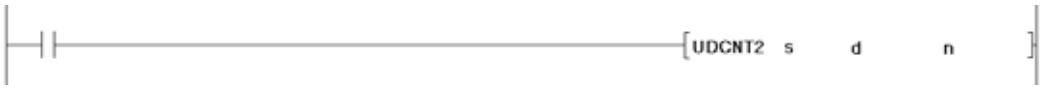
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	s+0: Input device number for count input (pulse signal, phase A)	Bit	Array [1..2] of BOOL
	s+1: Input device number of count input (pulse signal, phase B)		
d	Number of counter performing the UDCNT1 instruction	BIN 16-bit (counter only)	ANY16
n	Setting	BIN 16-bit	ANY16

Functions **2-phase count-up/-down counter**

UDCNT2 Counter instruction

The count of the counter designated by d is changed depending on the condition of the two inputs s+0 (array_s [0]) and s+1 (array_s [1]).

The direction of the count is determined as follows:

– If the input s+0 (array_s[0]) is set (1) and the input s+1 (array_s[1]) changes from 0 to 1 the current count is increased by 1.

– If the input s+0 (array_s[0]) is set (1) and the input s+1 (array_s[1]) changes from 1 to 0 the current count is decreased by 1.

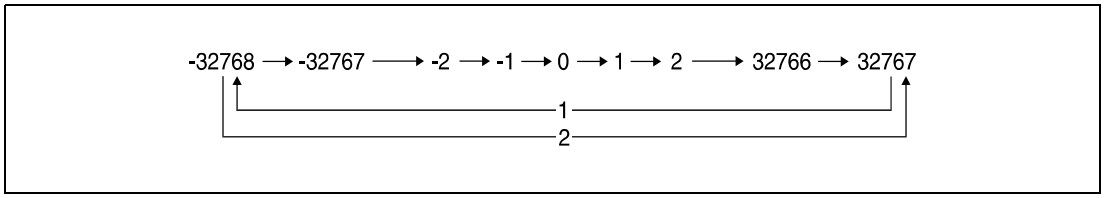
– If the input s+0 (array_s[0]) is reset (0) no counting operation is executed.

The count processing performs as follows:

– When counting up, the counter contact designated by d is set (1), if the current count value is identical to the setting value in n. The counting process continues while the counter contact is set (see program example).

– When counting down, the counter contact is reset (0), if the current count value is identical to n-1 (see program example).

– The counter designated by d is a ring counter. If the count reads 32767 and is increased by 1, the counter jumps to -32768. If the count reads -32768 and is decreased by 1, the counter jumps to 32767. The following diagram illustrates ring counting:



[1] Counting up

[2] Counting down

The UDCNT2 instruction is started when the execution condition is set and stopped when the execution condition is reset. If the counter is started once again, it counts on from the value before it was stopped.

An RST instruction resets the counter designated by d and the according counter contact.

NOTE

The counting process of a UDCNT2 instruction is performed during a CPU interrupt (1 ms). For this reason only pulses with set/reset times over 1 ms can be counted accurately.

The setting value cannot be changed during the counting process (-> the input designated by s+0 (Array_s [0]) is set). In order to change the setting, the input designated by s+0 (Array_s [0]) has to be reset.

Counters designated by a UDCNT2 instruction cannot be used by other instructions at the same time. In this case they would not return an accurate count.

The UDCNT2 instruction can be used as many as 5 times within all the programs being executed. The sixth and the subsequent UDCNT2 instructions are not processed.

Operation Errors

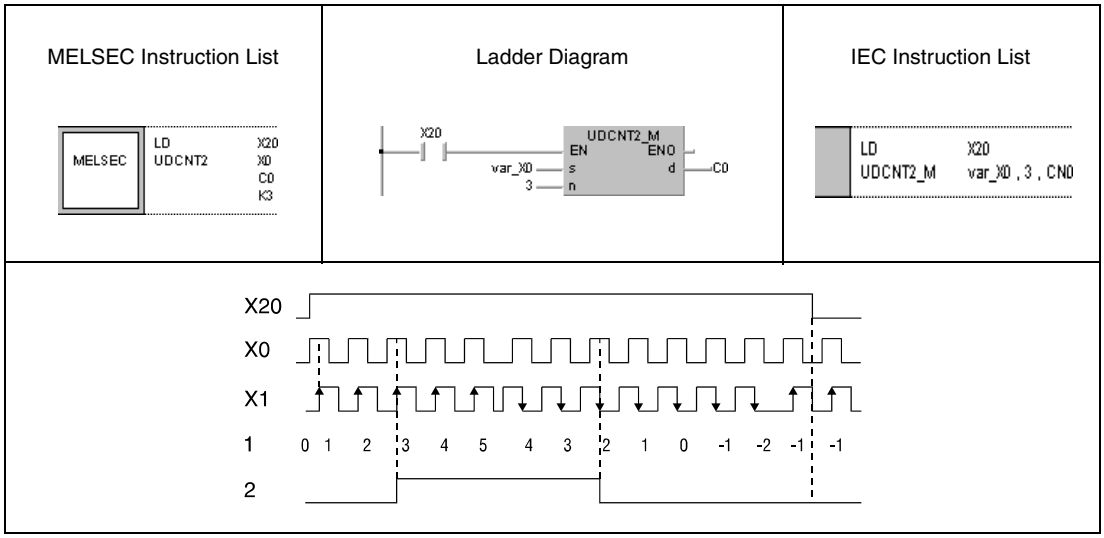
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by s exceeds the range of the corresponding device. (Error code 4101)

Program Example

UDCNT2

If X20 is set, the following program designates counter C0. The count and the count direction (up/down) depend on the conditions of X0 and X1.



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.8.3 TTMR

CPU

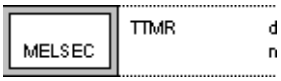
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

Devices

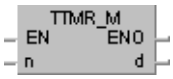
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	—	●	●	—	—	—	—	—	—
n	—	●	●	●	●	●	●	●	—

GX IEC Developer

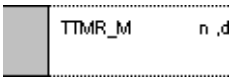
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
d	d+0: Device storing measurement value.	BIN 16-bit	Array [1..2] of ANY16
	d+1: For internal use by the CPU.		
n	Measurement value multiplier		ANY16

Functions

Programmable (teaching) Timer

TTMR Timer instruction

A timer programmed via the TTMR instruction measures the time of an input signal in seconds. The measurement value is multiplied with n and stored in d (array_d [0]+[1]).

With leading edge from the input the devices d+0 (array_d [0]) and d+1 (array_d [1]) are cleared.

The multipliers designated by n are as follows:

n = 0, multiplier 1

n = 1, multiplier 10

n = 2, multiplier 100

No processing is performed when the value specified by "n" is other than 0 to 2.

NOTE

Time measurement is performed during the execution of a TTMR instruction. Applying a JMP instruction or a similar instruction to the TTMR instruction causes inaccurate time measurement.

The multiplier n must not be changed during the execution of a TTMR instruction. A change would cause inaccurate measurement.

The TTMR instruction can also be used in low speed type programs.

The device designated by d+1 (array_d [1]) is used by the CPU. A change would cause inaccurate measurement.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

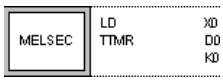
- The device specified by d exceeds the range of the corresponding device. (For the Universal model QCPU, LCPU) (Error code 4101)

Program Example

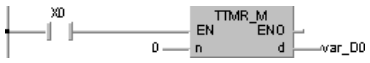
TTMR

If X0 is set, the following program measures the time in seconds (n = 0, multiplier = 1). The result is stored in D0.

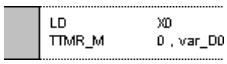
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.8.4 STMR

CPU

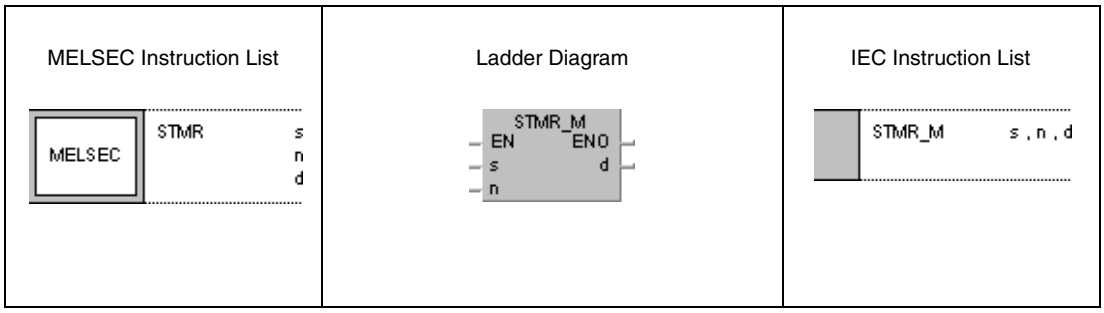
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

Devices

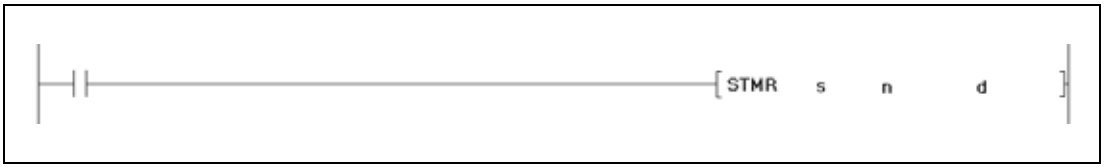
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
s	—	●[1)]	—	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—
d	—	—	—	—	—	—	—	—	—

[1] Can only be used by timer (T) data.

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	Number of timer	BIN 16-bit (timer only)	ANY16
n	Time setting	BIN 16-bit	ANY16
d	d+0: OFF delay timer output	Bit	Array [1..4] of BOOL
	d+1: One shot timer output after OFF (Set by trailing edge)		
	d+2: One shot timer output after ON (Set by leading edge)		
	d+3: ON delay and OFF delay timer output		

Functions **Special function timer**

STMR **Timer instruction for low speed timers**

The STMR instruction uses outputs designated by d+0 through d+3 (array_d [0] through array_d [3]) to perform four different timer functions:

- OFF delay timer output (d+0) (array_d [0])

  The output designated by d+0 (array_d [0]) is set (1) with leading edge from the execution condition. With trailing edge from the execution condition and after a period of time designated by n the output is reset (0) again.

- One shot timer output after OFF (Set by trailing edge, d+1 (array_d [1]))

  The output designated by d+1 (array_d [1]) is set (1) with trailing edge from the execution condition. After a period of time designated by n or with leading edge from the execution condition the output is reset (0) again.

- One shot timer output after ON (Set by leading edge, d+2 (array_d [2]))

  The output designated by d+2 (array_d [2]) is set (1) with leading edge from the execution condition. After a period of time designated by n or with trailing edge from the execution condition the output is reset (0) again.

- ON delay and OFF delay timer output (d+3 (Array [3]))

  The output designated by d+3 (array_d [3]) is set (1) with trailing edge from the timer coil. This corresponds to an ON delay time designated by n.

  The output d+3 is reset (0) when the amount of time designated by n has passed.

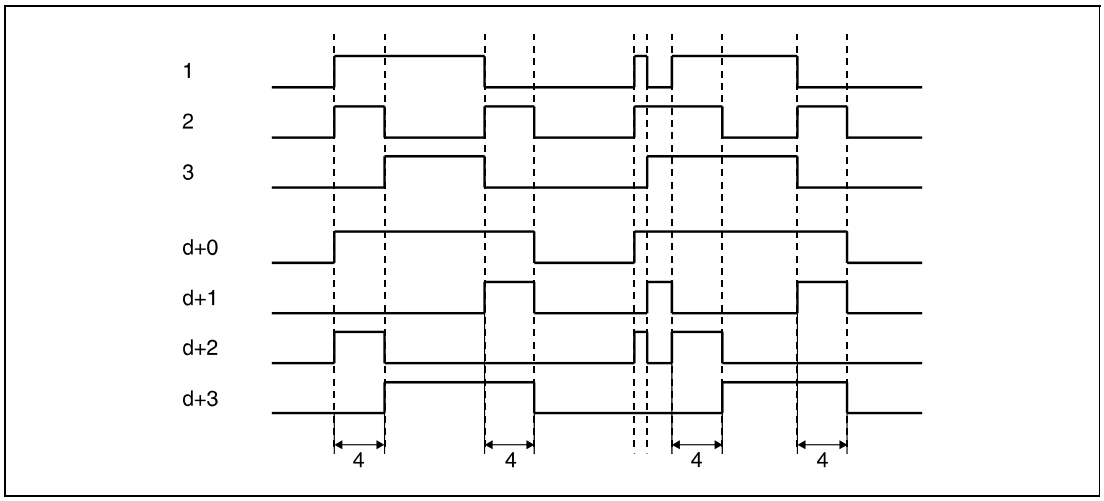
The timer coil of the timer designated by s is set (0) with leading edge from the execution condition and starts measuring the time designated by n.

The timer coil measures time until the measurement value matches the time setting n and then drops out.

If the execution condition is reset before the time setting n has passed, the timer coil remains set and time measurement is suspended at that point.

If the execution condition is set again the measurement value is cleared to 0 and time measurement starts again.

The timer contact designated by s is either set by trailing edge from the execution condition and set timer coil or by trailing edge from the timer coil and set execution condition. The timer contact is reset by trailing edge from the execution condition and reset timer coil. The timer contact is supplied for CPU internal use only.



[1] Execution condition

[2] Timer coil designated by s

[3] Timer contact designated by s

[4] Time setting n

Time measurement is performed during the execution of an STMR instruction. Applying a JMP instruction or a similar instruction to the STMR instruction causes inaccurate time measurement.

The realtime designated by d can be calculated by multiplying the time setting n with the time unit for low speed timers (default value = 100 ms).

The constant n has to range within 0 and 32767.

The timer designated by s cannot be used by an OUT instruction. If an OUT instruction and an STMR instruction use the same timer, the STMR instruction cannot be performed accurately.

Operation Errors

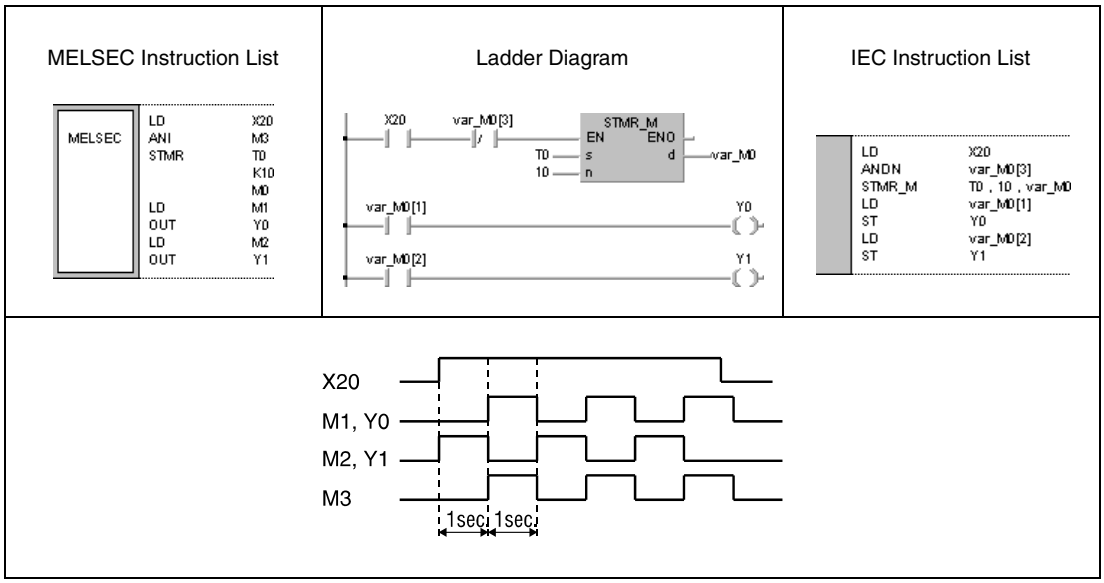
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by d exceeds the range of the corresponding device. (For the Universal model QCPU, LCPU) (Error code 4101)

Program Example

STMR

If X20 is set, the following program alternately sets the outputs Y0 and Y1 for 1 second each. The used timer is a 100 ms timer. The time period of 1 second is calculated by multiplying K10 with 100 ms.



NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.8.5 ROTC

CPU

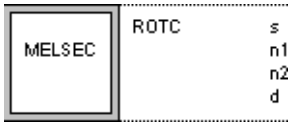
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

Devices

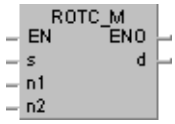
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
n1	●	●	●	●	●	●	●	●	—
n2	●	●	●	●	●	●	●	●	—
d	●	—	—	—	—	—	—	—	—

GX IEC Developer

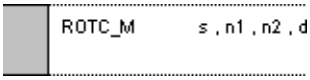
MELSEC Instruction List



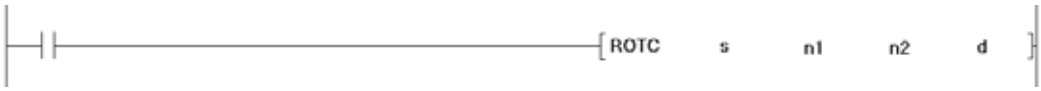
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	s+0: Measurement of table rpm (internal use only)	BIN 16-bit	Array [1..3] of ANY16
	s+1: Number of position		
	s+2: Number of sector		
n1	Number of sectors (divisions) on table (2 to 32767)		ANY16
n2	Number of low speed sectors (0 to n1)		ANY16
d	d+0: A-phase input signal	Bit	Array [1..8] of Bool
	d+1: B-phase input signal		
	d+2: Zero position detection input signal		
	d+3: High speed forward output signal (internal use only)		
	d+4: Low speed forward output signal (internal use only)		
	d+5: Stop output signal (internal use only)		
	d+6: High speed reverse output signal (internal use only)		
	d+7: Low speed reverse output signal (internal use only)		

Functions

Positioning instruction for rotary tables

ROTC Positioning instruction

The ROTC instruction rotates a sector designated by s+2 (array_s [2]) on a table with a specified number of sectors (divisions) designated by n1 to a specified position designated by s+1 (array_s [1]).

The positions and sectors on the rotary table are numbered counterclockwise.

The value in s+0 (array_s [0]) is internally used by the system to determine which sector is located where in relation to the zero position. This value must not be changed, otherwise the table will not be positioned accurately.

The value in n2 determines the number of sectors the table can be rotated by at low speed. This value must be equal or less than that designated by n1.

The A/B-phase inputs designated by d+0 (array_d [0]) and d+1 (array_d [1]) detect the direction of the rotation. Both inputs receive pulses. If the A-phase input d+0 (array_d [0]) is set, the direction of the rotation is determined by the pulse edge of the B-phase input d+1 (array_d [1]):

- If the B-phase is at leading edge at that moment the table rotates clockwise (to the right).
- If the B-phase is at trailing edge at that moment the table rotates counterclockwise (to the left).

The input designated by d+2 (array_d [2]) detects the zero position. This input is set, if sector 0 reaches position 0. If this input is set during the execution of a ROTC instruction, the value in s+0 (array_s [0]) is reset. For accurate positioning this value in s+0 (array_s [0]) should be reset before positioning via the ROTC instruction.

Data in d+3 (array_d [3]) through d+7 (array_d [7]) store output signals for operating the rotary table. Which output signal is set depends on the current operation result of the ROTC instruction.

If all operation results were 0 just before executing a ROTC instruction, the outputs designated by d+3 (array_d [3]) through d+7 (array_d [7]) are reset without positioning the table. After resetting the execution condition these outputs are reset either.

A ROTC instruction can only be executed once in a program. Repeated application within one program causes faulty operation of the instruction.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by s or d exceeds the range of the corresponding device. (For the Universal model QCPU, LCPU) (Error code 4101)

Program Example

ROTC

In the following program the contacts X0, X1 (incremental encoder), and X2 address the internal relays for detection of the rotating direction and zero position M0 (var_M0 array [0]) through M2 (var_M0 array [2]). The contact X2 is activated, if sector 0 is located at position 0 (zero position detection).

The rotary table shown below is divided into 10 sectors.

Which item (sector) will be moved to which station (position) has to be specified in D201 (var_D200 array [1]) and D202 (var_D200 array [2]) before the execution of the ROTC instruction.

Due to the value n1=10 the contact of the counter register outputs 10 pulses each rotation (division). The value n2=2 specifies the number of low speed divisions.

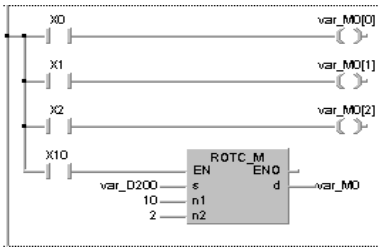
For example, if register D201 (var_D200 array [1]) stores the value 0 and register D202 (var_D200 [2]) stores the value 3, the rotary table moves item 3 (sector 3) to station 0 (position 0) travelling the shortest distance (clockwise). The sectors 1 through 3 rotate at low speed.

For an allocation of single registers and internal relays or array elements respectively to the corresponding functions see the table following the example.

MELSEC Instruction List

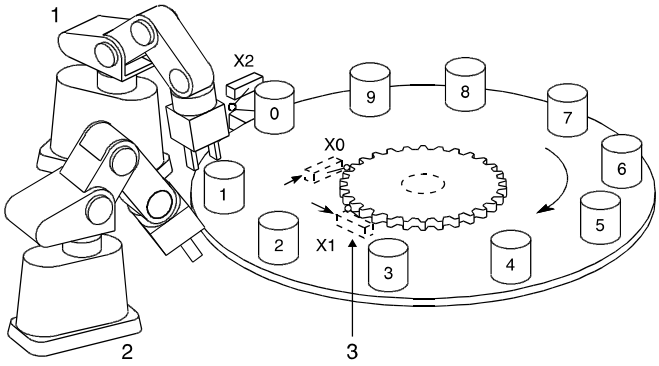
```
MELSEC
        LD    X0
        OUT   M0
        LD    X1
        OUT   M1
        LD    X2
        OUT   M2
        LD    X10
        ROTC  D200
              K10
              K2
              M0
```

Ladder Diagram



IEC Instruction List

```
LD    X0
ST    var_M0[0]
LD    X1
ST    var_M0[1]
LD    X2
ST    var_M0[2]
LD    X10
ROTC_M  var_D200 , 10 , 2 , var_M0
```



[1] Station 0 (position 0)

[2] Station 1 (position 1)

[3] Detection switch

Data register	Meaning	Remark
D200 (var_D200 Array [0])	Counter register	
D201 (var_D200 Array [1])	Position of station	These values are written to the data registers D201 (var_D200 array [1]) and D202 (var_D200 array [2]) via a MOV instruction.
D202 (var_D200 Array [2])	Position of item	
M0 (var_M0 Array [0])	A-phase signal	The internal relays M0 (var_M0 array [0]) through M2 (var_M0 array [2]) are addressed by the inputs X0 through X2 (see program example).
M1 (var_M0 Array [1])	B-phase signal	
M2 (var_M0 Array [2])	Zero position detection signal	
M3 (var_M0 Array [3])	High speed forward rotation	After X10 is set the ROTC instruction is activated and the internal relays M3 (var_M0 array [3]) through M7 (var_M0 array [7]) are assigned specified functions. After resetting X10 these internal relays are reset either.
M4 (var_M0 Array [4])	Low speed forward rotation	
M5 (var_M0 Array [5])	Stop signal	
M6 (var_M0 Array [6])	High speed reverse rotation	
M7 (var_M0 Array [7])	Low speed reverse rotation	

NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

6.8.6 RAMP

CPU

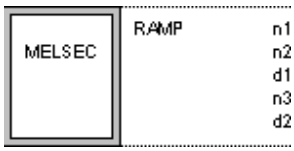
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

Devices

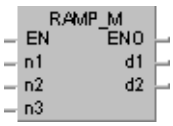
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
n1	●	●	●	●	●	●	●	●	—
n2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—
n3	●	●	●	●	●	●	●	●	—
d2	●	—	—	—	—	—	—	—	—

GX IEC Developer

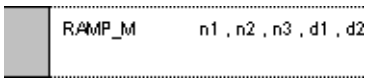
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
n1	Initial value of operation	BIN 16-bit	ANY16
n2	Final value of operation		ANY16
d1	(d1)+0: Device storing current value		Array [1..2] of ANY16
	(d1)+1: Device storing number of executed moves (internal use only)		
n3	Number of moves to be executed		ANY16
d2	(d2)+0: Bit to be set after completion	Bit	Array [1..2] of Bool
	(d2)+1: Bit determining storage of operation result		

Functions

Ramp signal

RAMP Instruction for changing the content of a device gradually

A RAMP instruction changes the content in (d1)+0 (array_d1 [0]) gradually from the initial value designated by n1 to the final value designated by n2.

The number of moves performing the gradual changes is designated by n3.

The number of moves already executed is stored in (d1)+1 (array_d1 [1]) for internal system use.

When the operation is completed the device designated by (d2)+0 (array_d2 [0]) is set.

The signal condition of the device (d2)+0 (array_d2 [0]) and the content of the device (d1)+0 (array_d1 [0]) depend on the signal condition of the device (d2)+1 (array_d2 [1]):

- If the device (d2)+1 (array_d2 [1]) is not set, the device (d2)+0 (array_d2 [0]) will be reset during the next scan and the RAMP instruction will begin a new move operation from the value currently stored in (d1)+0 (array_d1 [0]).
- If the device (d2)+1 (array_d2 [1]) is set, the device (d2)+0 (array_d2 [0]) remains set and the value in (d1)+0 (array_d1 [0]) is not changed (storage).

If the execution condition is reset during the operation, the content in (d1)+0 (array_d1 [0]) does not change. If the execution condition is set once again, the RAMP instruction changes the current content in (d1)+0 (array_d1 [0]) stored before the reset.

During the processing of the instruction the values in n1 and n2 must not be changed.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by d1 or d2 exceeds the range of the corresponding device. (For the Universal model QCPU, LCPU) (Error code 4101)

NOTE

When the digit specification of bit device is made to d1, the digit specification of bit device can only be used when the specification of digits is "K8".

Program Example

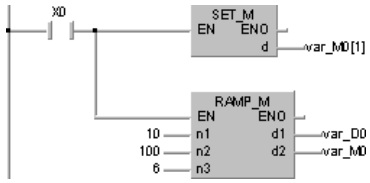
RAMP

The following program increases the content in D0 within 6 moves from 10 to 100 and stores the content in D0 when the operation is completed.

MELSEC Instruction List

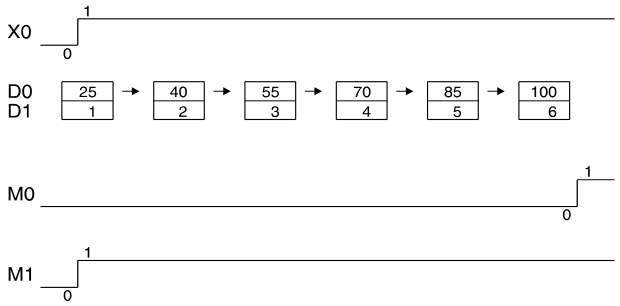
```
MELSEC
        LD      X0
        SET     M1
        RAMP    K10
                K100
                D0
                K6
                M0
```

Ladder Diagram



IEC Instruction List

```
        LD      X0
        SET_M   var_M0[1]
        RAMP_M  10 , 100 , 6 , var_D0 , var_M0
```



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

6.8.7 SPD

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

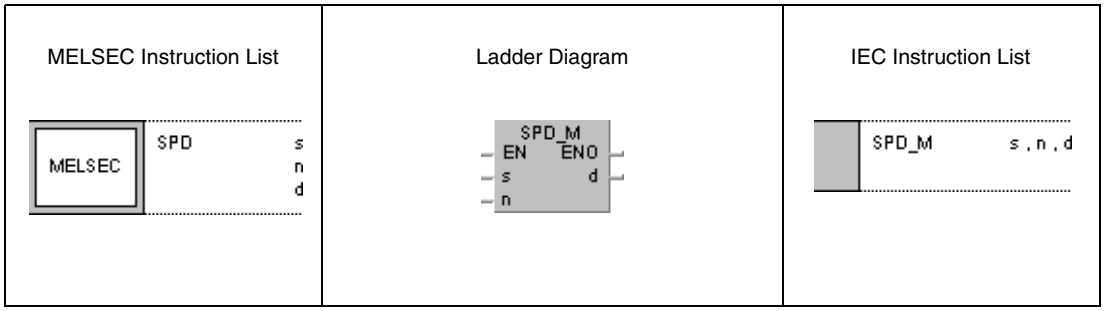
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
s	●[1)]	—	—	●	●	●	●	●	—
n	●[2)]	●[2)]	●[2)]	—	—	—	—	—	—
d	—	●[2)]	●[2)]	●	●	●	●	●	—

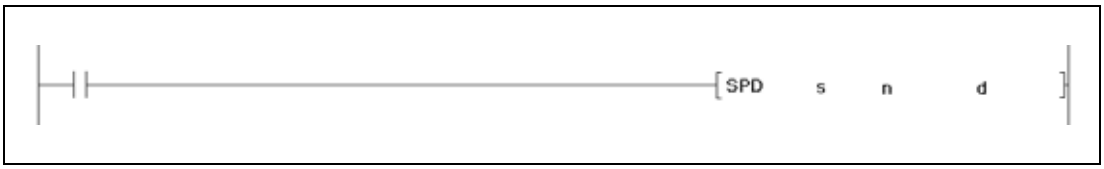
[1] Only X

[2] Local devices and the file registers set for individual programs cannot be used.

GX IEC Developer



GX Works2



Variables

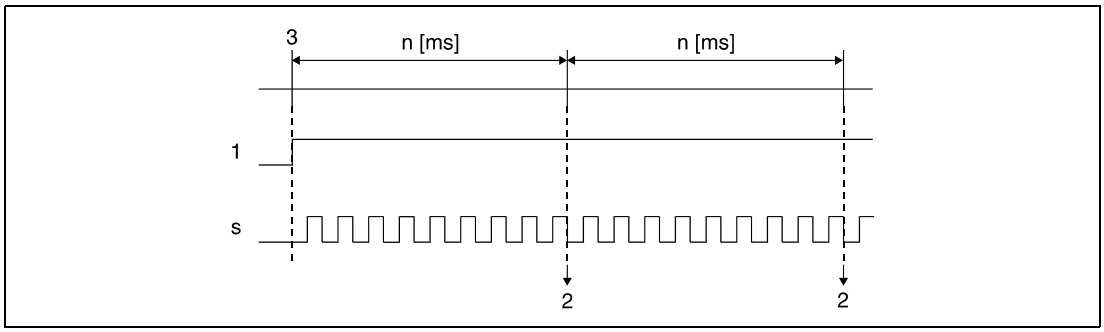
Set Data	Meaning	Data Type
s	Pulse input signal	Bit
n	Measurement time (unit: ms)	BIN 16-bit
d	First number of device storing measurement result	

Functions

Pulse density measurement

SPD Pulse density measurement

The SPD instruction counts pulses at the input designated by s for a period of time specified by n. The result of the measurement is stored in d.



[1] Execution condition.

[2] The result of the measurement is stored in d.

[3] Begin of measurement.

While the execution condition is set, the measurement begins again from 0 after the measurement time has passed. In order to stop the SPD measurement the execution condition has to be reset.

The SPD instruction stores the data from the designated devices in the CPU work area, and performs the current count operation during a 5 ms system interrupt. For this reason, the number of times the instruction can be used is limited. The SPD instructions exceeding this limit are not processed.

NOTES

The count processing for pulses used with the SPD instruction is conducted during an interrupt. Therefore, to count the pulses, it is necessary to provide their ON and OFF time as long as the interrupt time of the CPU or longer. The interrupt time is 1 ms.

When the High Performance model QCPU or Process CPU is used, the SPD instruction is not processed if n = 0.

The SPD instruction can be used as many as 6 times within all the programs being executed. The seventh and the subsequent SPD instructions are not processed.

While the measurement is in execution (while the command input is ON) by the SPD instruction, the setting value cannot be changed. Turn OFF the command input before changing the setting value.

Operation Errors

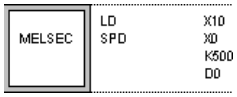
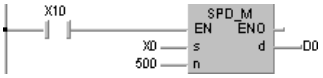
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by s exceeds the range of the corresponding device. (For the Universal model QCPU, LCPU) (Error code 4101)

Program Example

SPD

If X10 is set, the following program counts the pulses at X0 during a period of time of 500 ms. The result is stored in D0.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

6.8.8 PLSY

CPU

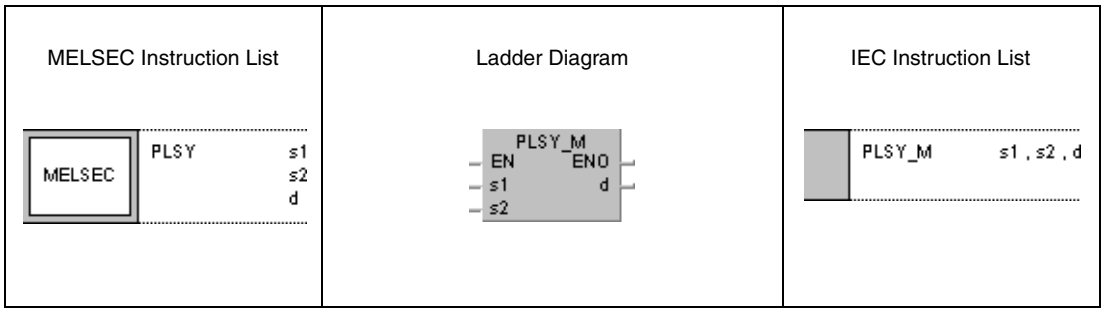
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

Devices

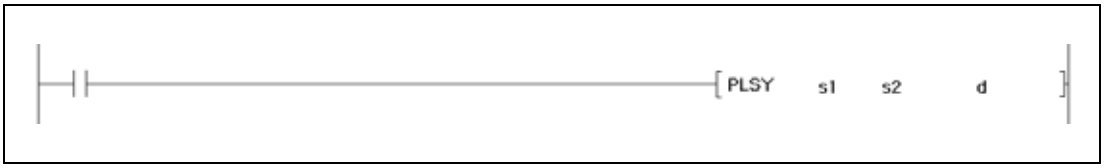
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d	●[1)]	—	—	—	—	—	—	—	—

[1] Y only

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
s1	Frequency or device storing pulse frequency setting.	BIN 16-bit
s2	Outputs count or the number of the device storing number of output pulses setting.	
d	Device storing output destination.	Bit

Functions

Pulse output with adjustable number of pulses

PLSY Pulse output instruction

The PLSY instruction outputs a number of pulses specified by s2 at a frequency specified by s1 to an output designated by d.

The frequency range in s1 can be specified from 1 to 100 Hz. If s1 is other than 1 to 100 Hz, the PLSY instruction will not be executed.

The number of output pulses in s2 can be specified from 0 to 65535 (0000H to FFFFH). If s2 is set to "0", pulses are continuously output.

Only outputs corresponding to the output module can be designated by d.

Pulse output begins with leading edge from the execution condition of the PLSY instruction. During pulse output the execution condition must not be reset. Resetting the execution condition suspends the pulse output.

NOTE

The PLSY instruction stores the data from the designated devices in the CPU work area, and and counting operation is processed as a system interrupt. The pulses that can be output must have longer ON and OFF times than the interrupt interval of the CPU module. The interrupt interval of individual modules is 1 ms.

Do not change the argument for the PLSY instruction during pulse output directed by the PLSY instruction (while the execution command is ON). To change the argument, turn OFF the execution command.

The PLSY instruction can be used only once in all programs executed by the CPU module. The second and the subsequent PLSY instructions are not processed.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

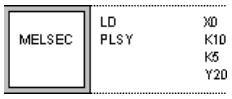
- The device specified by d exceeds the range of the corresponding device. (For the Universal model QCPU, LCPU) (Error code 4101)

Program Example

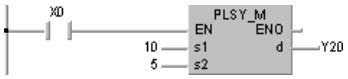
PLSY

If X0 is set, the following program outputs five 10 Hz pulses to Y20.

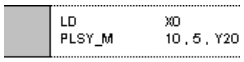
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



6.8.9 PWM

CPU

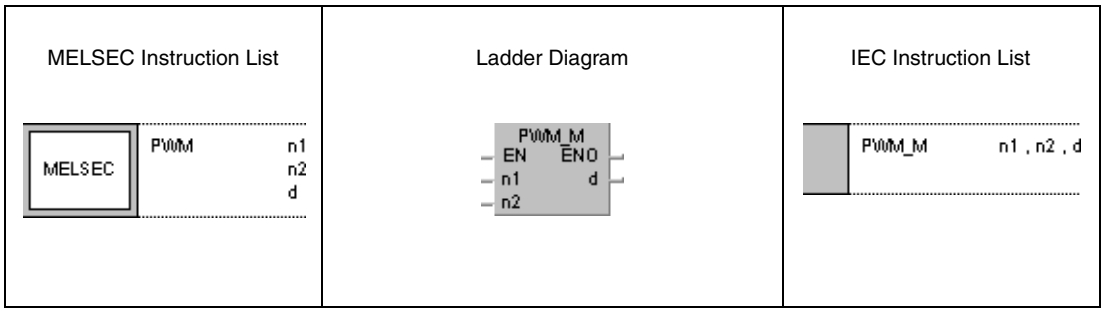
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

Devices

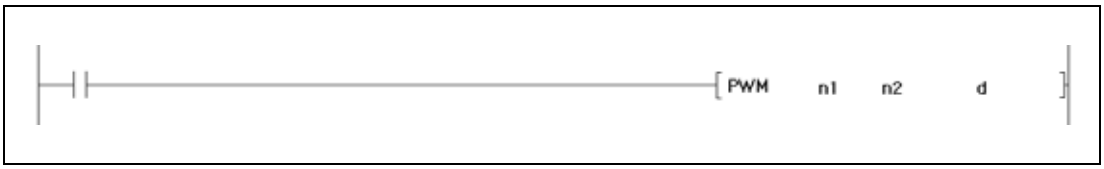
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
n1	●	●	●	●	●	●	●	●	—
n2	●	●	●	●	●	●	●	●	—
d	●[1)]	—	—	—	—	—	—	—	—

[1] Only Y

GX IEC Developer



GX Works2



Variables

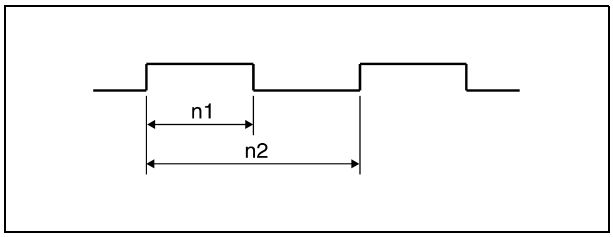
Set Data	Meaning	Data Type
n1	ON time or the number of device storing ON time setting.	BIN 16-bit
n2	Frequency or the number of device storing cycle time setting.	
d	Number of device storing output destination.	Bit

Functions

Pulse width modulation

PWM Modulation instruction

The PWM instruction outputs pulses at a cycle time specified by n2 and with an ON time specified by n1 to an output designated by d.



The times in n1and n2 can be specified from 1 to 65535 ms. The value set in n1 has to be less than that in n2.

NOTES

The PWM instruction registers the data from the designated devices in the work area of the CPU, and performs the current output operation during a system interrupt (1 ms).

For this reason, the PWM instruction can only be used once in a program.

The instruction is not processed in the following cases:

– When both n1 and n2 are 0

– When n2 is smaller or equal to n1

– When the PWM instruction is executed twice or more.

Do not change the argument for the PLSY instruction during pulse output directed by the PLSY instruction (while the execution command is ON). To change the argument, turn OFF the execution command.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

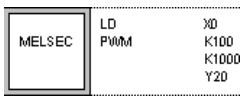
- The device specified by d exceeds the range of the corresponding device. (For the Universal model QCPU, LCPU) (Error code 4101)

Program Example

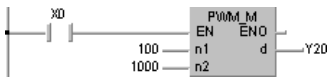
PWM

If X0 is set, the following program outputs pulses at a cycle time of 1 second and with an ON time of 100 ms to Y20.

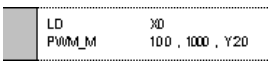
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



6.8.10 MTR

CPU

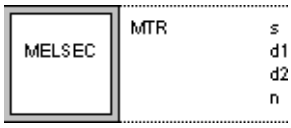
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●		●	●

Devices

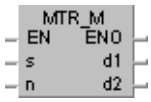
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
s	●	—	—	—	—	—	—	—	—
d1	●	—	—	—	—	—	—	—	—
d2	●	—	—	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

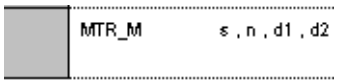
MELSEC Instruction List



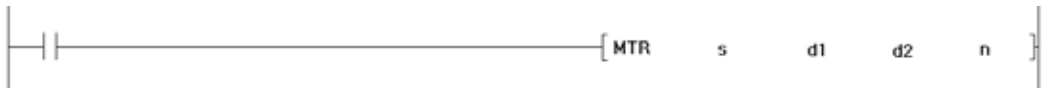
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	Initial input device.	Bit
d1	Initial output device.	
d2	First number of device storing matrix input data.	
n	Number of input rows.	BIN 16-bit

Functions

Building an input matrix

MTR Instruction for reading n data rows into an input matrix.

The MTR instruction reads the information of 16 bits (0/1) beginning from the device designated by s. The number of repetitions (rows) is designated by n. The conditions of read data are stored in the device designated by d2 onwards. This way, a matrix of 16 bits and n rows is built.

One row (16 bits) can be read each scan.

The reading process is continuously repeated from the first to the nth row.

Due to the format of the input matrix (16 bits x n rows) the device designated by d2 has to supply space for 16 bits x n rows either to store the data.

Each row is selected beginning with the output designated by d1. The corresponding output for each row of 16 bits to be read is set or reset by the system automatically. The number of outputs is identical with the number of rows. Thus, each single row can be addressed accurately by the system

The device numbers designated by s, d1, and d2 must be divisible by 16.

The number of rows n can be designated from 2 to 8.

Note, that the MTR instruction directly operates on current input and output data.

No processing is performed in the following cases:

- The device numbers designated by s, d1, and d2 are not divisible by 16.
- The device designated by s exceeds the current input range.
- The device designated by s exceeds the current output range.
- The matrix space 16 bits x n rows exceeds the relevant device range of d2.
- The value in n does not range within 2 and 8.

Operation Errors

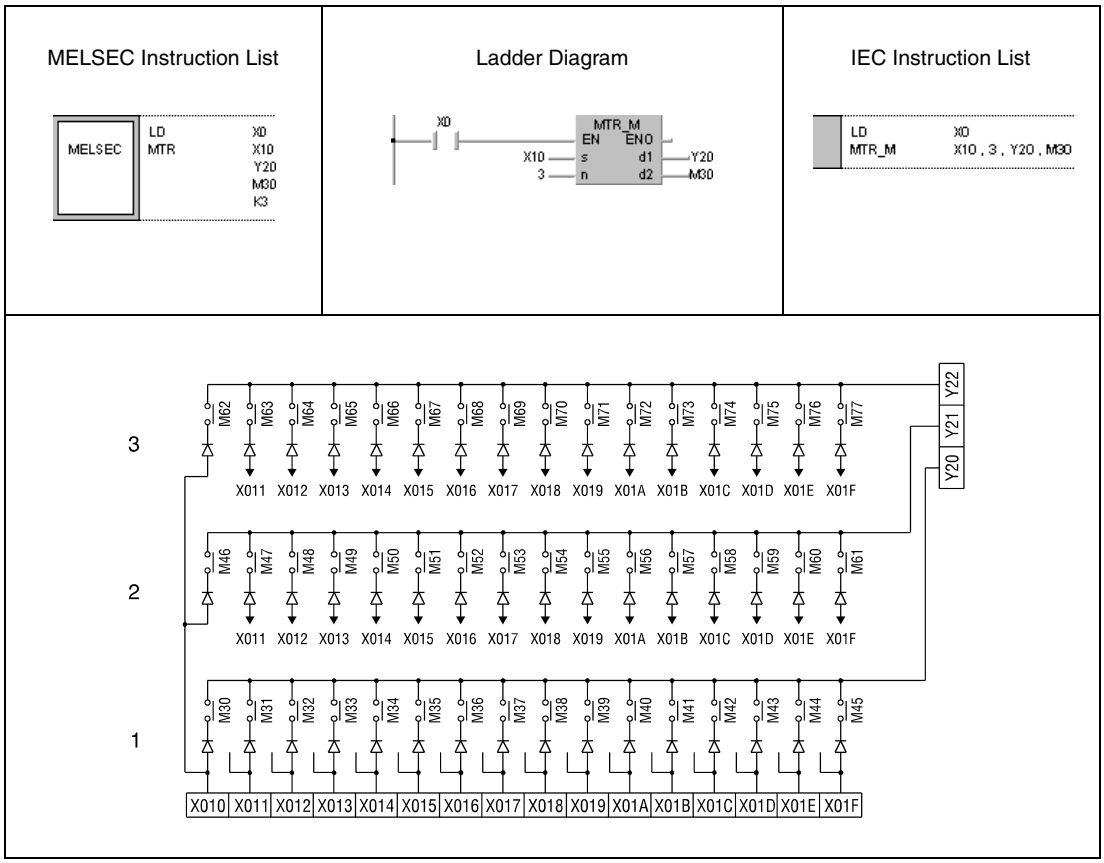
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device other than the input (X) was specified at s. (Error code 4101)
- The device other than the output (Y) was specified at d1. (Error code 4101)

Program Example

MTR

If X0 is set, the following program reads the inputs X10 through X1F three times and stores the results in M30 through M77. A matrix is built with 16 bits x 3 rows. The rows are addressed via the outputs Y20 through Y22.



[1] 1st row

[2] 2nd row

[3] 3rd row

7 Application Instructions, Part 2

The application instructions, part 2 are specific instructions for several special functions.

The following table shows the division of these functions:

Instruction	Meaning
Logical operation instructions	Logical AND / OR, logical exclusive OR / exclusive NOR
Rotation instructions	16-bit and 32-bit data right / left rotation
Shift instructions	Shift data by bit or word
Bit processing instructions	Set, reset, and test bits
Data processing instructions	Search, encode, and decode data at specified devices Disunite and unite data
Structured program instructions	Repeated operation, subroutine program calls, subroutine calls between program files, switching between main and subprogram parts, micro computer program calls, index qualification of entire ladders, store index qualification values in data tables
Data table operation instructions	Write to and read data from a data table, delete and insert data blocks in a data table
Buffer memory access instructions	Buffer memory access of special function modules
Display instructions	Output ASCII characters to the outputs of a module or to an LED display
Debugging and failure diagnosis instructions	Failure checks, setting and resetting status latch, sampling trace, program trace
Character string processing instructions	Character string (ASCII code) processing
Special function instructions	Trigonometrical functions, square root and exponential calculation with BCD data and floating point data
Data control instructions	Upper and lower limit control and storage of checked data
File register switching instructions	Switching between file register blocks and files
Clock instructions	Reading/writing of the values of year, month, day, hour, minute, second, and day of the week; addition/ subtraction of the values of hour, minute, and second; conversion of the values of hour, minute, and second into second; comparison between the values of year, month, and day; and comparison between the values of hour, minute, and second.
Expansion clock instruction	Reading of the values of year, month, day, hour, minute, second, millisecond, and day of the week; addition/ subtraction of the values of hour, minute, second, and millisecond
Peripheral device instructions	Message output and key input on peripheral units
Program instructions	Select different program execution modes
Other instructions	Reset watchdog timer (WDT), pulse generation, direct read from indirect access file registers, numerical key input from keyboard, batch save or recovery of index registers, reading module information/model name, trace set/trace reset, writing to and reading from files/standard ROM, program instructions, data transfer, user message

7.1 Logical operation instructions

Via the logical operation instructions logical connections such as logical sum or logical product are programmed.

The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
AND (logical product)	WAND	WAND_M, WAND_3_M
	WANDP	WANDP_M, WANDP_3_M
	DAND	DAND_M, DAND_3_M
	DANDP	DANDP_M, DANDP_3_M
	BKAND	BKAND_M
	BKANDP	BKANDP_M
OR (logical sum)	WOR	WOR_M, WOR_3_M
	WORP	WORP_M, WORP_3_M
	DOR	DOR_M, DOR_3_M
	DORP	DORP_M, DORP_3_M
	BKOR	BKOR_M
	BKORP	BKORP_M
Exclusive OR (XOR)	WXOR	WXOR_M, WXOR_3_M
	WXORP	WXORP_M, WXORP_3_M
	DXOR	DXOR_M, DXOR_3_M
	DXORP	DXORP_M, DXORP_3_M
	BKXOR	BKXOR_M
	BKXORP	BKXORP_M
Exclusive NOR (XNR)	WXNR	WXNR_M, WXNR_3_M
	WXNRP	WXNRP_M, WXNRP_3_M
	DXNR	DXNR_M, DXNR_3_M
	DXNRP	DXNRP_M, DXNRP_3_M
	BKXNR	BKXNR_M
	BKXNRP	BKXNRP_M

NOTE *Within the IEC editors please use the IEC instructions.*

Logical instructions are processed bit by bit as binary data. The two conditions (0 and 1) are connected and the result of the connection is output to a destination address.

The following table shows the logical connection results of the conditions 0 and 1. A and B are input variables and Y is the output variable.

Logical Connection	Processing Details	Operation Expression	Example		
			A	B	Y
Logical AND	Output Y set to 1, only if both inputs A **and** B are set to 1.	Y = A x B	0	0	0
			0	1	0
			1	0	0
			1	1	1
Logical OR	Output Y set to 1, if at least one of the inputs A **or** B is set to 1.	Y = A + B	0	0	0
			0	1	1
			1	0	1
			1	1	1
Logical exclusive OR (XOR)	Output Y set to 1, if the inputs A and B are different, and is set to 0 if A and B are equal.	$Y = \overline{A} \times B + A \times \overline{B}$	0	0	0
			0	1	1
			1	0	1
			1	1	0
Logical exclusive NOR (XNR)	Output Y set to 1, if the inputs A and B are equal, and is set to 0, if A and B are different.	$Y = (\overline{A} + B)\ (A + \overline{B})$	0	0	1
			0	1	0
			1	0	0
			1	1	1

7.1.1 WAND, WANDP, DAND, DANDP

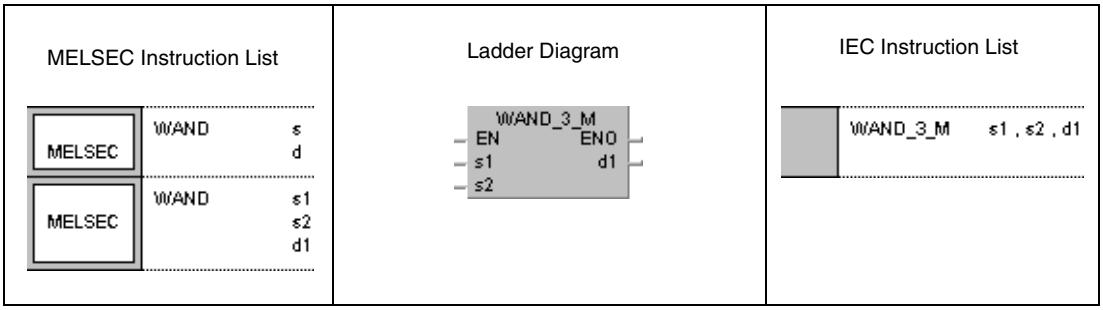
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

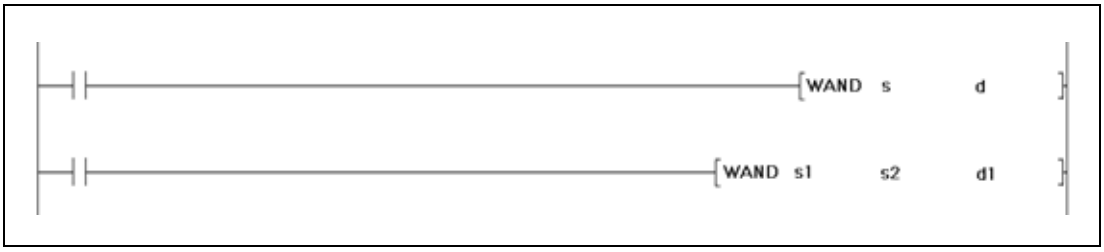
Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
WAND									
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—
DAND									
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
s	Data for logical product, or first number of device storing such data.	BIN 16-/32-bit
d		
s1	Data for logical product, or first number of device storing such data.	
s2		
d1 (for DAND d)	First number of device storing result of logical operation.	

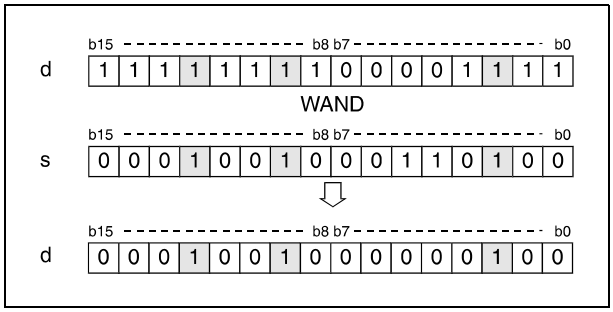
Functions **Logical AND**

WAND 16-bit data

The logical AND forms the logical product of two input variables.

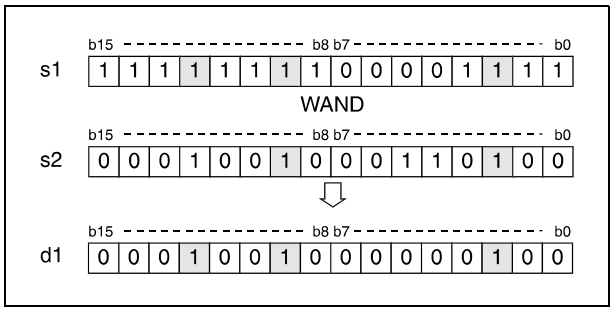
- Variation 1:

16-bit data designated by s and d form the logical product bit by bit. The result is output to the device designated by d.



- Variation 2:

16-bit data designated by s1 and s2 form the logical product bit by bit. The result is output to the device designated by d1.

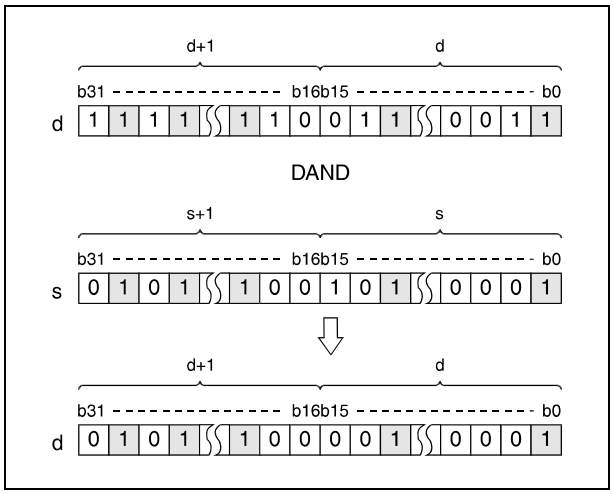


Bits exceeding the digit designation are set to 0. For example, if the digit designation is specified by K2, the higher 8 bits (b8 through b15) are processed as 0.

DAND 32-bit data

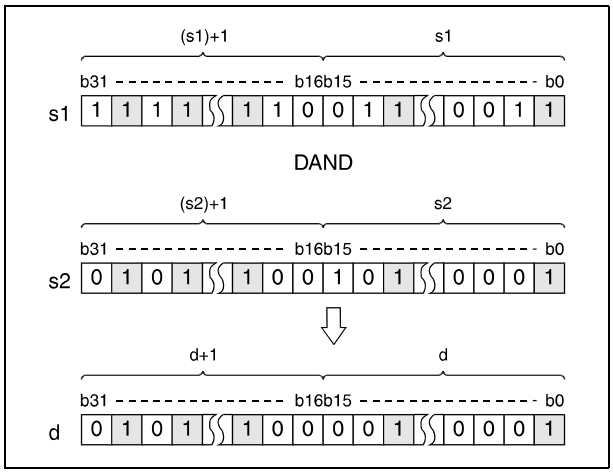
● Variation 1:

32-bit data designated by s and d form the logical product bit by bit. The result is output to the device designated by d.



● Variation 2:

32-bit data designated by s1 and s2 form the logical product bit by bit. The result is output to the device designated by d.

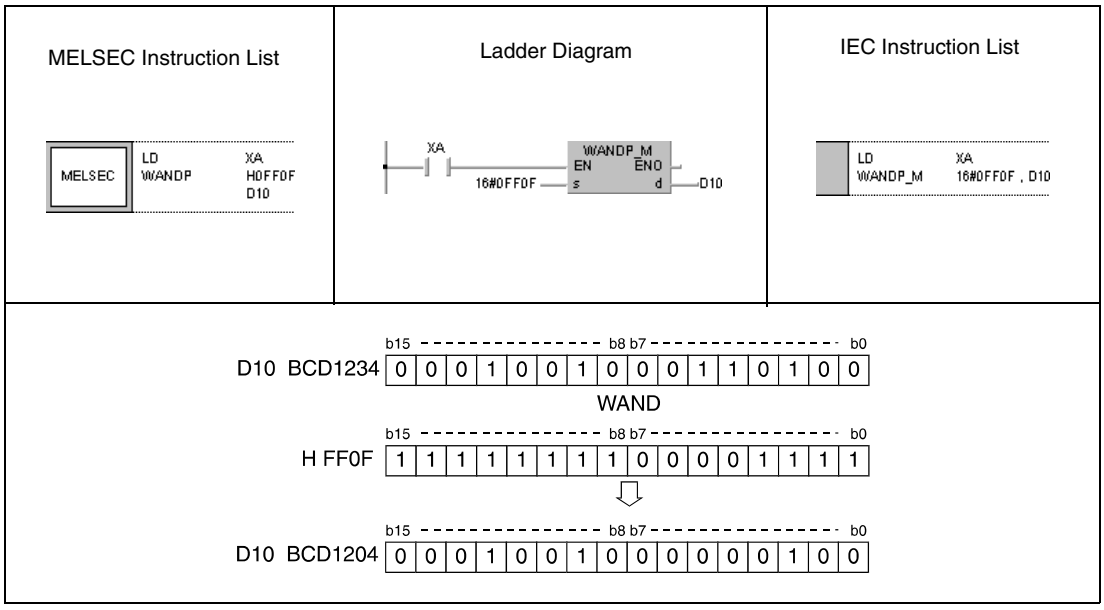


After executing the connection, all bits exceeding the digit designation are set to 0.

Program Example 1

WANDP (s, d)

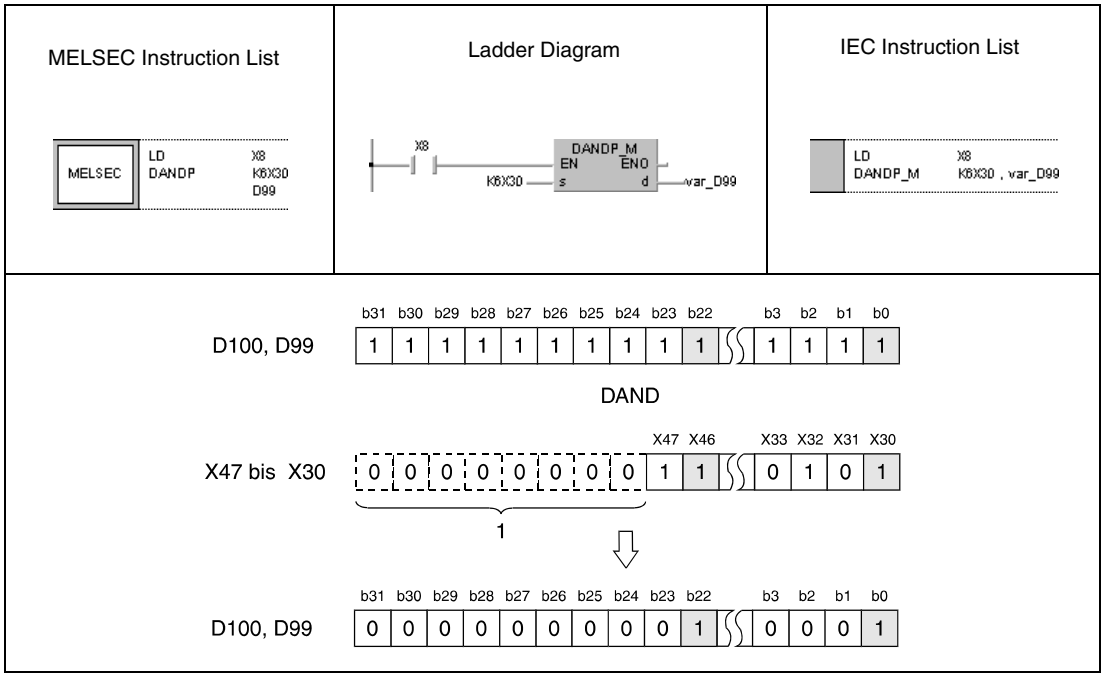
With leading edge from XA, the following program sets the digit of tens (b5-b7) in the BCD 4-digit value in D10 to 0. The result is stored again in D10.



Program Example 2

DANDP (s, d)

With leading edge from X8, the following program forms the logical product of 32-bit data in D99 and D100 and 24-bit data at X30 through X47. The result is stored again in D99 and D100.

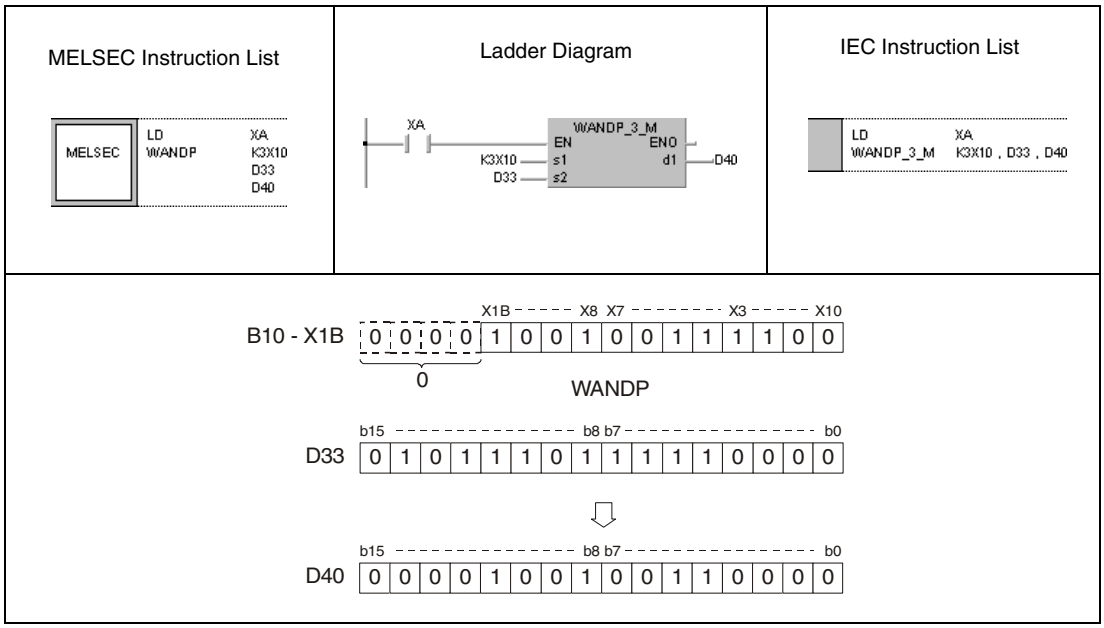


[1] These bits are set to 0.

Program Example 3

WANDP (s1, s2, d1)

With leading edge from XA, the following program forms the logical product of data in X10 through X1B and data in D33. The result is stored in D40.

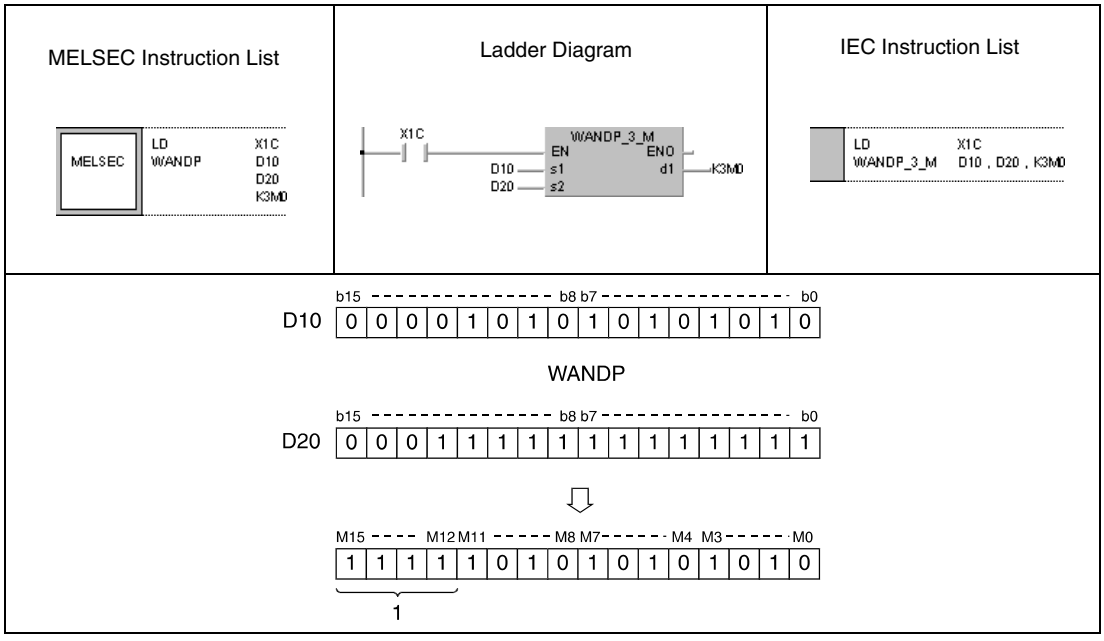


[1] These bits are set to 0.

Program Example 4

WANDP (s1, s2, d1)

With leading edge from X1C, the following program forms the logical product of data in D10 and D20. The result is stored in M0 through M11.

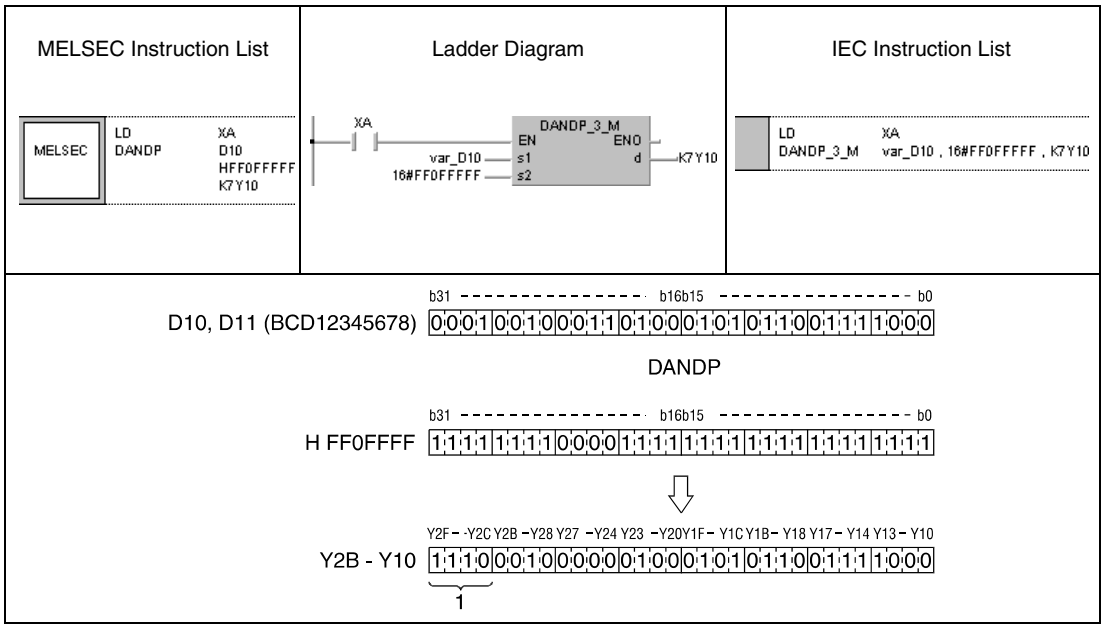


[1] These bits remain unchanged.

Program Example 5

DANDP (s1, s2, d)

With leading edge from XA, the following program sets the digit of hundreds in the BCD 4-digit value in D10 and D11 to 0. The result is output at Y10 through Y2B.



[1] These bits remain unchanged.

NOTE *The program examples 2 and 5 will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.1.2 BKAND, BKANDP

CPU

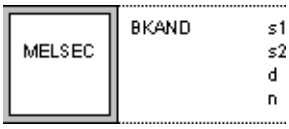
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

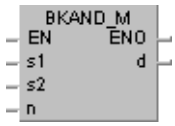
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

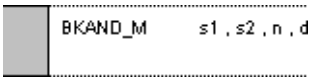
MELSEC Instruction List



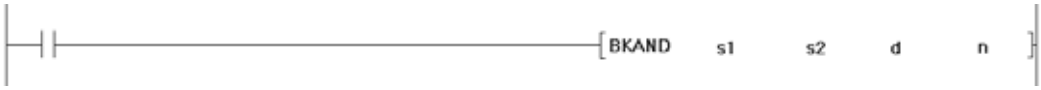
Ladder Diagram



IEC Instruction List



GX Works2



Variables

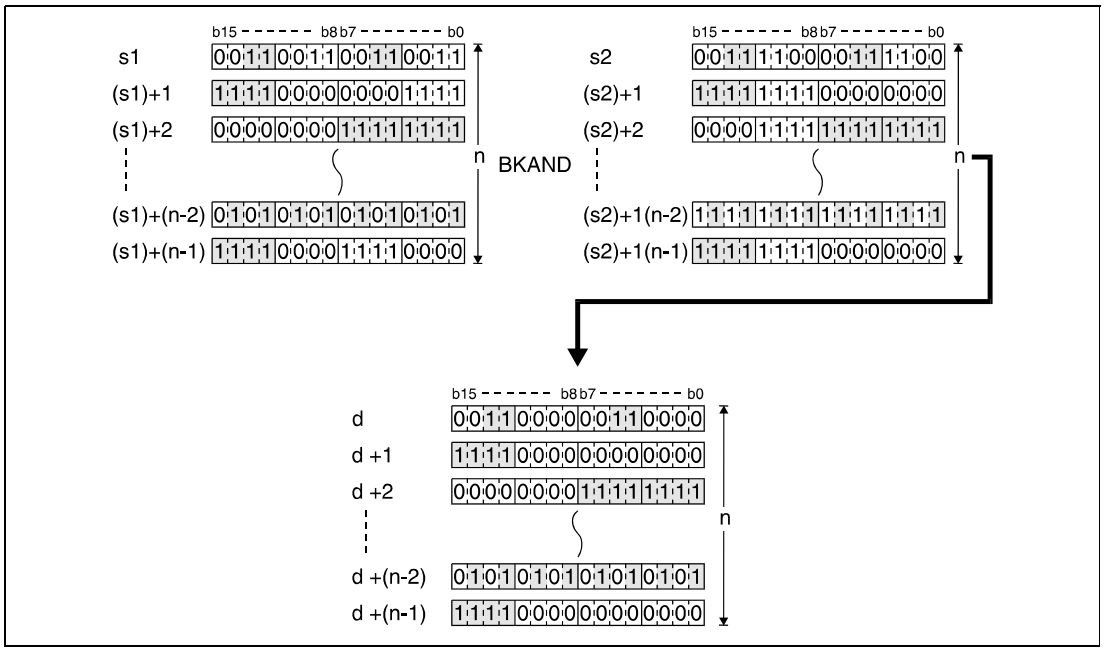
Set Data	Meaning	Data Type
s1	First number of device storing data for logical product.[1)]	BIN 16-bit
s2	First number of data or first number of device storing data for logical operation.[1)]	
d	First number of device storing result of logical operation.[1)]	
n	Number of data blocks forming the logical product.	

[1] The same device number can be specified for s1 and d or s2 and d.

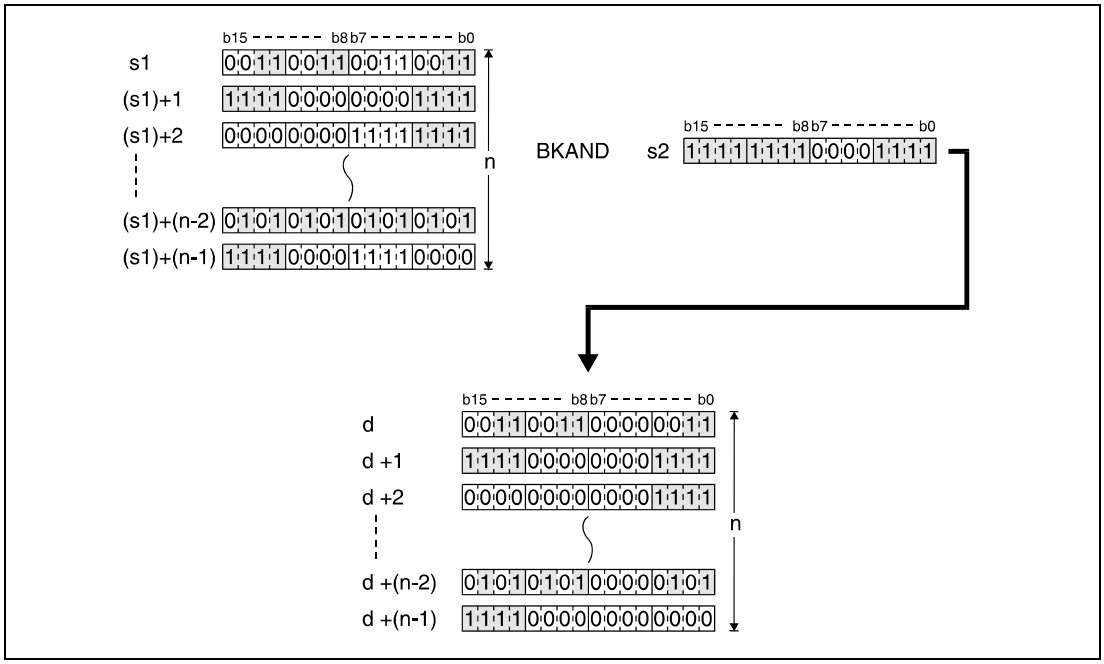
Functions **Forming a logical product with 16-bit data blocks**

BKAND Forming a logical product with data blocks

The BKAND instruction forms the logical product beginning with the nth 16-bit data block from s1 onwards and with the nth 16-bit data block from s2 onwards. The according 16-bit block of the result is stored beginning from device d onwards. The number of blocks to be processed is specified by n.



The constant in s2 must range within -32768 and 32767.



Operation Errors

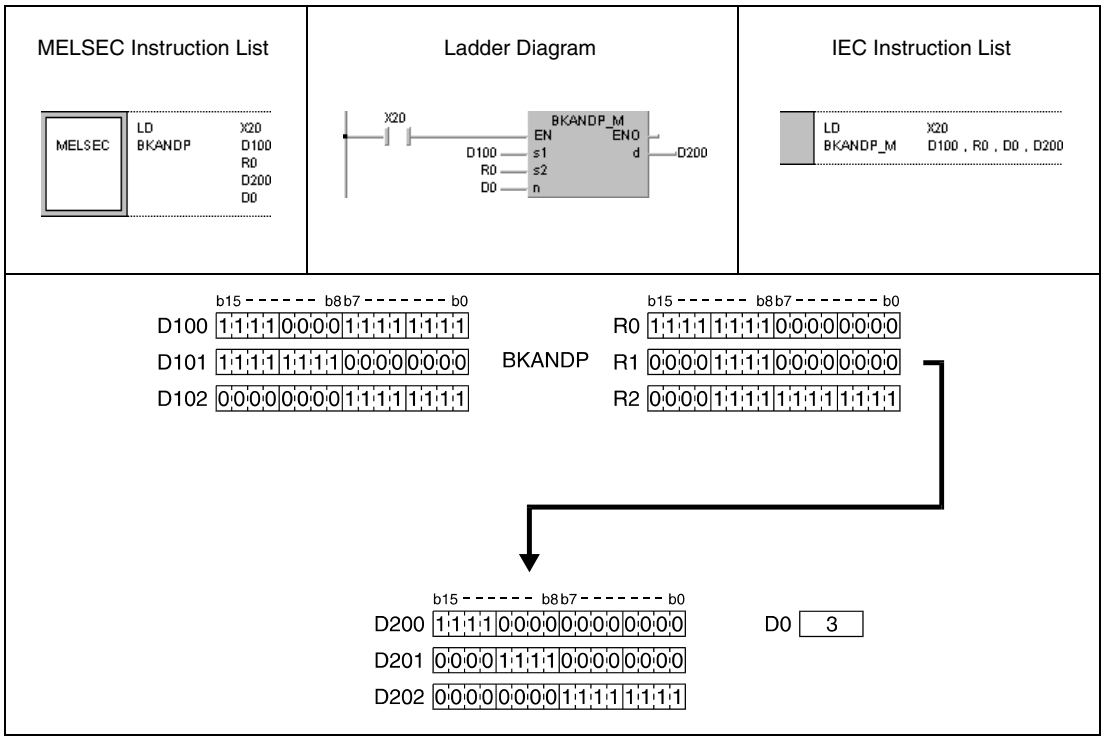
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of data blocks determined by n exceeds the storage device numbers designated by s1, s2, or d.
  (Error code 4101)
- The device range for n points starting from the device designated by s1 overlaps with the device range for n points starting from the device designated by d (except when the same device is specified for s1 and d).
  (Error code: 4101)
- The device range for n points starting from the device designated by s2 overlaps with the device range for n points starting from the device designated by d (except when the same device is specified for s2 and d).
  (Error code: 4101)

Program Example

BKANDP

With leading edge from X20, the following program forms the logical product of data in registers D100 through D102 and data in registers R0 through R2. The result is stored in registers D200 through D202. The number of 16-bit data blocks (3) to be processed is stored in D0.



7.1.3 WOR, WORP, DOR, DORP

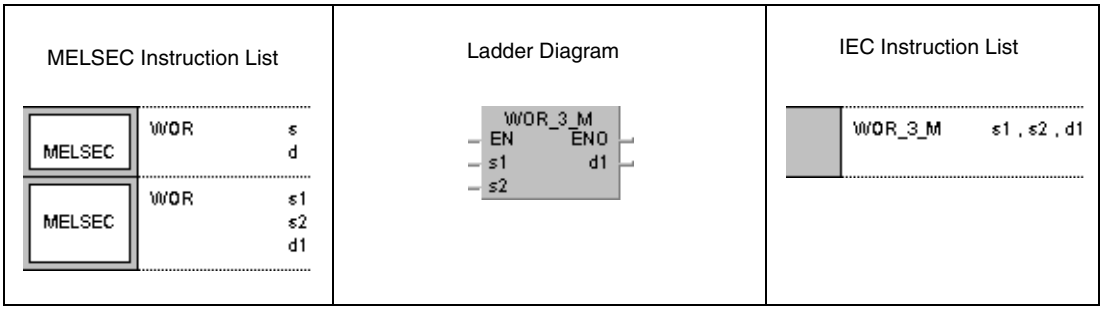
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

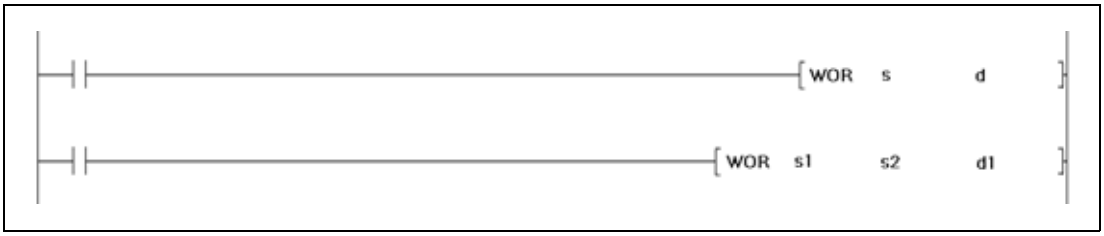
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
WOR									
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—
DOR									
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
s	Data for logical sum, or first number of device storing such data.	BIN 16-/32-bit
d		
s1	Data for logical sum, or first number of device storing such data.	
s2		
d1 (for DOR d)	First number of device storing result of logical operation.	

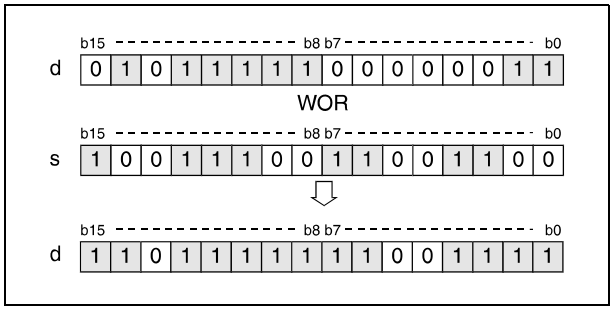
Functions **Logical OR**

WOR 16-bit data

The logical OR forms the logical sum of two input variables.

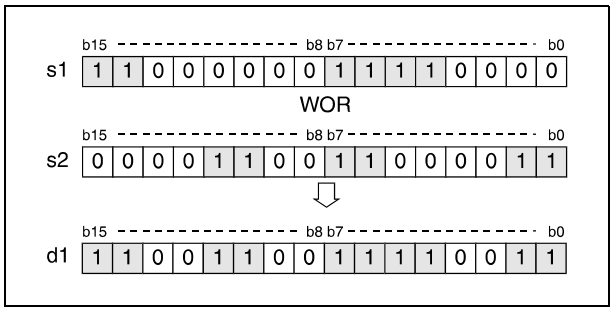
● Variation 1:

16-bit data designated by s and d are added bit by bit. The result is output to the device designated by d.



● Variation 2:

16-bit data designated by s1 and s2 are added bit by bit. The result is output to the device designated by d1.

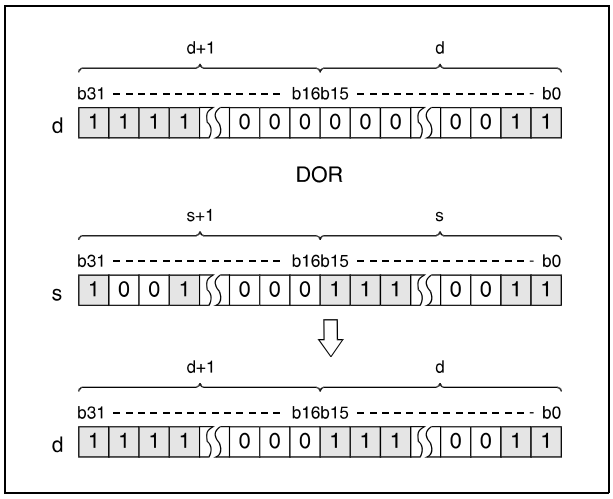


Bits exceeding the digit designation are set to 0. For example, if the digit designation is specified by K2, the higher 8 bits (b8 through b15) are processed as 0.

DOR 32-bit data

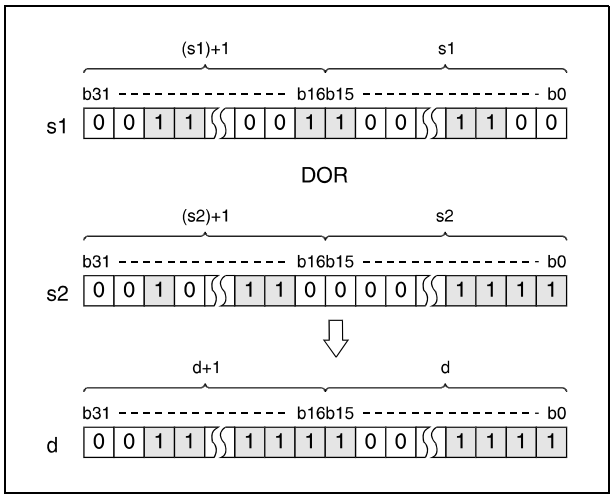
- Variation 1:

32-bit data designated by s and d are added bit by bit. The result is output to the device designated by d.



- Variation 2:

32-bit data designated by s1 and s2 are added bit by bit. The result is output to the device designated by d.

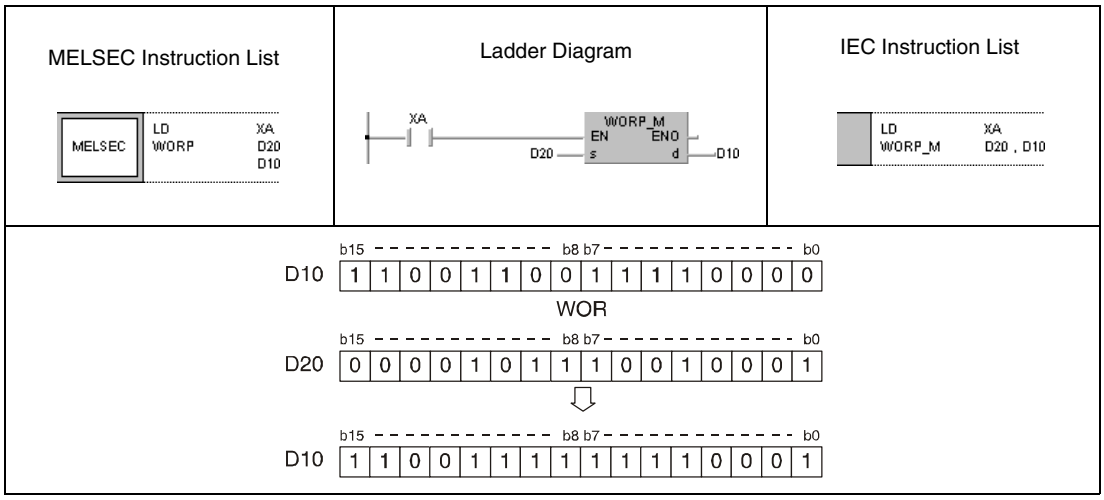


After executing the connection, all bits exceeding the digit designation are set to 0.

Program Example 1

WORP (s, d)

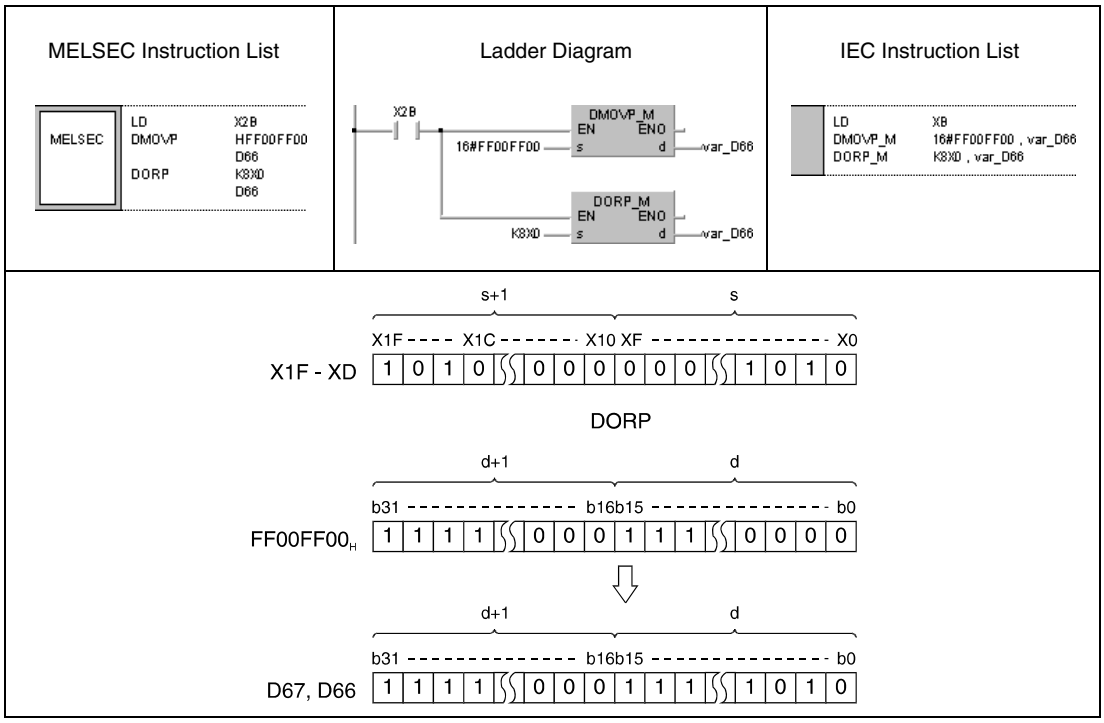
With leading edge from XA, the following program adds data in D10 to data in D20. The result is stored in D10.



Program Example 2

DORP (s, d)

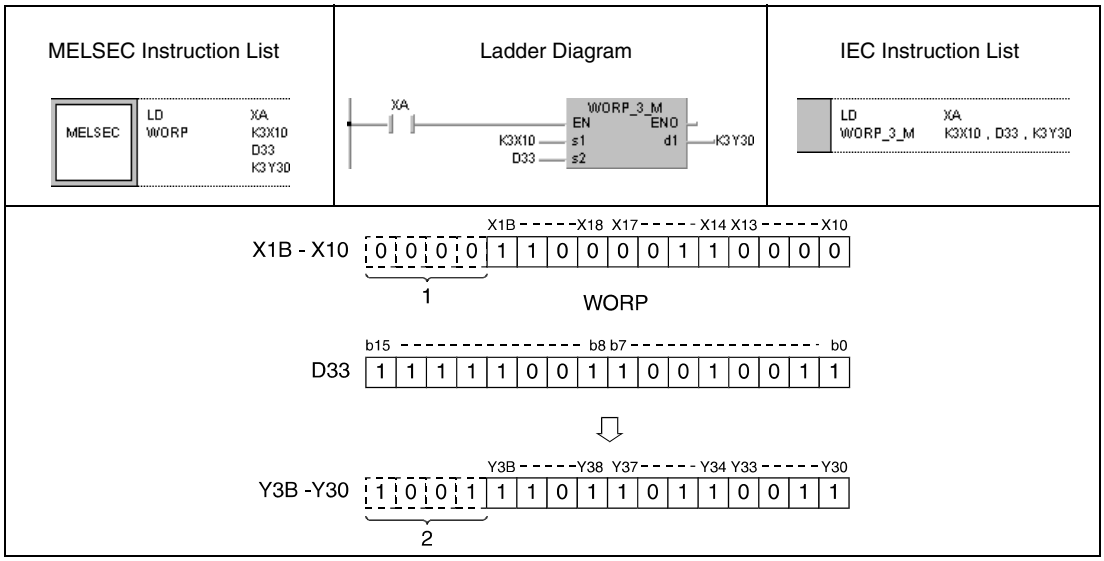
With leading edge from X2B, the following program adds data at the inputs X0 through X1F to a hexadecimal value FF00FF00. The result is stored in D66 and D67.



Program Example 3

WORP (s1, s2, d1)

With leading edge from XA, the following program adds data at the inputs X10 through X1B to data in D33. The result is output to the outputs Y30 through Y3B.



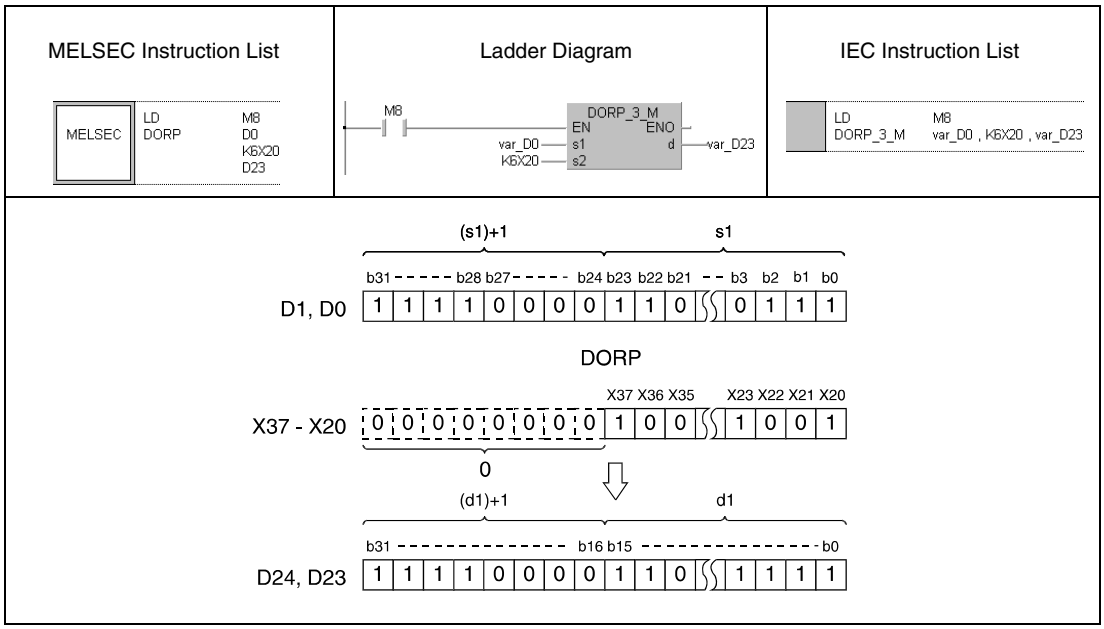
[1] These bits are set to 0.

[2] These bits remain unchanged.

Program Example 4

DORP (s1, s2, d)

With leading edge from M8, the following program adds 32-bit data in D0 and D1 to 24-bit data at the inputs X20 through X37. The result is stored in D23 and D24.



NOTE *The program examples 2 and 4 will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.1.4 BKOR, BKORP

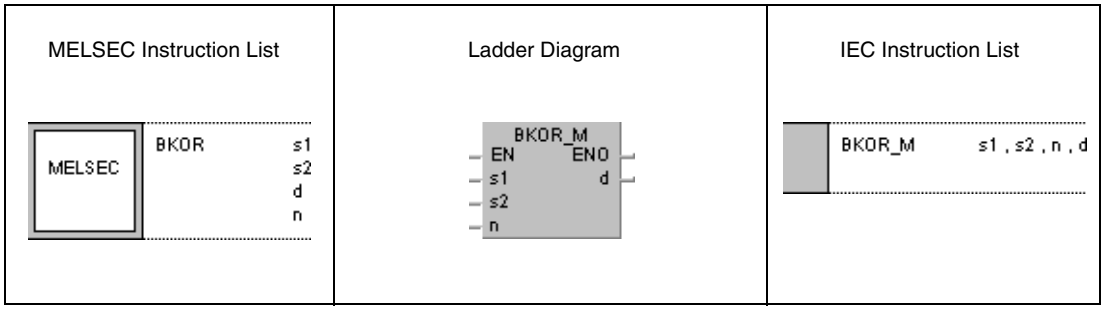
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

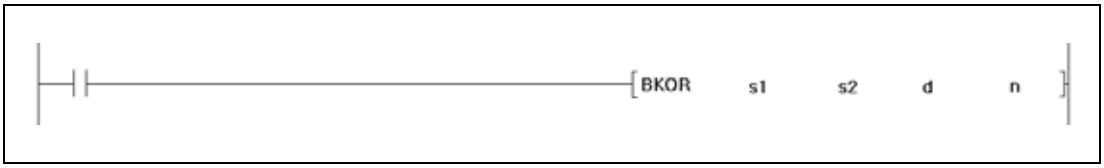
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer



GX Works2



Variables

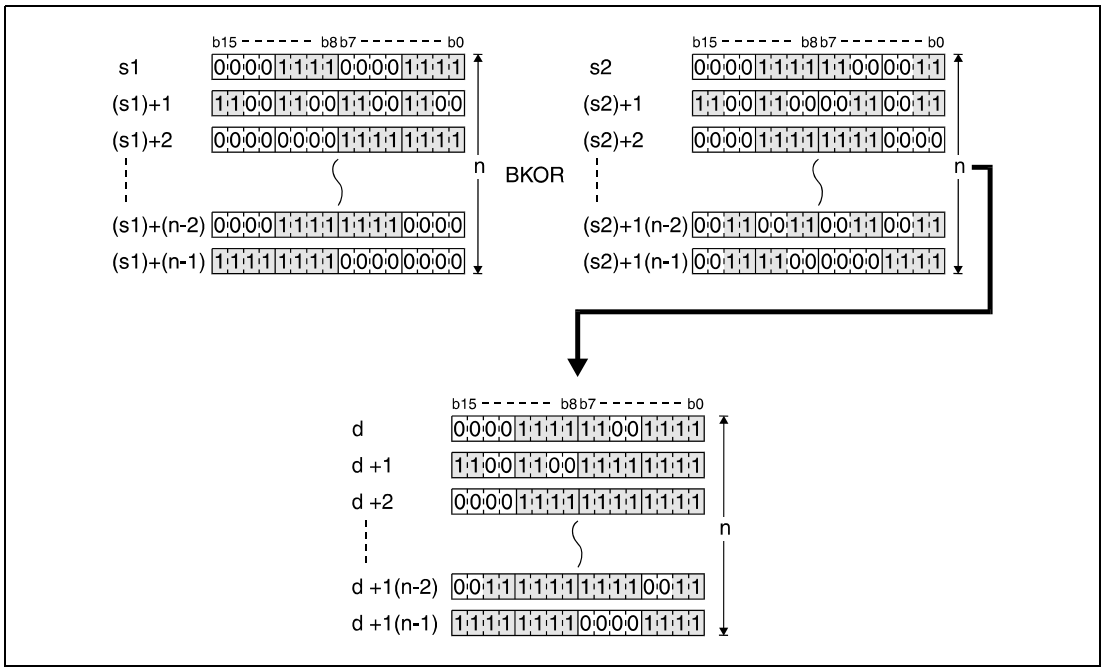
Set Data	Meaning	Data Type
s1	First number of device storing data for logical sum.[1)]	BIN 16-bit
s2	First number of data, or first number of device storing data for logical sum.[1)]	
d	First number of device storing result of logical operation.[1)]	
n	Number of data blocks forming the logical sum.	

[1] The same device number can be specified for s1 and d or s2 and d.

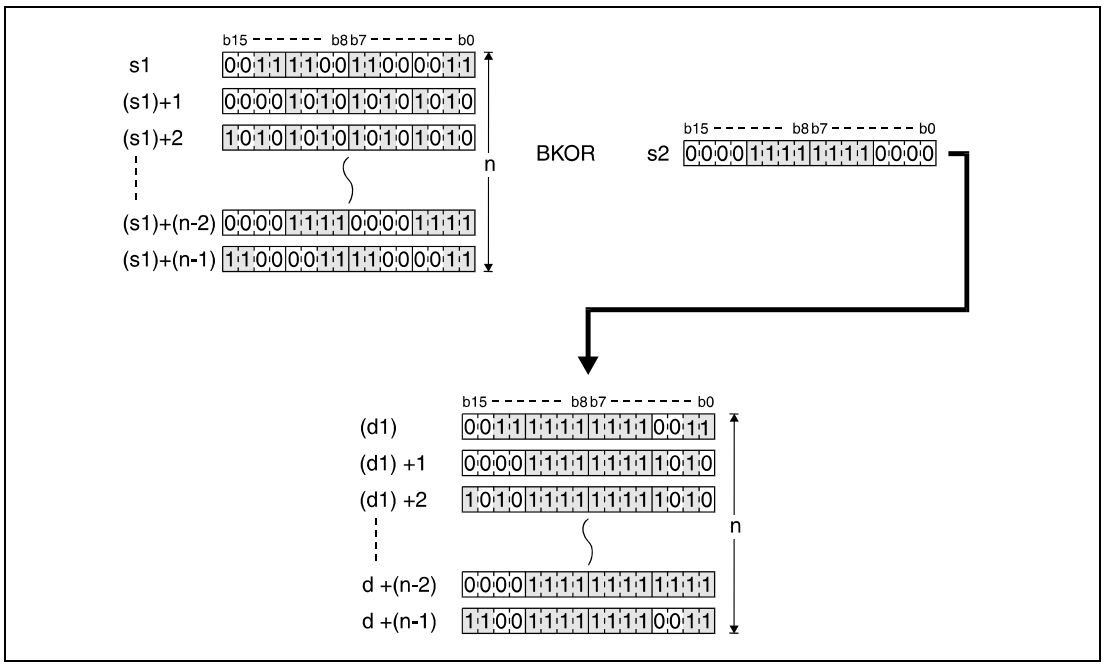
Functions **Forming a logical sum with 16-bit data blocks**

BKOR Forming a logical sum with data blocks

The BKOR instruction forms the logical sum beginning with the nth 16-bit data block from s1 onwards and with the nth 16-bit data block from s2 onwards. The according 16-bit block of the result is stored beginning from device d onwards. The number of blocks to be processed is specified by n.



The constant in s2 must range within -32768 and 32767.



Operation Errors

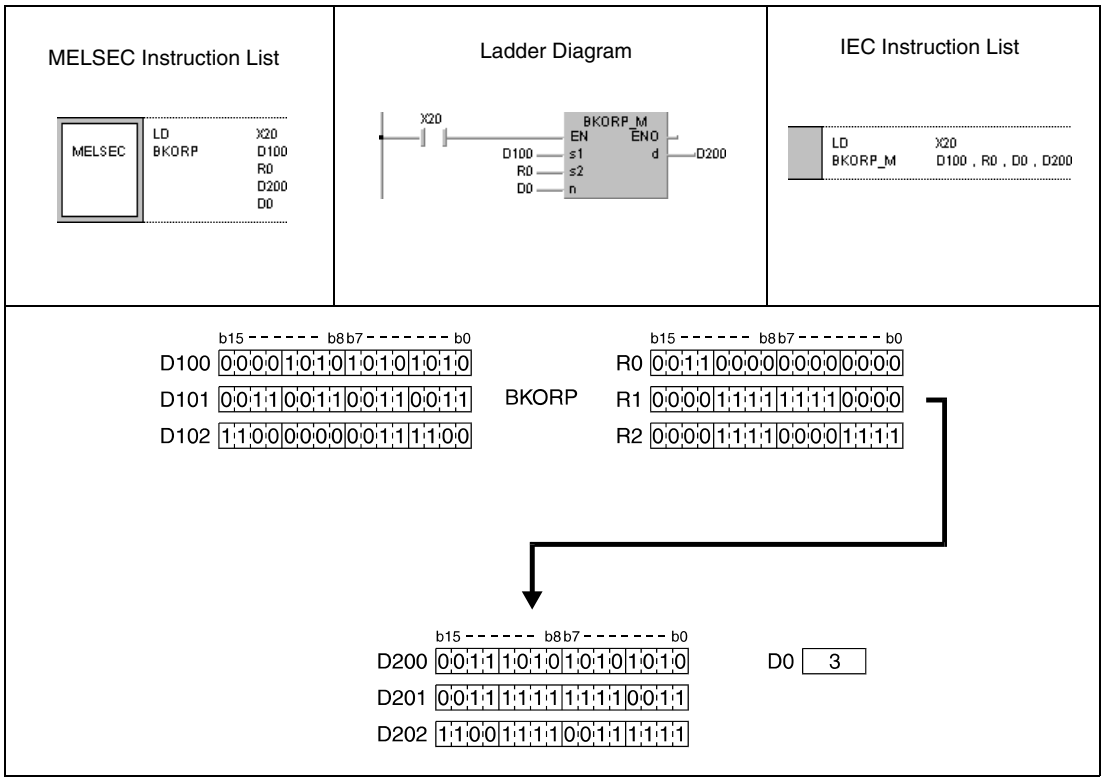
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of data blocks determined by n exceeds the storage device numbers designated by s1, s2, or d.
  (Error code 4101)
- The device range for n points starting from the device designated by s1 overlaps with the device range for n points starting from the device designated by d (except when the same device is specified for s1 and d).
  (Error code: 4101)
- The device range for n points starting from the device designated by s2 overlaps with the device range for n points starting from the device designated by d (except when the same device is specified for s2 and d).
  (Error code: 4101)

Program Example

BKORP

With leading edge from X20, the following program forms the logical sum of data in registers D100 through D102 and data in registers R0 through R2. The result is stored in registers D200 through D102. The number of 16-bit data blocks (3) to be processed is stored in D0.



7.1.5 WXOR, WXORP, DXOR, DXORP

CPU

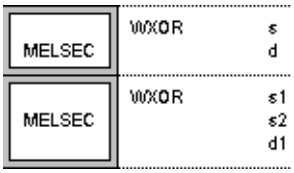
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

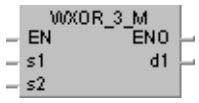
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
WXOR									
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—
DXOR									
	●	●	●	●	●	●	●	●	—
	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer

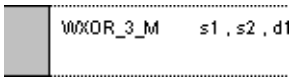
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
s	Data for exclusive OR operation, or first number of device storing such data.	BIN 16-/32-bit
d		
s1	Data for exclusive OR operation, or first number of device storing such data.	
s2		
d1 (for DXOR d)	First number of device storing result of logical operation.	

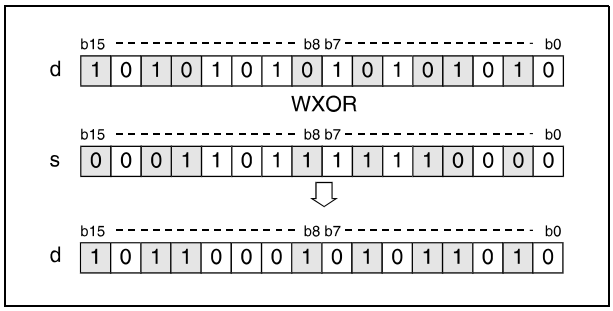
Functions **Logical exclusive OR**

WXOR 16-bit data

The logical exclusive OR forms the logical sum of two input variables (Y= ($\overline{A}$xB)+(Ax$\overline{B}$)).

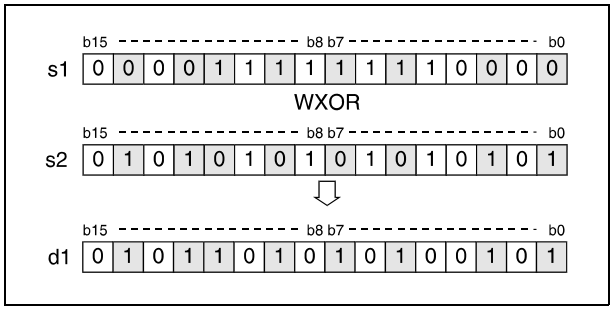
● Variation 1:

16-bit data designated by s and d form a logical exclusive OR connection. The result is output to the device designated by d.



● Variation 2:

16-bit data designated by s1 and s2 form a logical exclusive OR connection. The result is output to the device designated by d.

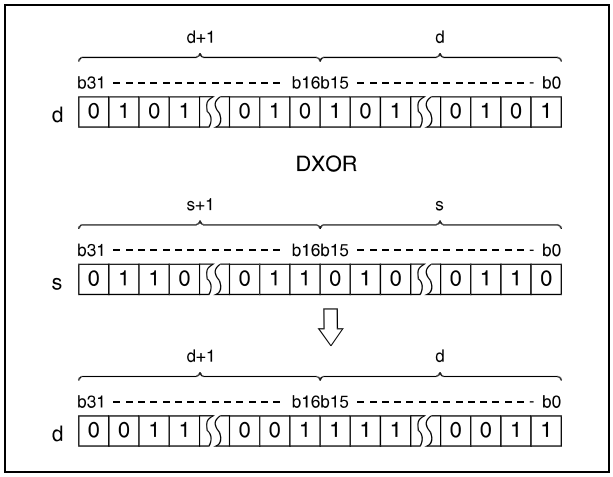


Bits exceeding the digit designation are set to 0. For example, if the digit designation is specified by K2, the higher 8 bits (b8 through b15) are processed as 0.

DXOR 32-bit data

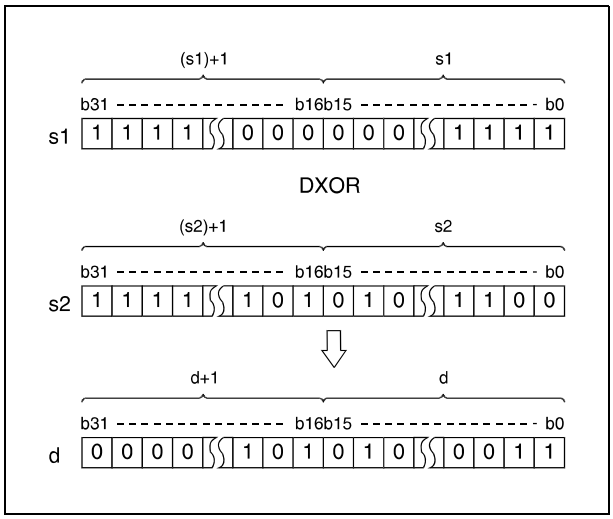
- Variation 1:

32-bit data designated by s and d form a logical exclusive OR connection. The result is output to the device designated by d.



- Variation 2:

32-bit data designated by s1 and s2 form a logical exclusive OR connection. The result is output to the device designated by d.

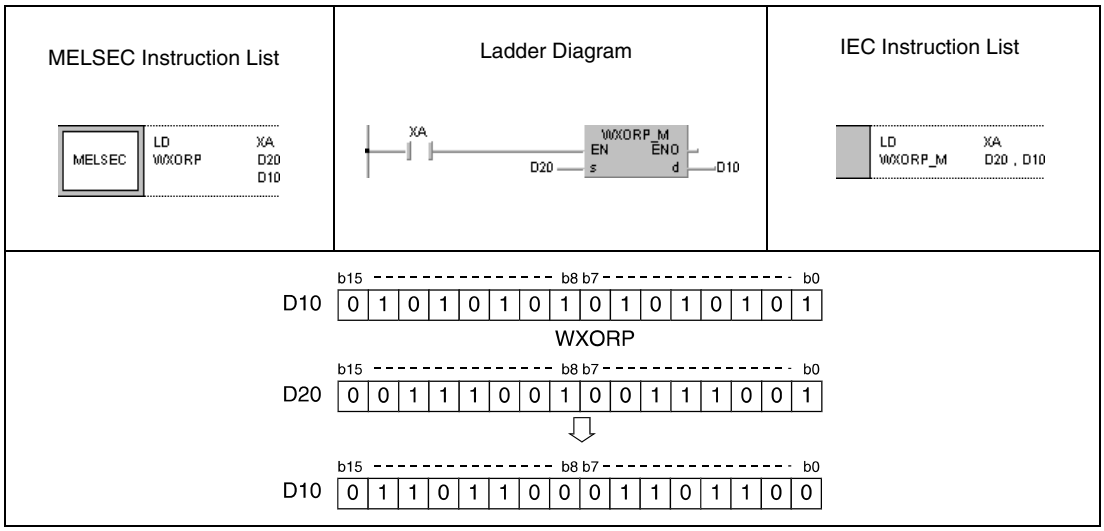


After executing the connection, all bits exceeding the digit designation are set to 0.

Program Example 1

WXORP (s, d)

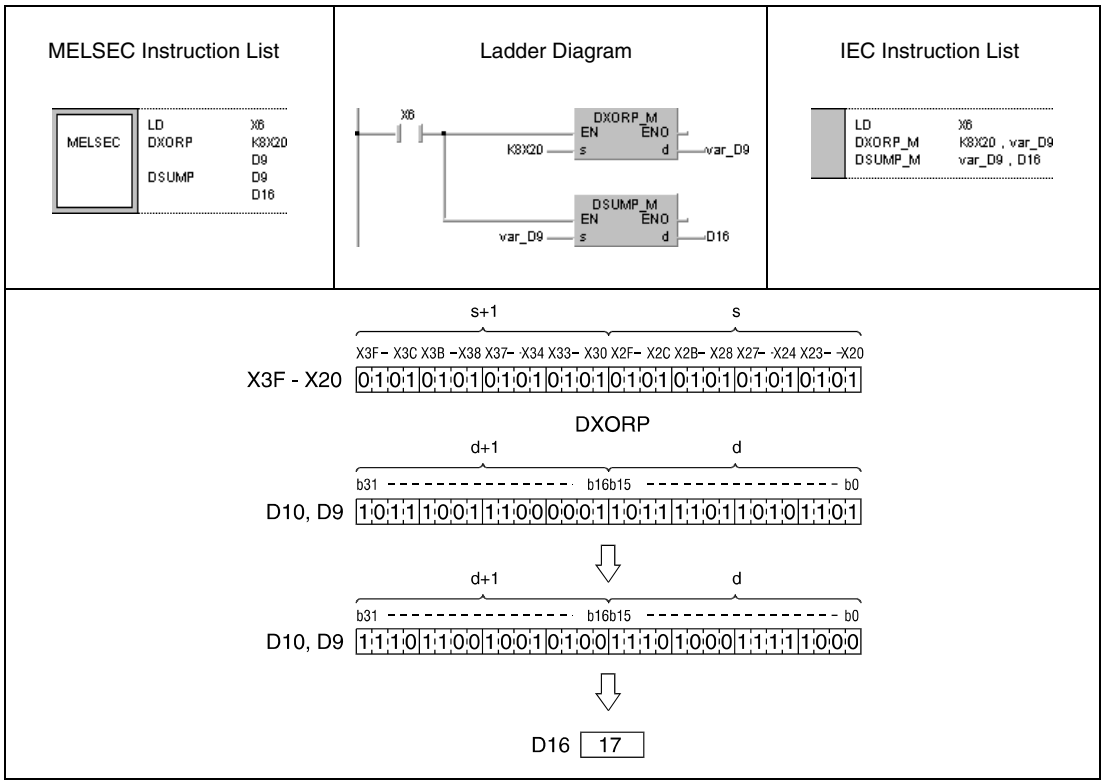
With leading edge from XA, the following program connects data in D10 with data in D20. The result is stored again in D10.



Program Example 2

DXORP (s, d)

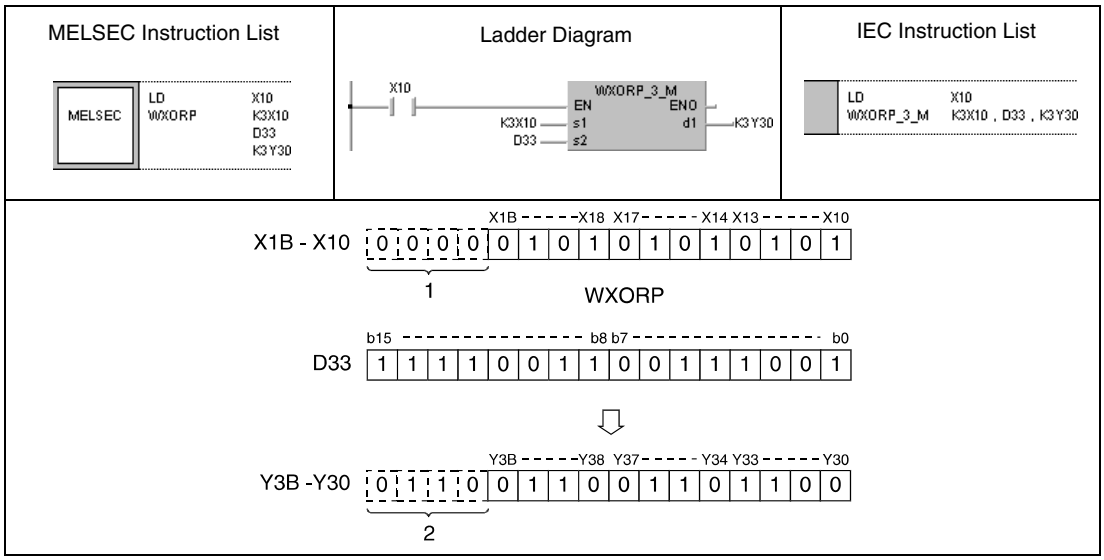
With leading edge from X6, the following program compares 32-bit data at the inputs X20 through X3F to the bit pattern in data registers D9 and D10. The result is stored again in D9 and D10. The number of set bits in D9 and D10 is stored in D16.



Program Example 3

WXORP (s1, s2, d1)

With leading edge from X10, the following program forms an exclusive OR connection of input data X10 through X1B with data in D33. The result is stored in D33 and output to Y30 through Y3B.



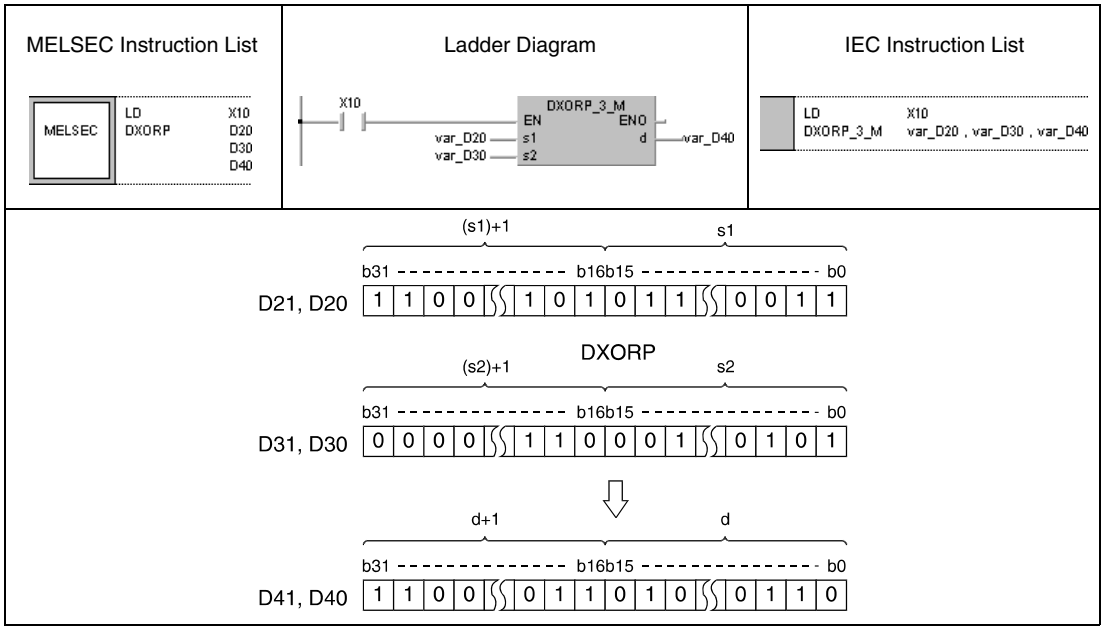
[1] These bits are set to 0.

[2] These bits remain unchanged.

Program Example 4

DXORP (s1, s2, d)

With leading edge from X10, the following program forms an exclusive OR connection of data in D20 and D21 with data in D30 and D31. The result is stored in D40 and D41.



NOTE *The program examples 2 and 4 will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.1.6 BKXOR, BKXORP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

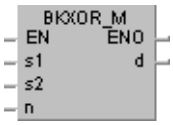
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

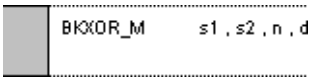
GX IEC Developer

MELSEC Instruction List

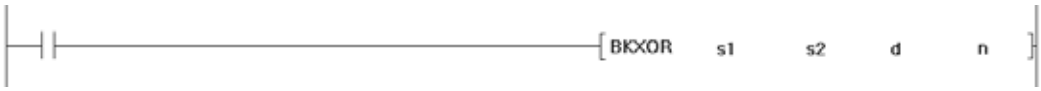
Ladder Diagram



IEC Instruction List



GX Works2



Variables

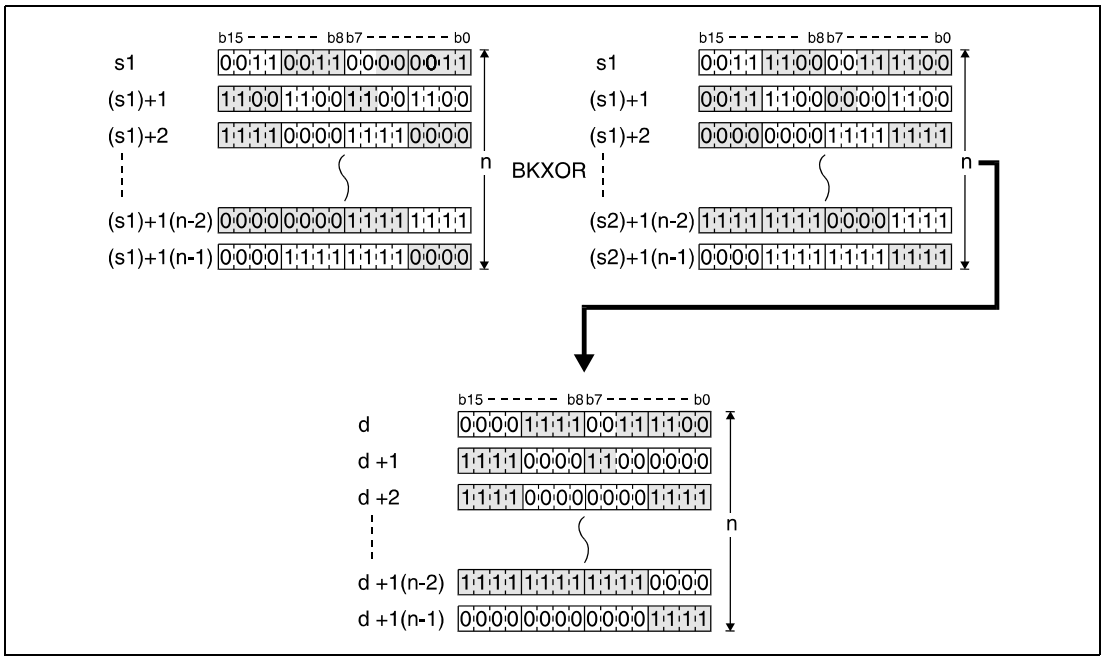
Set Data	Meaning	Data Type
s1	First number of device storing data for logical operation.[1)]	BIN 16-bit
s2	First number of data, or first number of device storing data for logical operation.[1)]	
d	First number of device storing result of operation.[1)]	
n	Number of data blocks forming the exclusive OR operation.	

[1] The same device number can be specified for s1 and d or s2 and d.

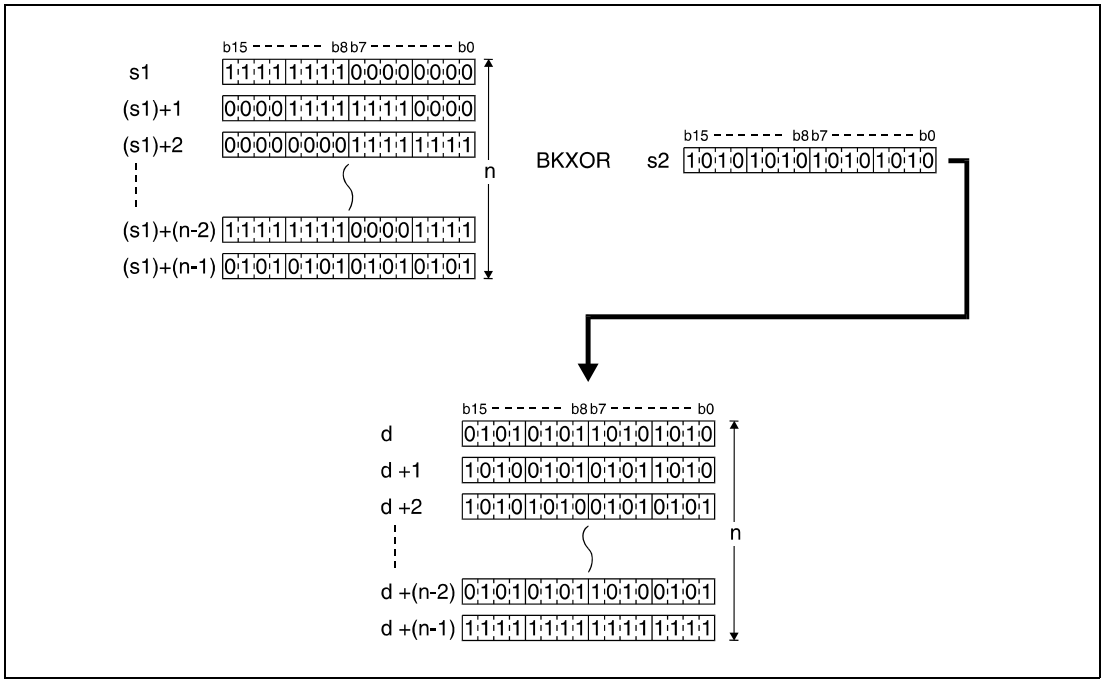
Functions **Exclusive OR operations with 16-bit data blocks**

BKXOR Exclusive OR operations with data blocks

The BKXOR instruction performs an exclusive OR operation beginning with the nth 16-bit data block from s1 onwards and with the nth 16-bit data block from s2 onwards. The according 16-bit block of the result is stored beginning from device d onwards. The number of blocks to be processed is specified by n.



The constant in s2 must range within -32768 and 32767.



Operation Errors

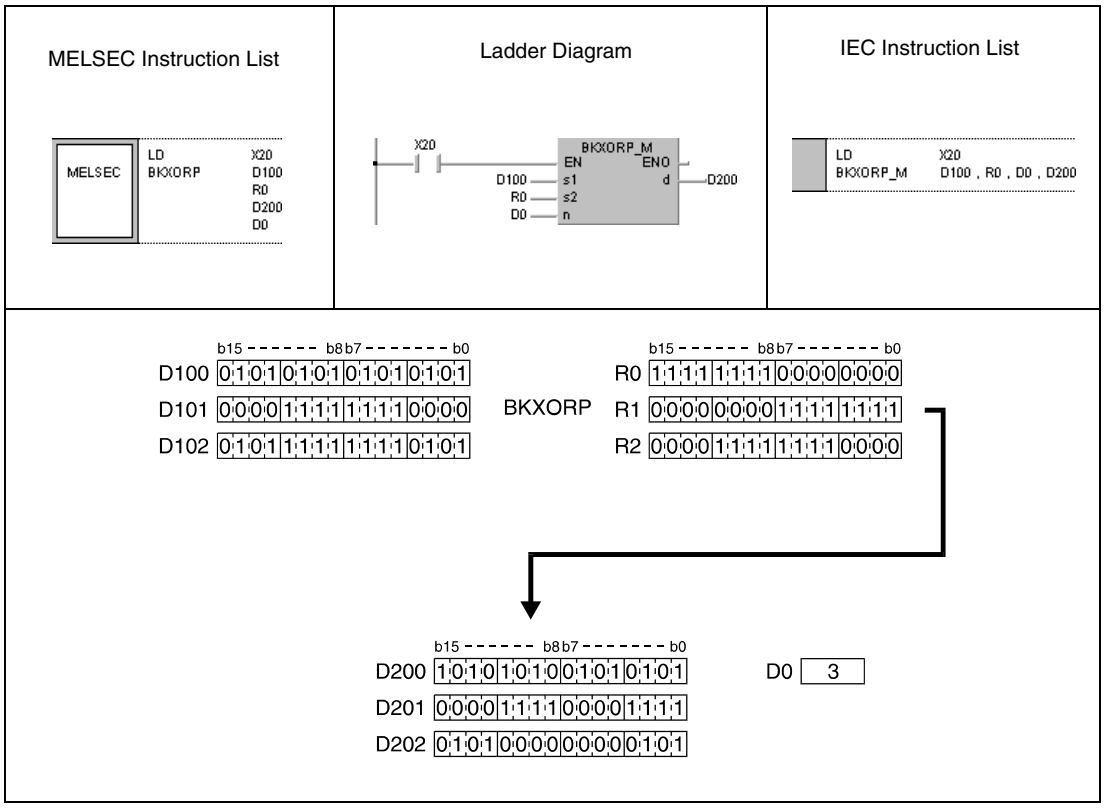
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of data blocks determined by n exceeds the storage device numbers designated by s1, s2, or d. (Error code 4101)
- The device range for n points starting from the device designated by s1 overlaps with the device range for n points starting from the device designated by d (except when the same device is specified for s1 and d).
  (Error code: 4101)
- The device range for n points starting from the device designated by s2 overlaps with the device range for n points starting from the device designated by d (except when the same device is specified for s2 and d).
  (Error code: 4101)

Program Example

BKXORP

With leading edge from X20, the following program performs an exclusive OR operation with data in registers D100 through D102 and data in registers R0 through R2. The result is stored in registers D200 through D202. The number of 16-bit data blocks (3) to be processed is stored in D0.



7.1.7 WXNR, WXNRP, DXNR, DXNRP

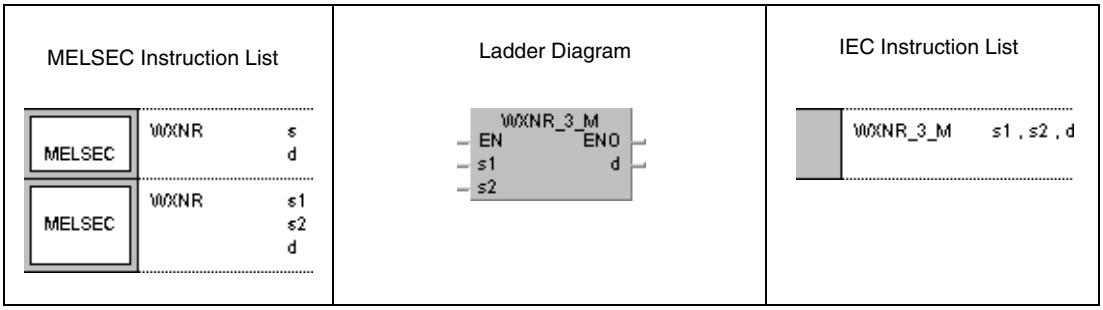
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

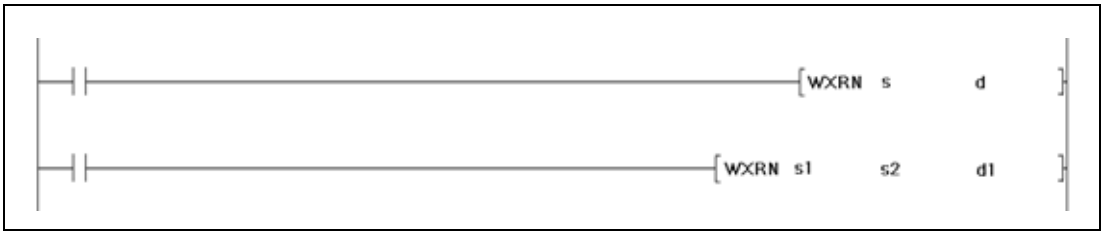
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
WXNR, WXNRP									
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
DXNR, DXNRP									
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	●	●	●	●	●	●	—	—

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
s	Data for exclusive NOR operation, or first number of device storing such data	BIN 16-/32-bit
d		
s1	Data for exclusive NOR operation, or first number of device storing such data	
s2		
d (d1 for WXNRP)	First number of device storing result of logical operation	

Functions

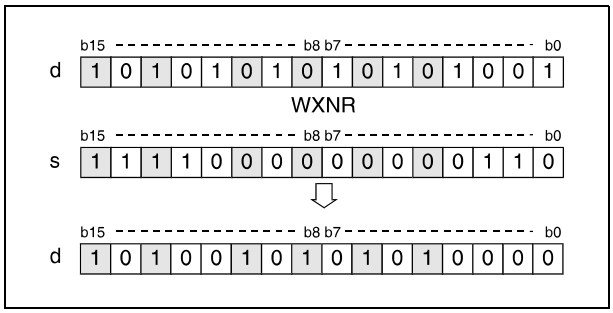
Logical exclusive NOR

WXNR 16-bit data

The logical exclusive NOR forms the logical product of the logical sum of two input variables ($Y= (\overline{A}+B) \times (A+\overline{B})$).

- Variation 1:

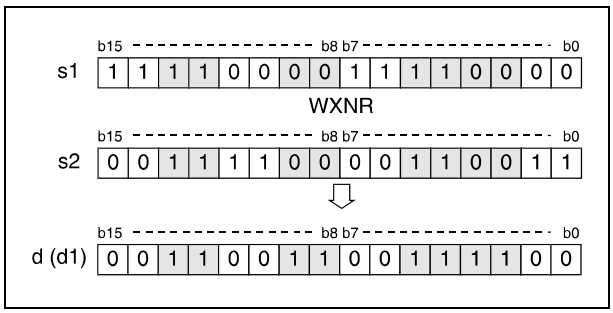
16-bit data designated by s and d form a logical exclusive NOR connection. The result is output to the device designated by d.



- Variation 2:

16-bit data designated by s1 and s2 form a logical exclusive NOR connection. The result is output to the device designated by d.

The WXNRP operation instruction outputs the result to the device designated by d1.

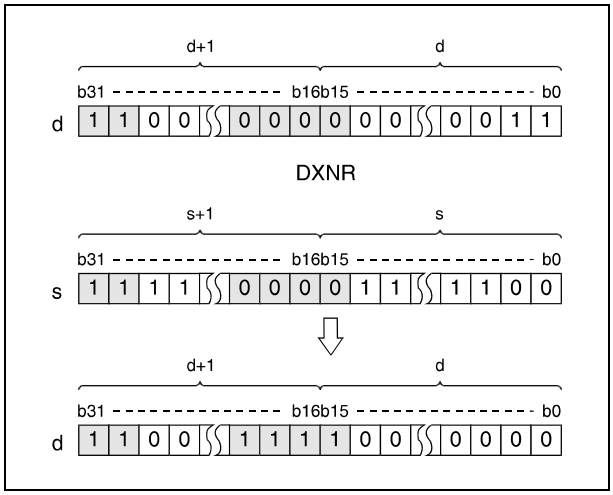


Bits exceeding the digit designation are set to 0. For example, if the digit designation is specified by K2, the higher 8 bits (b8 through b15) are processed as 0.

DXNR 32-bit data

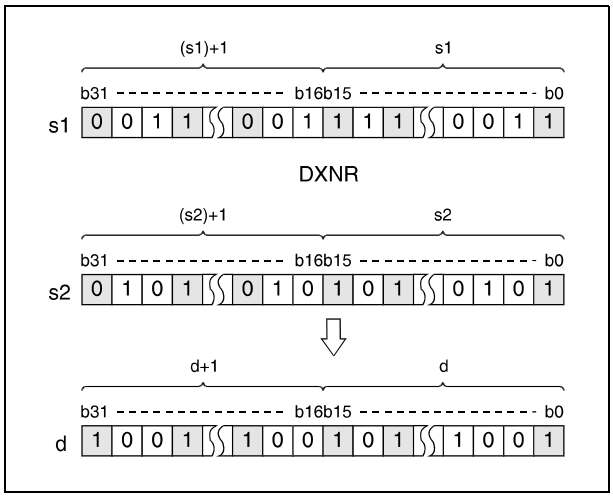
- Variation 1:

32-bit data designated by s and d form a logical exclusive NOR connection. The result is output to the device designated by d.



- Variation 2:

16-bit data designated by s1 and s2 form a logical exclusive NOR connection. The result is output to the device designated by d.



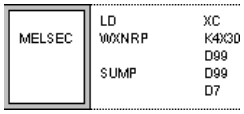
After executing the connection, all bits exceeding the digit designation are set to 0.

Program Example 1

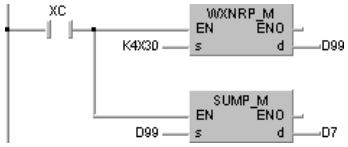
WXNRP (s, d)

With leading edge from XC, the following program compares the bit pattern of the 16-bit data value at the inputs X30 through X3F to the data value in D99. The result of the operation is stored again in D99. The number of set bits is stored in D7.

MELSEC Instruction List

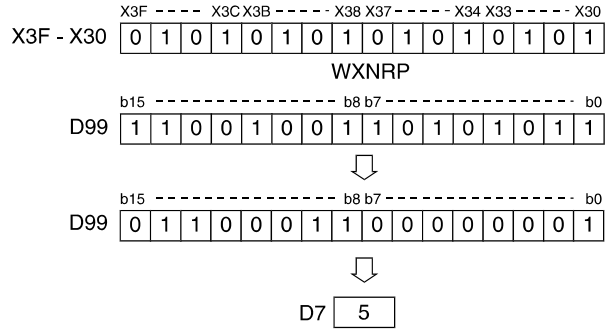


Ladder Diagram



IEC Instruction List

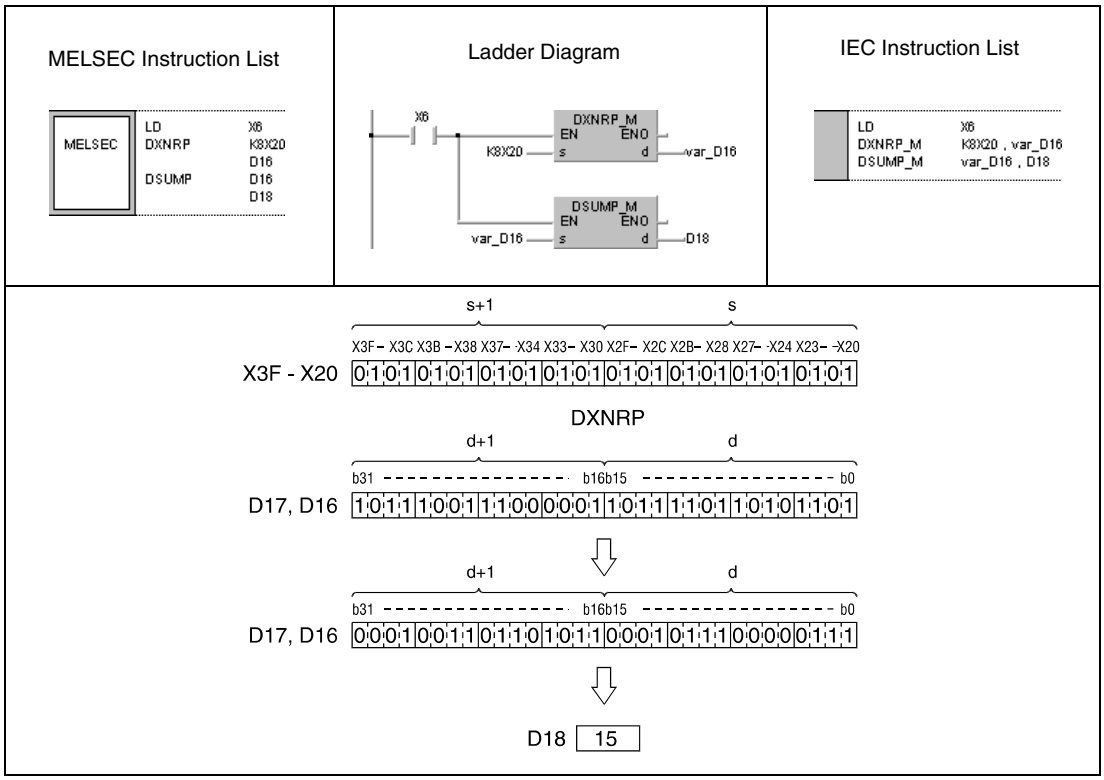
```
LD      XC
WXNRP_M K4X30 , D99
SUMP_M  D99 , D7
```



Program Example 2

DXNRP (s, d)

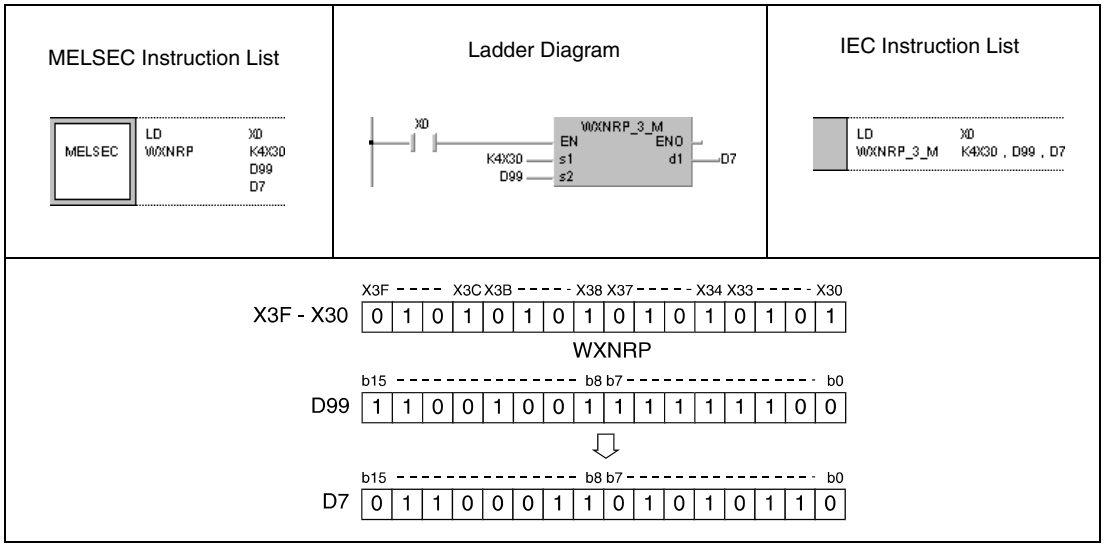
With leading edge from X6, the following program compares the bit pattern of the 32-bit data value at the inputs X20 through X3F to data in D16 and D17. The result of the operation is stored again in D16 and D17. The number of set bits is stored in D18.



Program Example 3

WXNRP (s1, s2, d1)

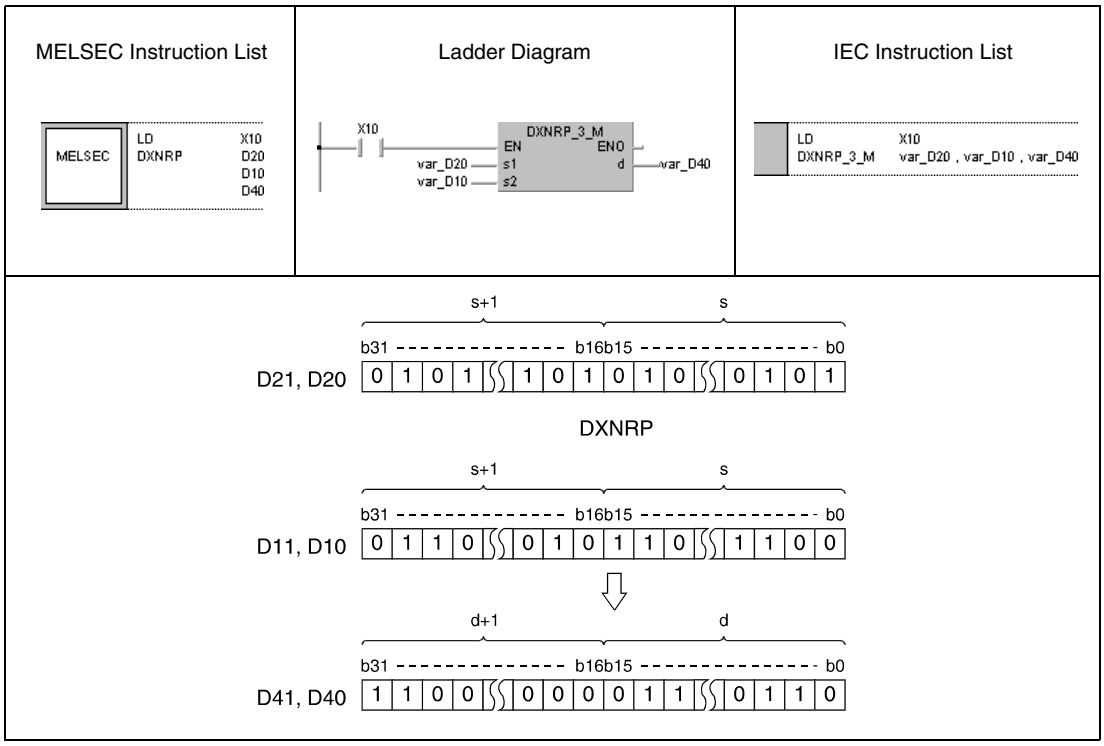
With leading edge from X0, the following program performs an exclusive NOR operation with 16-bit data at the inputs X30 through X3F and data in D99. The result of the operation is stored in D7.



Program Example 4

DXNRP (s1, s2, d)

With leading edge from X10, the following program performs an exclusive NOR operation with 32-bit data in the registers D20 and D21 and with data in D10 and D11. The result of the operation is stored in D40 and D41.



NOTE

The program examples 2 and 4 will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.1.8 BKXNR, BKXNRP

CPU

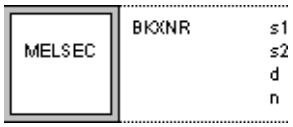
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

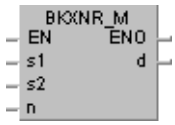
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

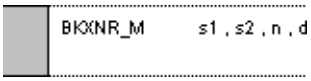
MELSEC Instruction List



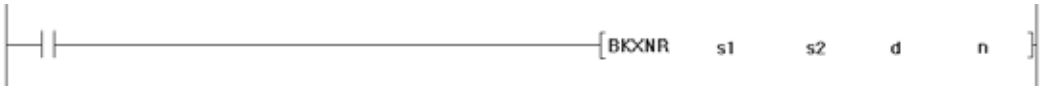
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s1	First number of device storing data for logical operation[1)]	BIN 16-bit
s2	First number of data, or first number of device storing data for logical operation[1)]	
d	First number of device storing result of logical operation [1)]	
n	Number of data blocks to be processed	

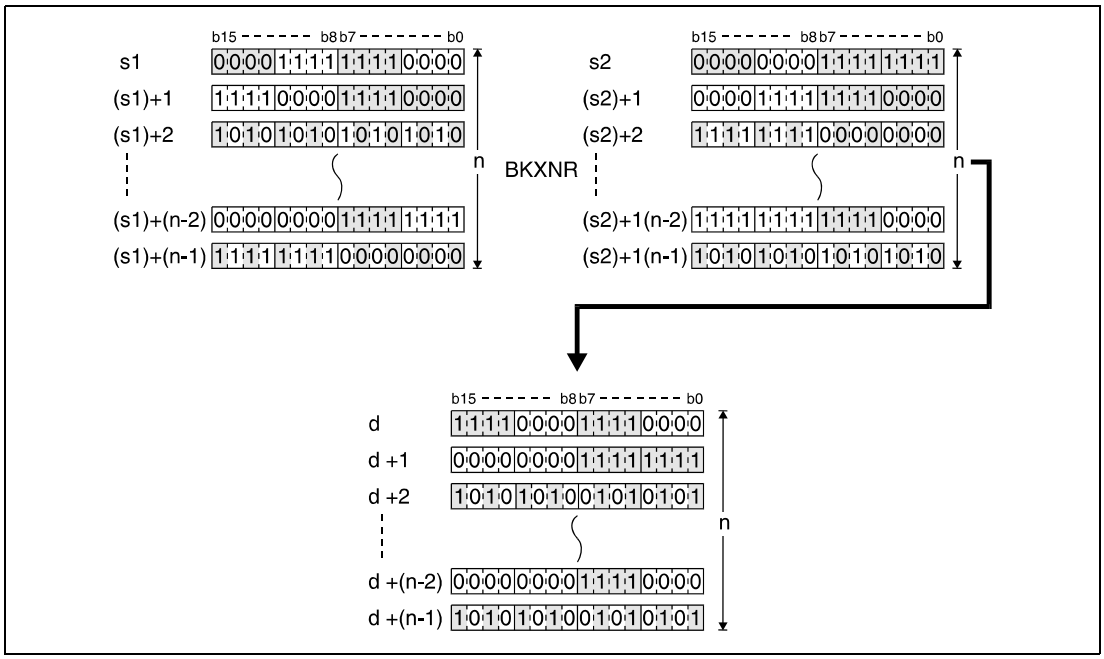
[1] The same device number can be specified for s1 and d or s2 and d.

Functions

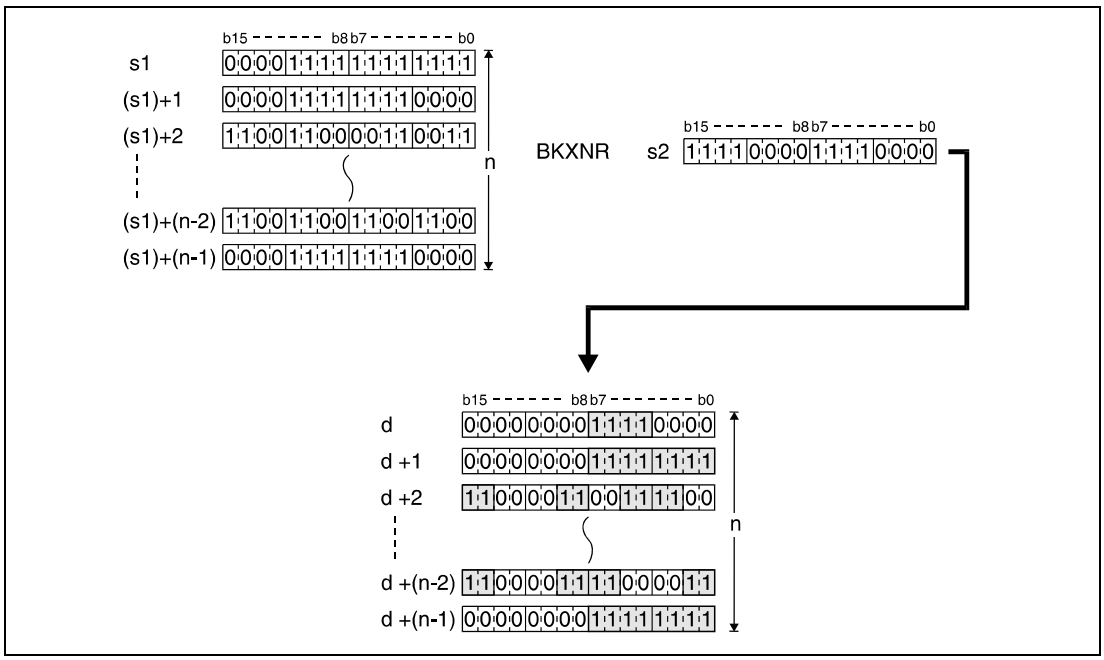
Exclusive NOR operations with 16-bit data blocks

BKXNR Exclusive NOR operations with data blocks

The BKXNR instruction performs an exclusive NOR operation beginning with the nth 16-bit data block from s1 onwards and with the nth 16-bit data block from s2 onwards. The according 16-bit block of the result is stored beginning from device d onwards. The number of blocks to be processed is specified by n.



The constant in s2 must range within -32768 and 32767.



Operation Errors

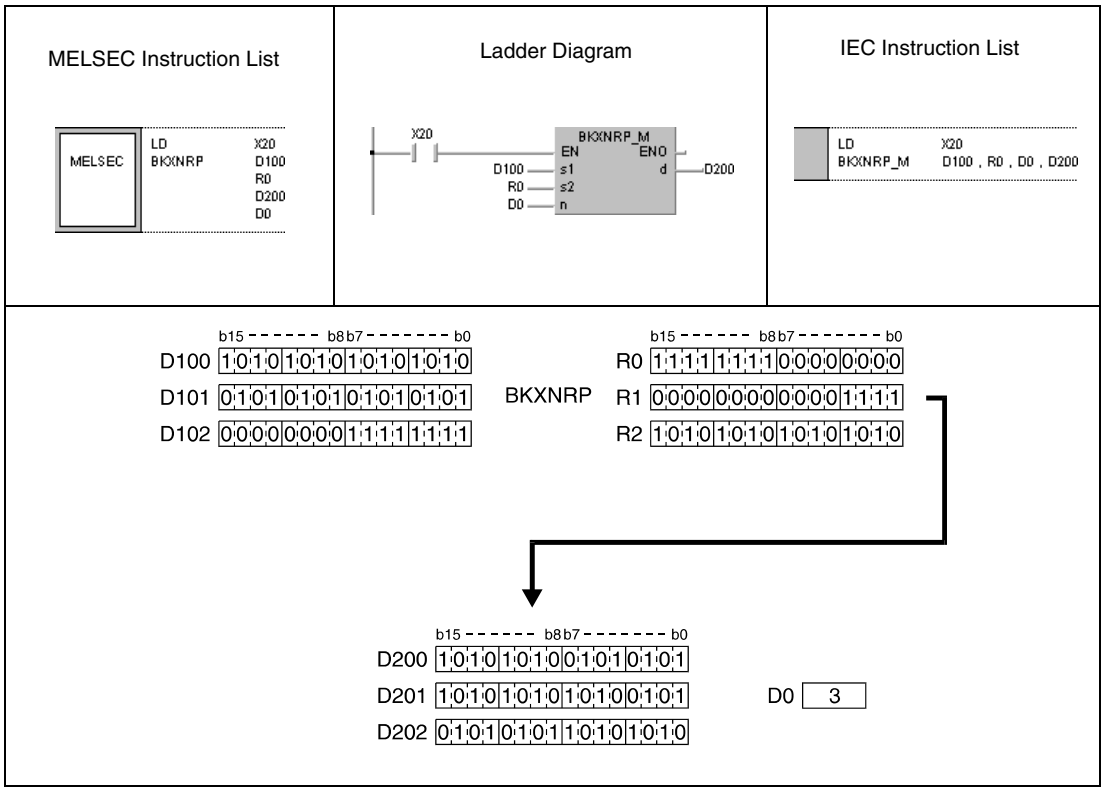
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of data blocks determined by n exceeds the storage device numbers designated by s1, s2, or d. (Error code 4101)
- The device range for n points starting from the device designated by s1 overlaps with the device range for n points starting from the device designated by d (except when the same device is specified for s1 and d).
  (Error code: 4101)
- The device range for n points starting from the device designated by s2 overlaps with the device range for n points starting from the device designated by d (except when the same device is specified for s2 and d).
  (Error code: 4101)

Program Example

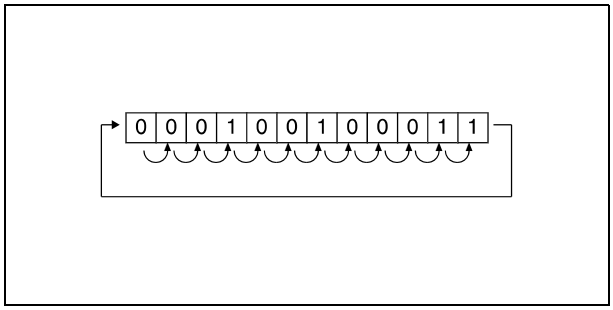
BKXNRP

With leading edge from X20, the following program performs an exclusive NOR operation with data in registers D100 through D102 and with data in registers R0 through R2. The result of the operation is stored in the registers D200 through D202. The number of 16-bit blocks (3) to be processed is stored in D0.



7.2 Data rotation instructions

The following rotation instructions rotate data stored in devices bit by bit. Data can be rotated to the right as well as to the left.



Example for a rotation to the right.

Rotation instructions can alternatively be applied with or without carry flag. The rotation instructions are suitable for 16-bit and 32-bit data. In total, 16 different rotation instructions are supplied:

Function	MELSEC Instruction in MELSEC Editor	IEC Instruction in IEC Editor
Data rotation to the right (16-bit)	ROR	ROR_M
	RORP	RORP_M
	RCR	RCR_M
	RCRP	RCRP_M
Data rotation to the left (16-bit)	ROL	ROL_M
	ROLP	ROLP_M
	RCL	RCL_M
	RCLP	RCLP_M
Data rotation to the right (32-bit)	DROR	DROR_M
	DRORP	DRORP_M
	DRCR	DRCR_M
	DRCRP	DRCRP_M
Data rotation to the left (32-bit)	DROL	DROL_M
	DROLP	DROLP_M
	DRCL	DRCL_M
	DRCLP	DRCLP_M

NOTE *Within the IEC editors please use the IEC instructions.*

7.2.1 ROR, RORP, RCR, RCRP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

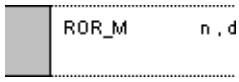
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	●	●	●	●	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2

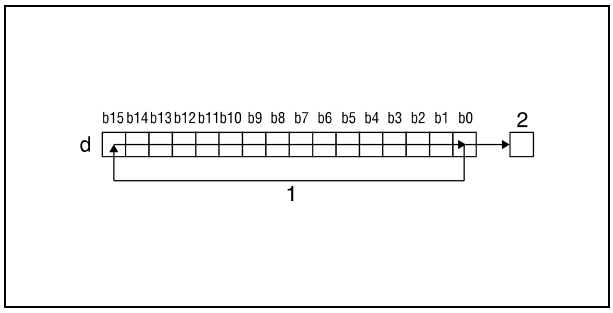
Variables

Set Data	Meaning	Data Type
d	First number of device performing data rotation	BIN 16-bit
n	Number of rotations (0 to 15)	

Functions **Data rotation to the right (16-bit)**

ROR Rotation instruction without carry flag

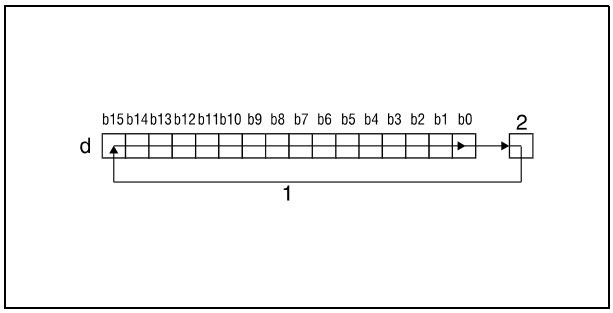
The ROR instruction rotates data bits in the device designated by d by n bits to the right. The carry flag (SM700) is not included. It retains the condition of the latest bit rotated from b0 to b15.



[1] Rotation by n bits
[2] Carry flag SM700

RCR Rotation instruction with carry flag

The RCR instruction rotates data bits in the device designated by d by n bits to the right, including the carry flag. The carry flag (SM700) retains the condition of the bit rotated by n bits. The condition of the carry flag (0 or 1) prior to the rotation is moved to the right within d by n bits beginning from b15.



[1] Rotation by n bits
[2] Carry flag SM700

NOTE *If a bit device is designated by d, the rotation operation is performed with a device supplying the specified number of digits. The number of digits the bits are rotated by is determined by the remainder of the following quotient:*

Number of rotations n / number of bits

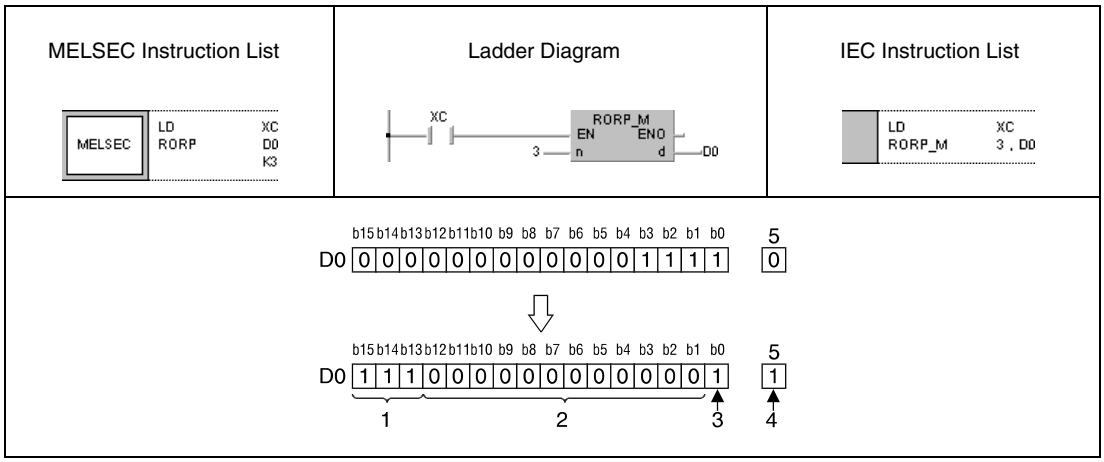
For example, 16 rotations of 12 bits correspond to a rotation by 4 bits, since the remainder of the quotient 16/12 equals 4. The reason for this is that a bit x in 12 bits after 12-fold rotation again reaches the same position prior to the rotation.

For this reason, specify a value in the range from 0 to 15 as n.

Program Example 1

RORP

With leading edge from XC, the following program rotates the contents of D0 by 3 bits to the right.



[1] Contents of bits b0–b2 before the rotation

[2] Contents of bits b4–b15 before the rotation

[3] Contents of bit b3 before the rotation

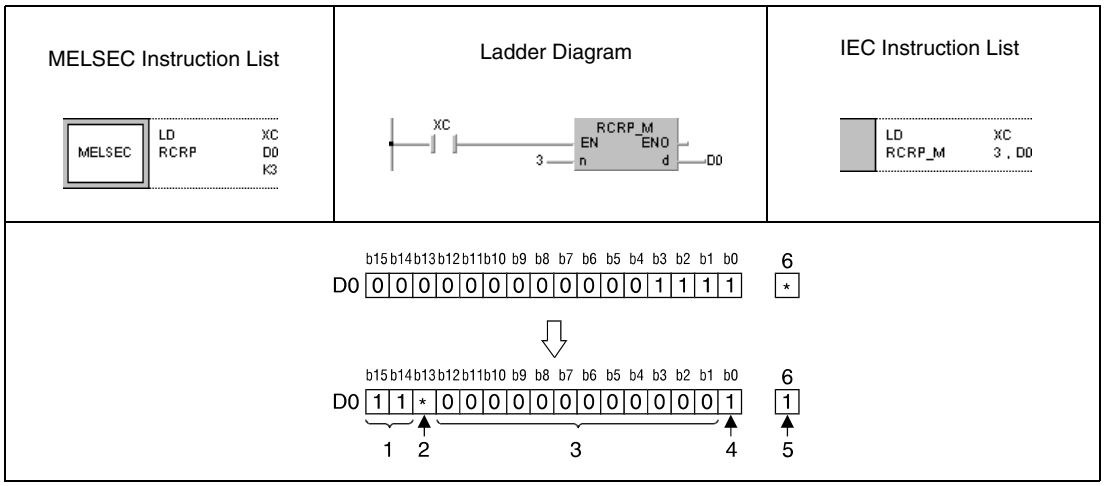
[4] Contents of bit b2 before the rotation

[5] Carry flag SM700

Program Example 2

RCRP

With leading edge from XC, the following program rotates the contents of D0 by 3 bits to the right; the carry flag SM700 is included. The condition of SM700 (0/1) prior to the rotation is moved to the right by 3 digits.



[1] Contents of bits b1 and b0 before the rotation

[2] Contents of carry flag before the rotation

[3] Contents of bits b4–b15 before the rotation

[4] Contents of bit b3 before the rotation

[5] Contents of bit b2 before the rotation

[6] Carry flag SM700

7.2.2 ROL, ROLP, RCL, RCLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

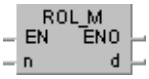
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	●	●	●	●	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

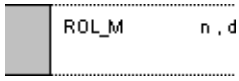
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

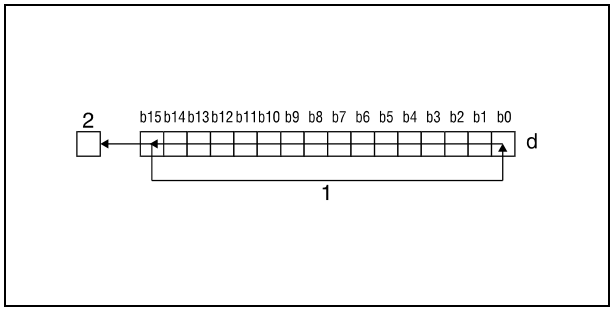
Variables

Set Data	Meaning	Data Type
d	First number of device performing data rotation	BIN 16-bit
n	Number of rotations (0 to 15)	

Functions **Data rotation to the left (16-bit)**

ROL Rotation instruction without carry flag

The ROL instruction rotates data bits in the device designated by d by n bits to the left. The carry flag (SM700) is not included. It retains the condition of the latest bit rotated from b0 to b15.

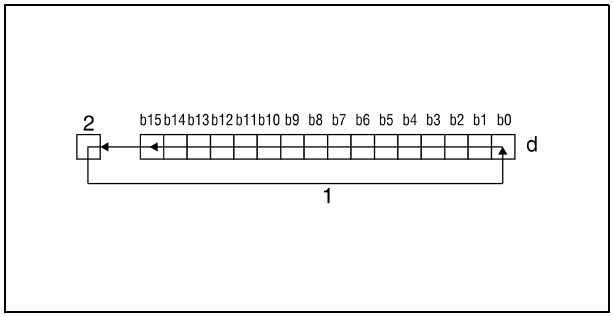


[1] Rotation by n bits

[2] Carry flag SM700

RCL Rotation instruction with carry flag

The RCL instruction rotates data bits in the device designated by d by n bits to the left, including the carry flag. The carry flag (SM700) retains the condition of the bit rotated by n bits. The condition of the carry flag (0 or 1) prior to the rotation is moved to the left within d by n bits beginning from b15.



[1] Rotation by n bits

[2] Carry flag SM700

NOTE *If a bit device is designated by d, the rotation operation is performed with a device supplying the specified number of digits. The number of digits the bits are rotated by is determined by the remainder of the following quotient:*

Number of rotations n / number of bits

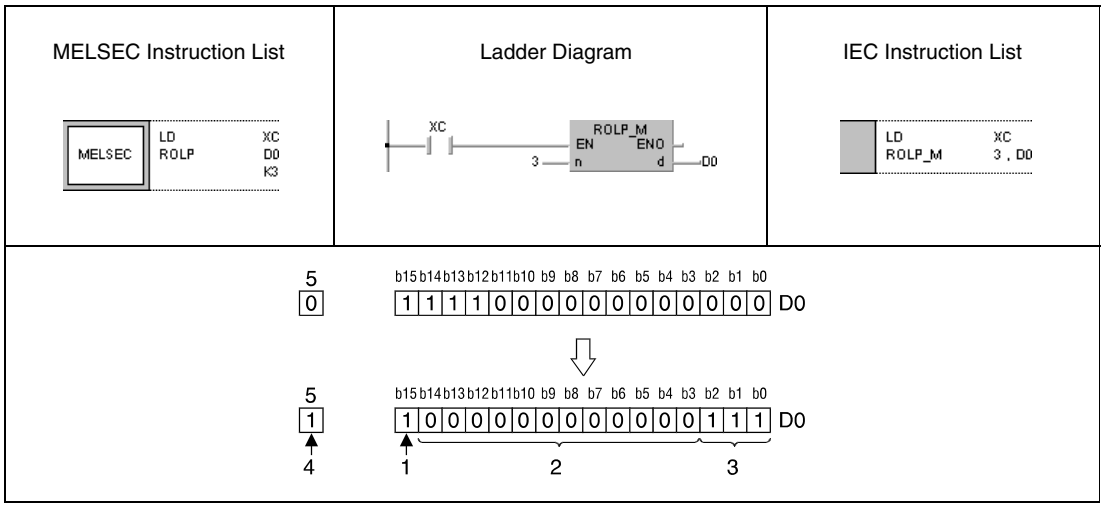
For example, 16 rotations of 12 bits correspond to a rotation by 4 bits, since the remainder of the quotient 16/12 equals 4. The reason for this is that a bit x in 12 bits after 12-fold rotation again reaches the same position prior to the rotation.

For this reason, specify a value in the range from 0 to 15 as n.

Program Example 1

ROLP

With leading edge from XC, the following program rotates the contents of D0 by 3 bits to the left.

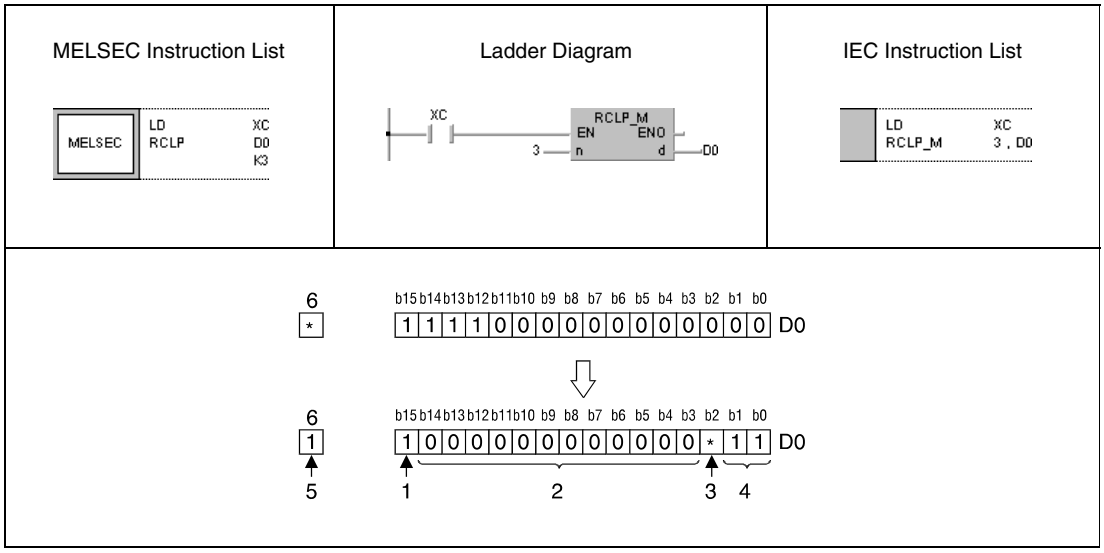


[1] Contents of bit b12 before the rotation
[2] Contents of bits b11–b0 before the rotation
[3] Contents of bits b15–b13 before the rotation
[4] Contents of bit b12 before the rotation
[5] Carry flag SM700

Program Example 2

RCLP

With leading edge from XC, the following program rotates the contents of D0 by 3 bits to the left; the carry flag SM700 is included. The condition of SM700 (0/1) prior to the rotation is moved to the left by 3 digits.



[1] Contents of bit b12 before the rotation
[2] Contents of bits b11–b0 before the rotation
[3] Contents of carry flag SM700
[4] Contents of bits b14 and b15 before the rotation
[5] Contents of carry flag SM700 before the rotation
[6] Carry flag SM700

7.2.3 DROR, DRORP, DRCR, DRCRP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

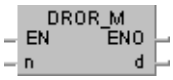
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	●	●	●	●	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

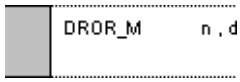
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

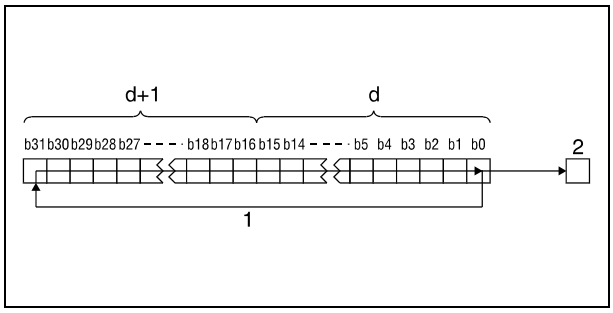
Set Data	Meaning	Data Type
d	First number of device performing data rotation	BIN 32-bit
n	Number of rotations (0 to 31)	BIN 16-bit

Functions

Data rotation to the right (32-bit)

DROR Rotation instruction without carry flag

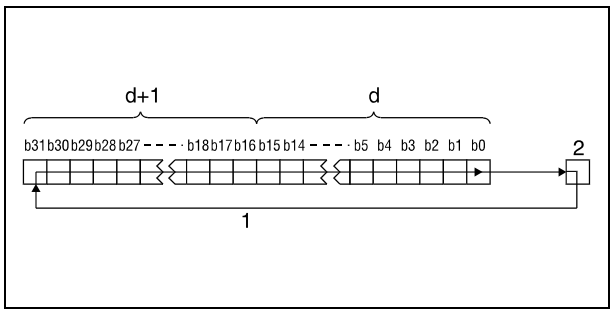
The DROR instruction rotates data bits in the device designated by d by n bits to the right. The carry flag (SM700) is not included. It retains the condition of the latest bit rotated from b0 to b31.



[1] Rotation by n bits
[2] Carry flag SM700

DRCR Rotation instruction with carry flag

The DRCR instruction rotates data bits in the device designated by d by n bits to the right, including the carry flag. The carry flag (SM700) retains the condition of the bit rotated by n bits. The condition of the carry flag (0 or 1) prior to the rotation is moved to the right within d (A0, A1) by n bits beginning from b31.



[1] Rotation by n bits
[2] Carry flag SM700

NOTE

If a bit device is designated by d, the rotation operation is performed with a device supplying the specified number of digits. The number of digits the bits are rotated by is determined by the remainder of the following quotient:

Number of rotations n / number of bits

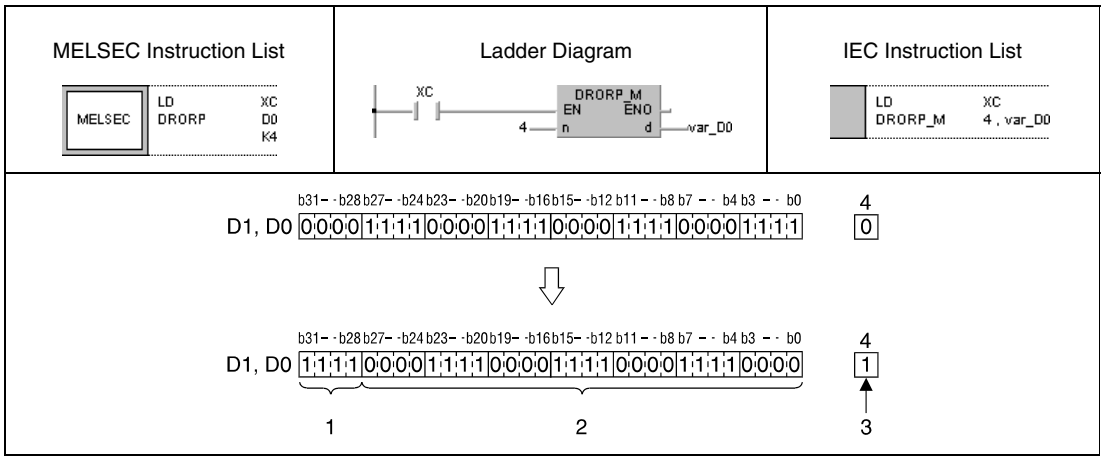
For example, 31 rotations of 24 bits correspond to a rotation by 7 bits, since the remainder of the quotient 31/24 equals 7. The reason for this is that a bit x in 24 bits after 24-fold rotation again reaches the same position prior to the rotation.

For this reason, specify a value in the range from 0 to 31 as n.

Program Example 1

DRORP

With leading edge from XC, the following program rotates the contents of D0 and D1 by 4 bits to the right.



[1] Contents of bits b3-b0 before the rotation

[2] Contents of bits b31-b4 before the rotation

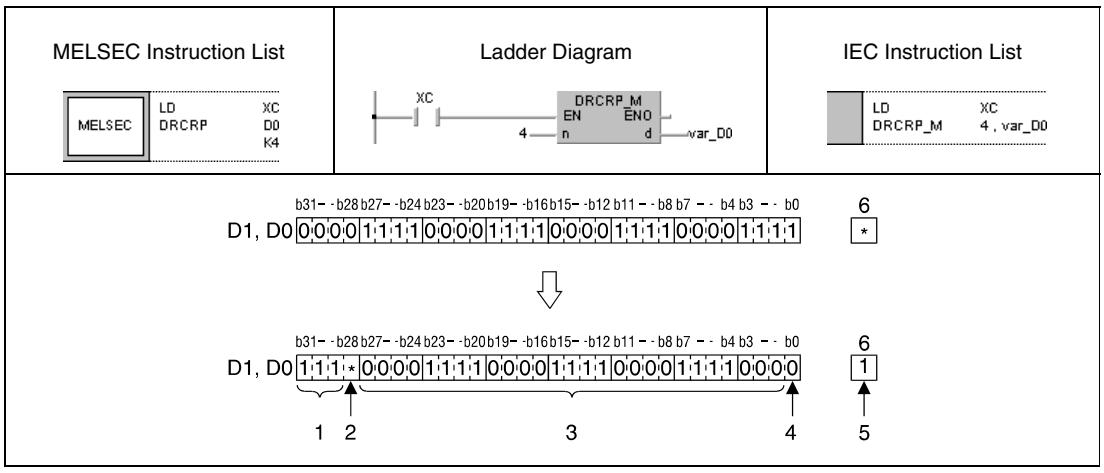
[3] Contents of bit b3 before the rotation

[4] Carry flag SM700

Program Example 2

DRCRP

With leading edge from XC, the following program rotates the contents of D0 and D1 by 4 bits to the right; the carry flag SM700 is included. The condition of SM700 (0/1) prior to the rotation is moved to the right by 4 digits.



[1] Contents of bits b2–b0 before the rotation

[2] Contents of carry flag SM700 before the rotation

[3] Contents of bits b5–b31 before the rotation

[4] Contents of bit b4 before the rotation

[5] Contents of bit b3 before the rotation

[6] Carry flag SM700

NOTE *These programs will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.2.4 DROL, DROLP, DRCL, DRCLP

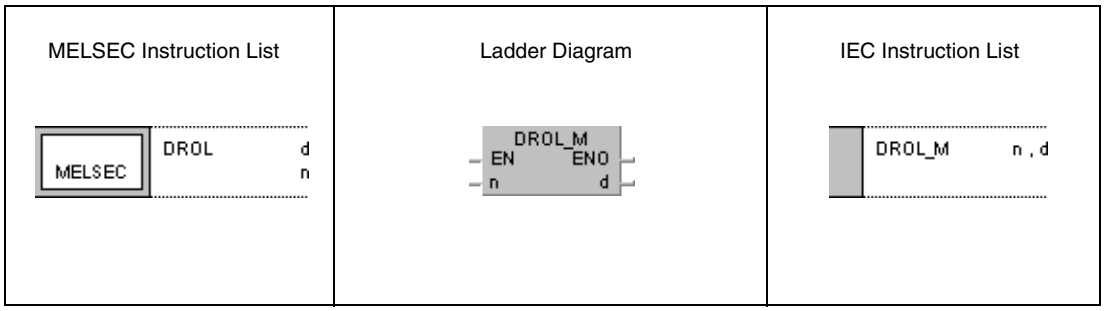
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

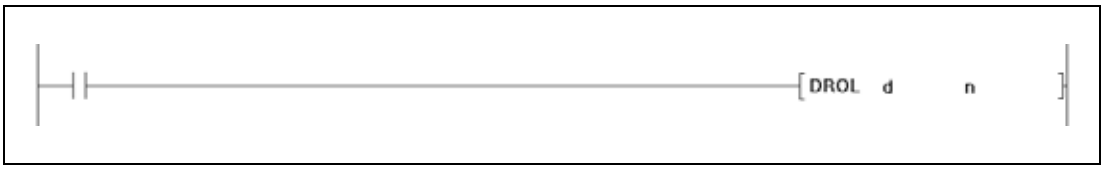
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
d	●	●	●	●	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer



GX Works2



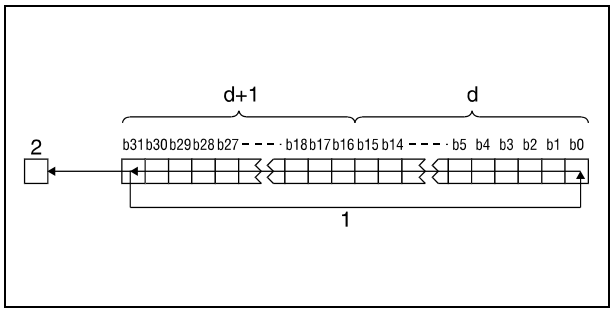
Variables

Set Data	Meaning	Data Type
d	First number of device performing data rotation	BIN 32-bit
n	Number of rotations (0 to 31)	BIN 16-bit

Functions **Data rotation to the left (32-bit)**

DROL Rotation instruction without carry flag

The DROL instruction rotates data bits in the device designated by d by n bits to the left. The carry flag (SM700) is not included. It retains the condition of the latest bit rotated from b31 to b0.

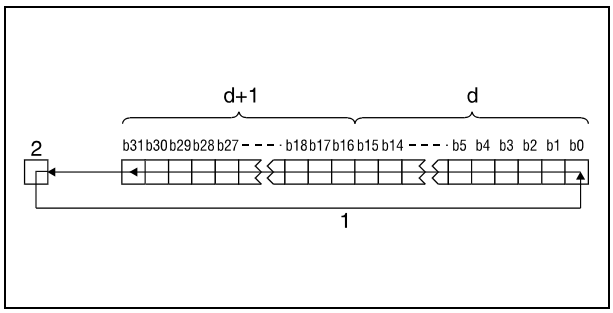


[1] Rotation by n bits

[2] Carry flag SM700

DRCL Rotation instruction with carry flag

The DRCR instruction rotates data bits in the device designated by d by n bits to the left, including the carry flag. The carry flag (SM700) retains the condition of the bit rotated by n bits. The condition of the carry flag (0 or 1) prior to the rotation is moved to the left within d (A0, A1) by n bits beginning from b31.



[1] Rotation by n bits

[2] Carry flag SM700

NOTE *If a bit device is designated by d, the rotation operation is performed with a device supplying the specified number of digits. The number of digits the bits are rotated by is determined by the remainder of the following quotient:*

Number of rotations n / number of bits

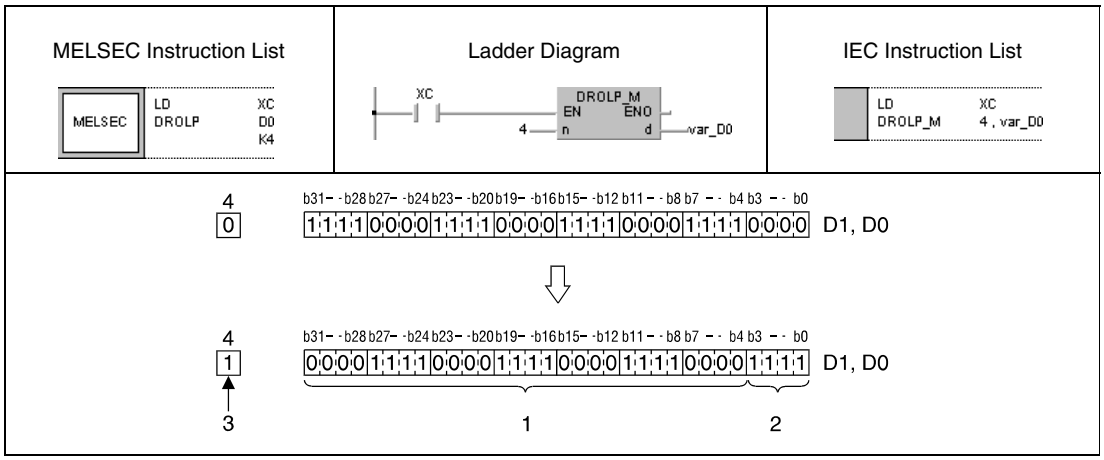
For example, 31 rotations of 24 bits correspond to a rotation by 7 bits, since the remainder of the quotient 31/24 equals 7. The reason for this is that a bit x in 24 bits after 24-fold rotation again reaches the same position prior to the rotation.

For this reason, specify a value in the range from 0 to 31 as n.

Program Example 1

DROLP

With leading edge from XC, the following program rotates the contents of D0 and D1 by 4 bits to the left.



[1] Contents of bits b27–b0 before the rotation

[2] Contents of bits b31–b28 before the rotation

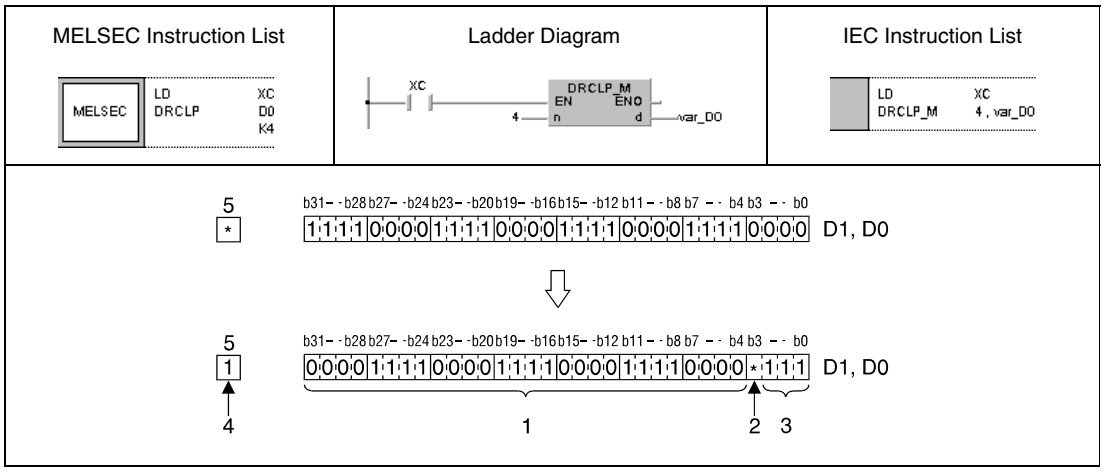
[3] Contents of bit b28 before the rotation

[4] Carry flag SM700

Program Example 2

DRCLP

With leading edge from XC, the following program rotates the contents of D0 and D1 by 4 bits to the left; the carry flag (SM700) is included. The condition of SM700 (0/1) prior to the rotation is moved to the left by 4 digits.



[1] Contents of bits b27–b0 before the rotation

[2] Contents of carry flag before the rotation

[3] Contents of bits b31–b29 before the rotation

[4] Contents of bit b28 before the rotation

[5] Carry flag SM700

NOTE

These programs will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.3 Data shift instructions

The shift instructions move data by bits or blocks of data within one data word. Data can be shifted to the right as well as to the left.

The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Shift a 16-bit data word by n bits	SFR	SFR_M
	SFRP	SFRP_M
	SFL	SFL_M
	SFLP	SFLP_M
Shift n bit devices by 1 bit	BSFR	BSFR_M
	BSFRP	BSFRP_M
	BSFL	BSFL_M
	BSFLP	BSFLP_M
Shift n bit devices by n bits	SFTBR	
	SFTBRP	
	SFTBL	
	SFTBLP	
Shift n word devices by one digit	DSFR	DSFR_M
	DSFRP	DSFRP_M
	DSFL	DSFL_M
	DSFLP	DSFLP_M
Shift n word devices by n words	SFTWR	
	SFTWRP	
	SFTWL	
	SFTWLP	

NOTE *Within the IEC editors please use the IEC instructions.*

7.3.1 SFR, SFRP, SFL, SFLP

CPU

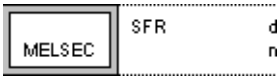
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	●	●	●	●	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

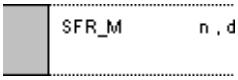
GX IEC Developer

MELSEC Instruction List

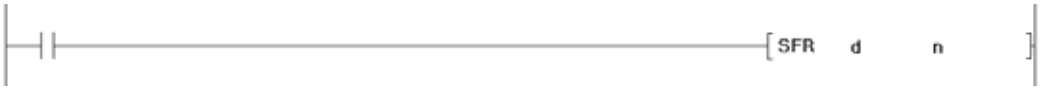


Ladder Diagram

IEC Instruction List



GX Works2



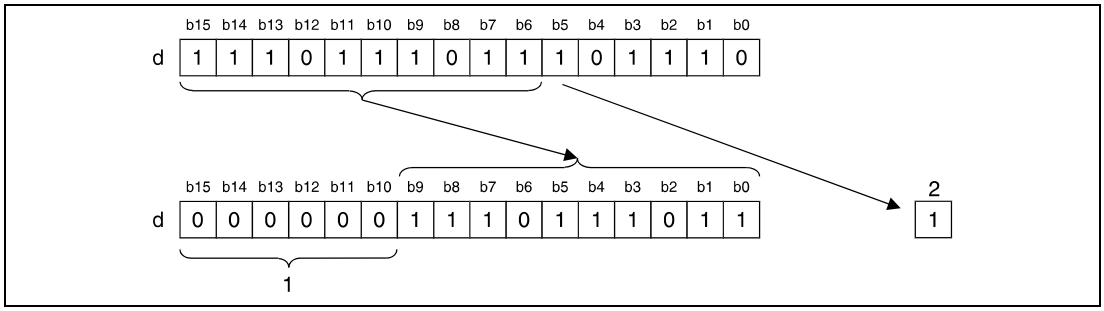
Variables

Set Data	Meaning	Data Type
d	First number of device storing data to be shifted	BIN 16-bit
n	Number of shiftings (0 to 15)	

Functions **Shifting a 16-bit data word by n bits**

SFR Shifting to the right

The SFR instruction shifts the 16-bit data word designated by d by n bits to the right.



[1] These bits are set to 0.

[2] Carry flag SM700

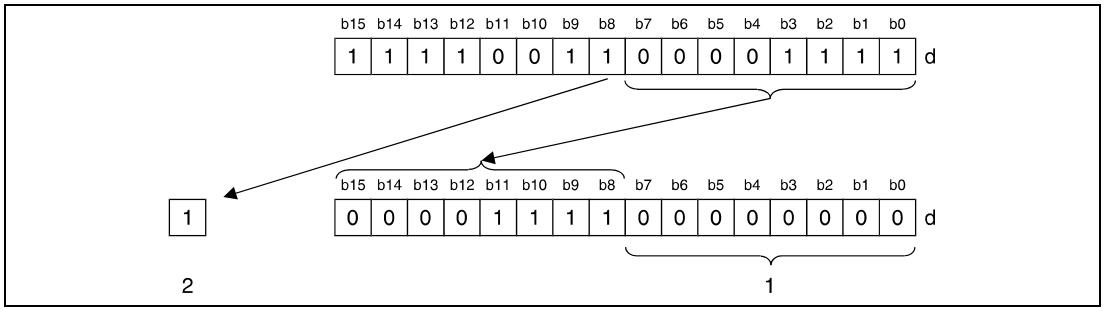
The most significant n bits beginning from bit b15 on are set to 0. The nth bit (b(n-1)) to be shifted is moved to the carry flag (SM700).

For timers and counters, the actual value (count) is shifted. The setting value cannot be shifted.

For bit devices, shifting within a device with a specified number of bits is feasible (see program example 1).

SFL Shifting to the left

The SFL instruction shifts the 16-bit data word designated by d by n bits to the left.



[1] These bits are set to 0.

[2] Carry flag SM700

The least significant n bits beginning from bit b0 on are set to 0. The nth bit (b(15-n)) to be shifted is moved to the carry flag (SM700).

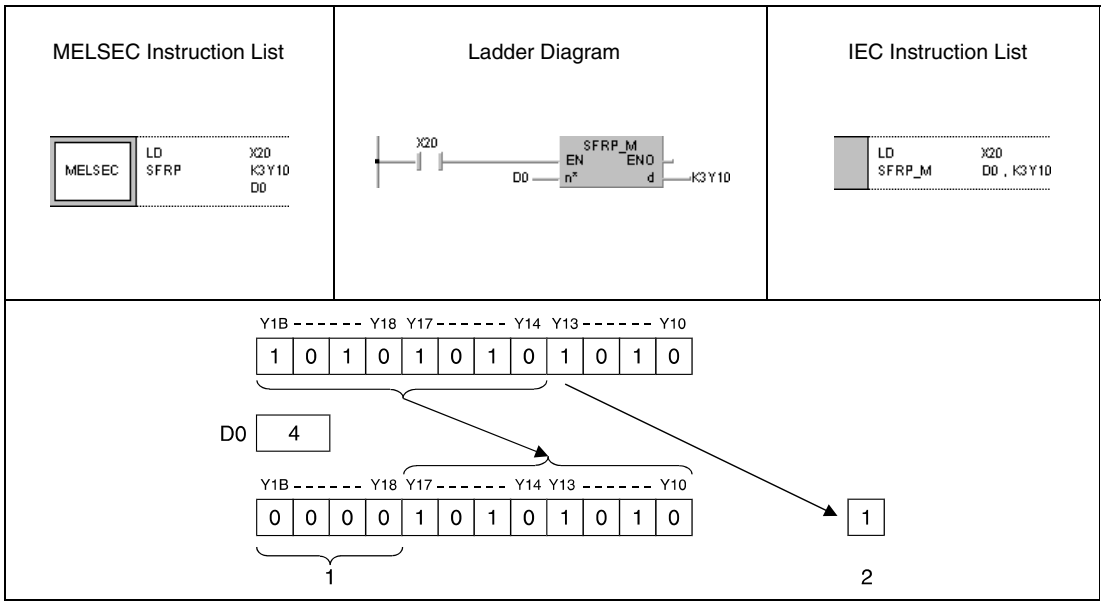
For timers and counters, the actual value (count) is shifted. The setting value cannot be shifted.

For bit devices, shifting within a device with a specified number of bits is feasible (see program example 1).

Program Example 1

SFRP

With leading edge from X20, the following program shifts the content of Y10 through Y1B by the number of bits specified by D0 to the right. The condition of Y13 is stored in the carry flag (SM700).



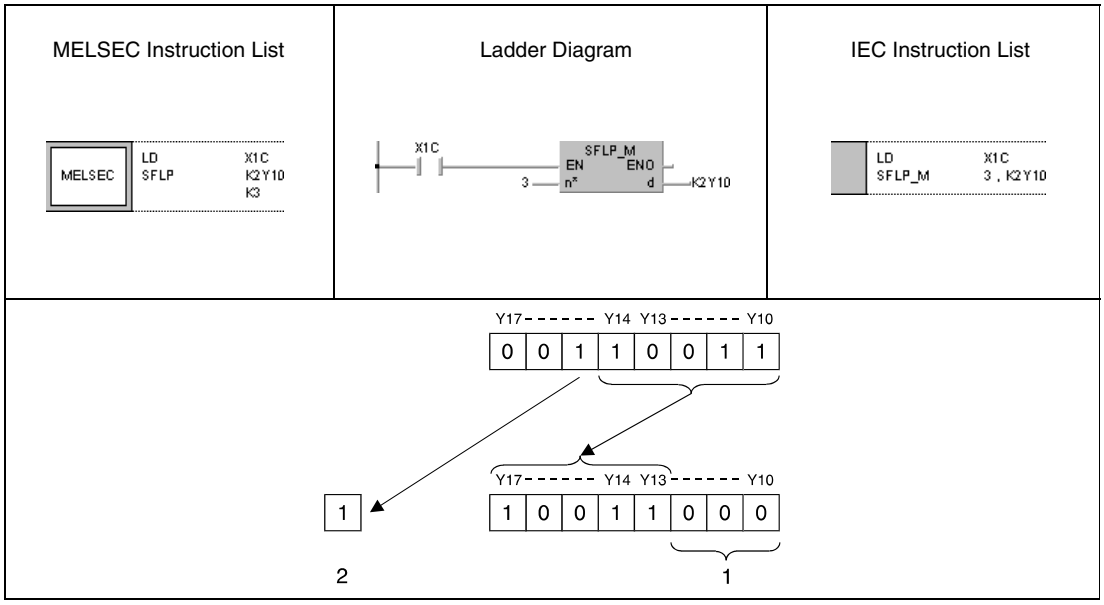
[1] These bits are set to 0.

[2] Carry flag SM700

Program Example 2

SFLP

With leading edge from X1C, the following program shifts the content of Y10 through Y18 by 3 bits to the left. The condition of Y15 is stored in the carry flag (SM700).



[1] These bits are set to 0.

[2] Carry flag SM700

7.3.2 BSFR, BSFRP, BSFL, BSFLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

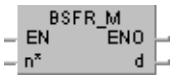
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				U
d	●	—	—	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

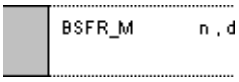
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

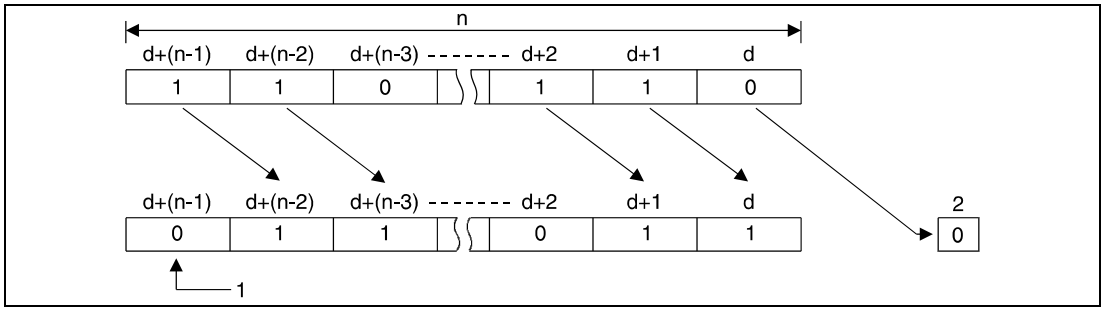
Set Data	Meaning	Data Type
d	First number of device to be shifted	Bit
n	Number of devices to be shifted	BIN 16-bit

Functions

Shifting n bit devices by 1 bit

BSFR Shifting to the right

The BSFR instruction shifts the contents of specified bit devices by 1 bit to the right. The shift operation starts from the address of the device designated by d and is proceeded for the following n addresses.

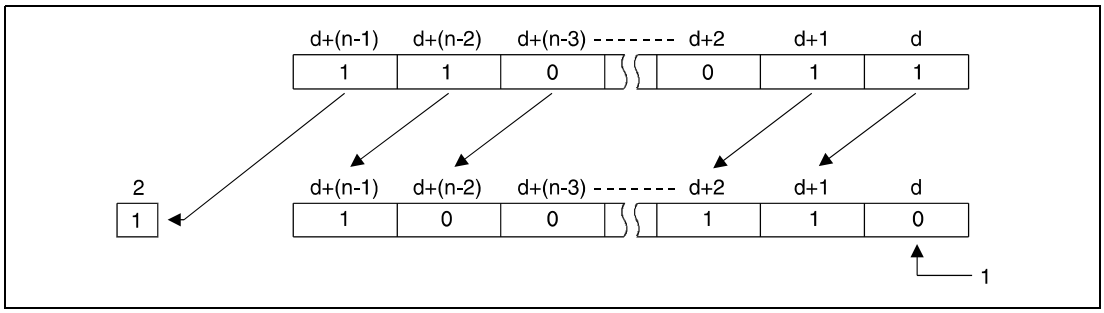


[1] This bit is set to 0.

[2] Carry flag SM700

BSFL Shifting to the left

The BSFL instruction shifts the contents of specified bit devices by 1 bit to the left. The shift operation starts from the address of device designated by d and is proceeded for the following n addresses.



[1] This bit is set to 0.

[2] Carry flag SM700

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

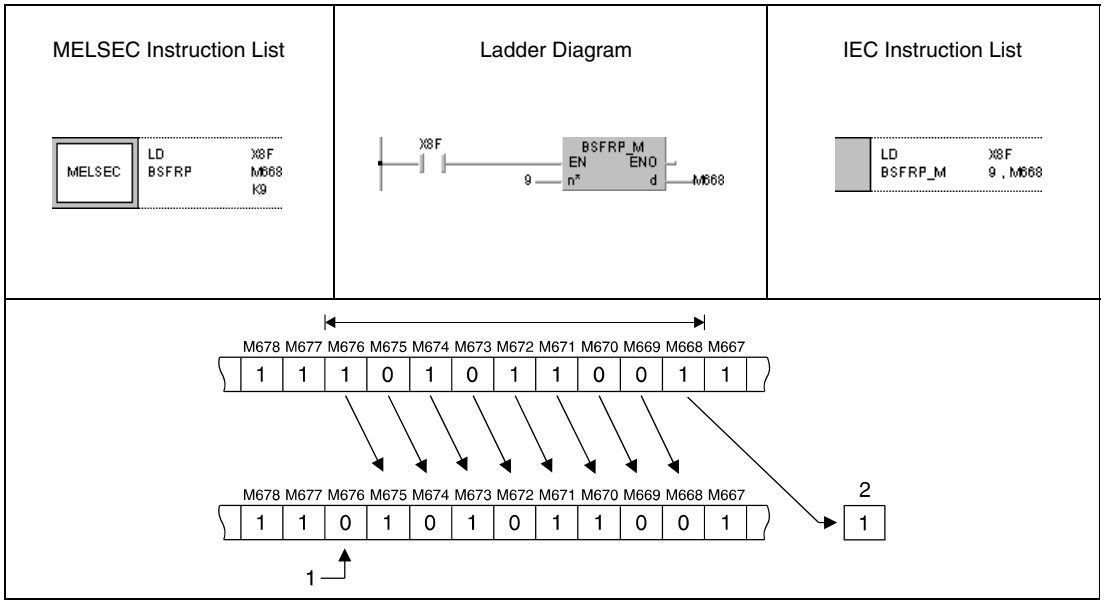
- The value in n exceeds the available number of bits in the device designated by d. (Error code 4101)

Program Example 1

BSFRP

With leading edge from X8F, the following program shifts data of the internal relays M668 through M676 by one bit to the right. M668 retains the value of M669, M669 that of M670 etc.

The contents of the first device (M668) is written to the carry flag (SM700), and the last device (M676) retains the value 0.



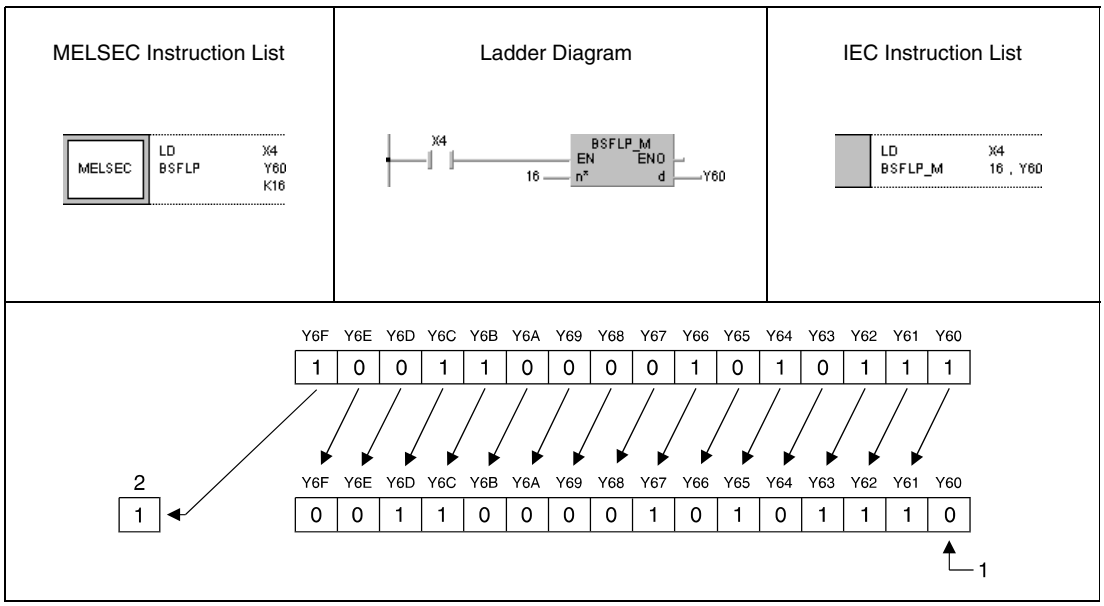
[1] This bit is set to 0.

[2] Carry flag SM700

Program Example 2

BSFLP

With leading edge from X4, the following program shifts the contents of the outputs Y60 through Y6F by one device to the left. The contents of the last output (Y6F) is stored in the carry flag (SM700), and the first output (Y60) is reset to 0.



[1] This bit is set to 0.

[2] Carry flag SM700

7.3.3 SFTBR, SFTBRP, SFTBL, SFTBLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

Devices

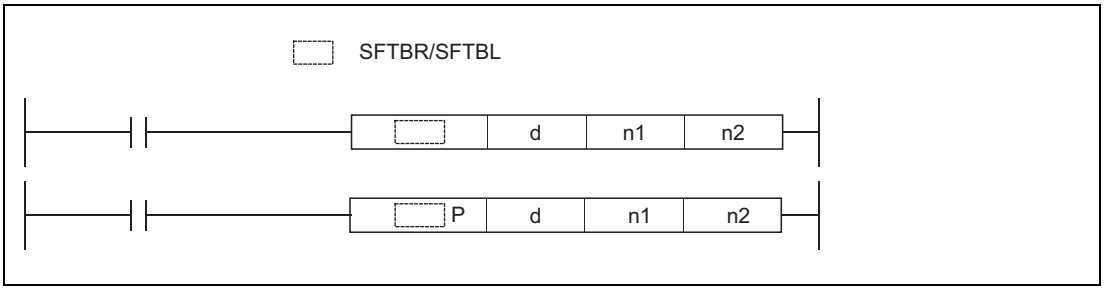
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
d	●[1)]	—	●	—	—	—	—	—	—
n1	—	●	●	●	●	●	●	●	—
n2	—	●	●	●	●	●	●	●	—

[1] Except T, C, ST and S

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



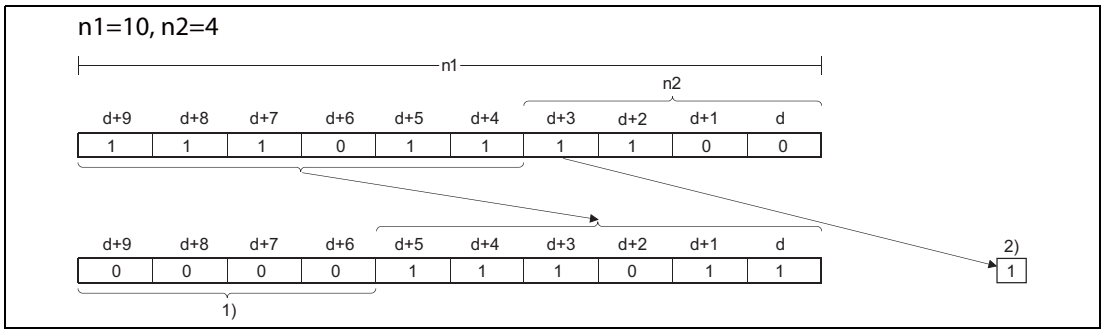
Variables

Set Data	Meaning	Data Type
d	First number of device to be shifted	Bit
n1	Number of bits to be shifted	BIN 16-bit
n2	Number of shifts	

Functions **Shifting n bit devices by n bits**

SFTBR Shifting to the right

This instruction shifts the n1 bits data in the devices starting from the device specified by d to the right by n2 bits.



[1] These bits are set to 0.

[2] Carry flag SM700

n1 and n2 are specified under the condition that n1 is larger than n2. If the value of n2 is equal to or larger than the value of n1, the remainder of n2 / n1 (n2 devided by n1) is used for a shift.

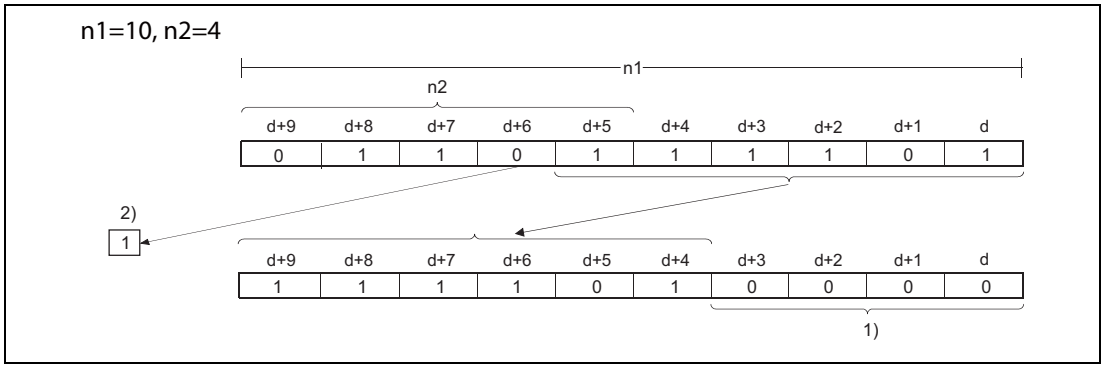
This instruction specifies n1 ranged from 1 to 64.

Bits starting from the highest bit to n2th bit are filled with 0s. If the value of n2 is larger than the value of n1, the remainder of n2 / n1 will be 0.

If the value specified by n1 or n2 is 0, the instruction will be not processed.

SFTBL Shifting to the left

This instruction shifts the n1 bits data in the devices starting from the device specified by to the left by n2 bits.



[1] These bits are set to 0.

[2] Carry flag SM700

n1 and n2 are specified under the condition that n1 is larger than n2. If the value of n2 is equal to or larger than the value of n1, the remainder of n2 / n1 (n2 devided by n1) is used for a shift. However, if the remainder of n2 / n1 is 0, the instruction will be not processed.

This instruction specifies n1 ranged from 1 to 64.

Bits starting from the highest bit to n2th bit are filled with 0s. If the value of n2 is larger than the value of n1, the remainder of n2 / n1 will be 0.

If the value specified by n1 or n2 is 0, the instruction will be not processed.

Operation Errors

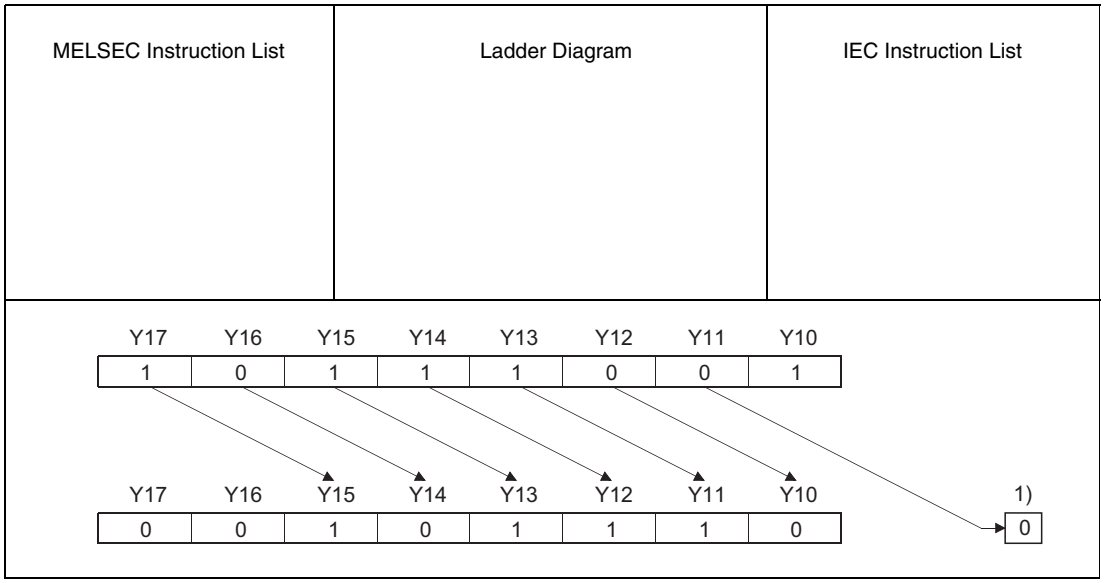
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value specified by n1 is other than 0 to 64. (Error code 4100)
- The value in n2 is negative. (Error code 4100)
- The value in n1 exceeds the available number of bits in the device designated by d. (Error code 4101)

Program Example 1

SFTBRP

The following program shifts the data of Y10 to Y17 (8 bits) specified by d to the right by 2 bits (n2), when M0 is turned on.

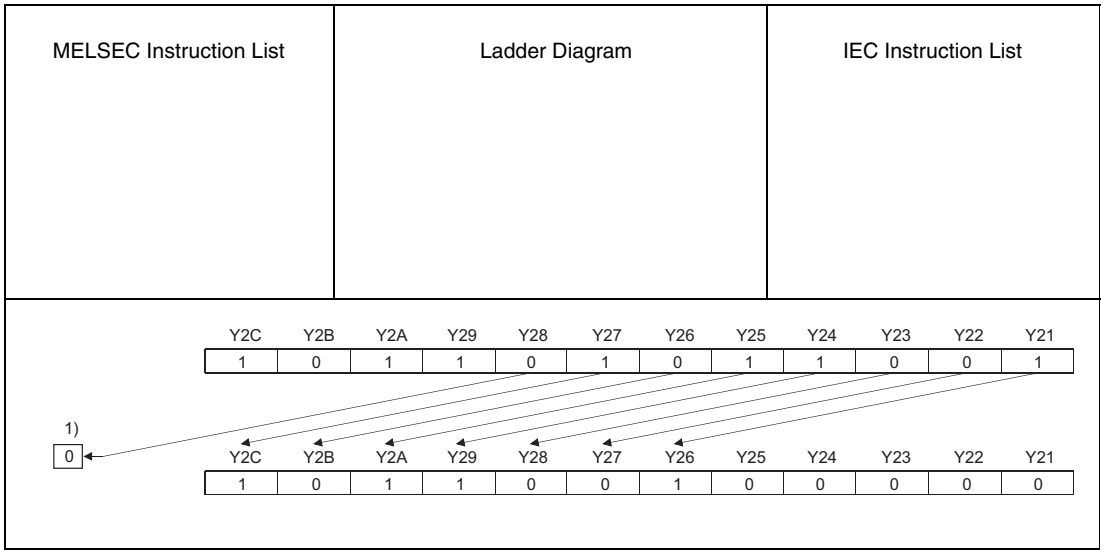


[1] Carry flag SM700

Program Example 2

SFTBLP

The following program shifts the data of Y21 to Y2C (12 bits) specified by d to the left by 5 bits (n2), when M0 is turned on.



[1] Carry flag SM700

7.3.4 DSFR, DSFRP, DSFL, DSFLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

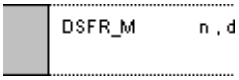
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
d	First number of device to be shifted	BIN 16-bit
n	Number of devices to be shifted	BIN 16-bit

Functions

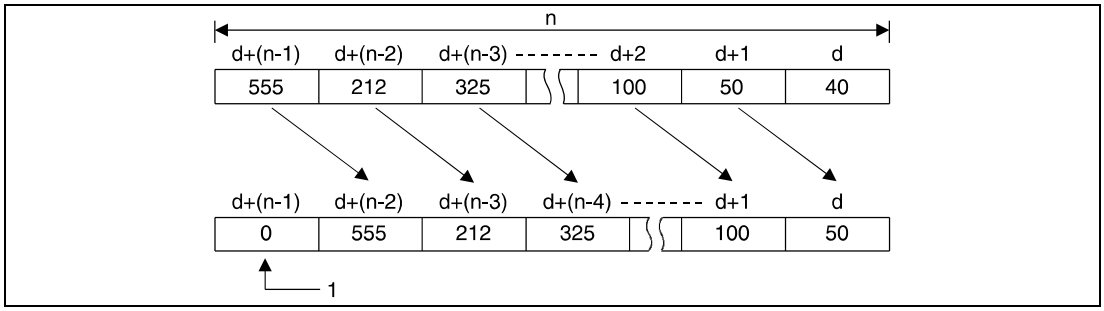
Shifting n word devices by 1 address

DSFR Shifting to the right

The DSFR instruction shifts the contents of specified word devices by one address to the right. The shift operation starts from the address designated by d and is proceeded for the following n addresses.

The contents of the most significant device is reset to 0 after the shifting.

For timers and counters, the actual value (count) is shifted. The setting value cannot be shifted.



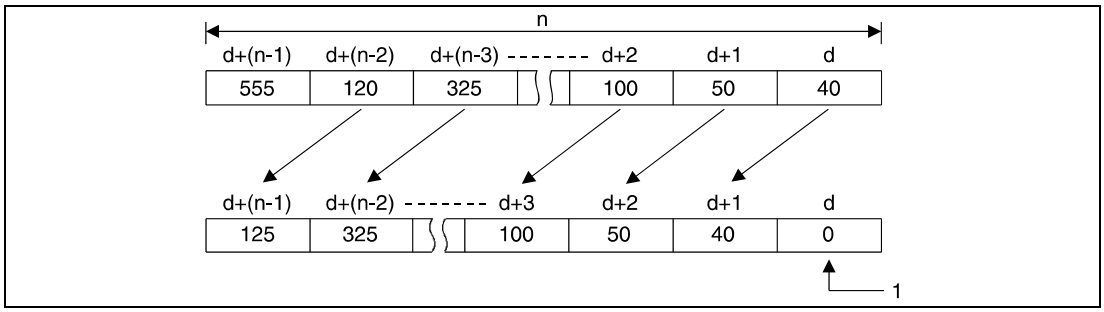
[1] This device is set to 0.

DSFL Shifting to the left

The DSFR instruction shifts the contents of specified word devices by one address to the left. The shift operation starts from the address designated by d and is proceeded for the following n addresses.

The contents of the least significant device is reset to 0 after the shifting.

For timers and counters, the actual value (count) is shifted. The setting value cannot be shifted.



[1] This device is set to 0.

Operation Errors

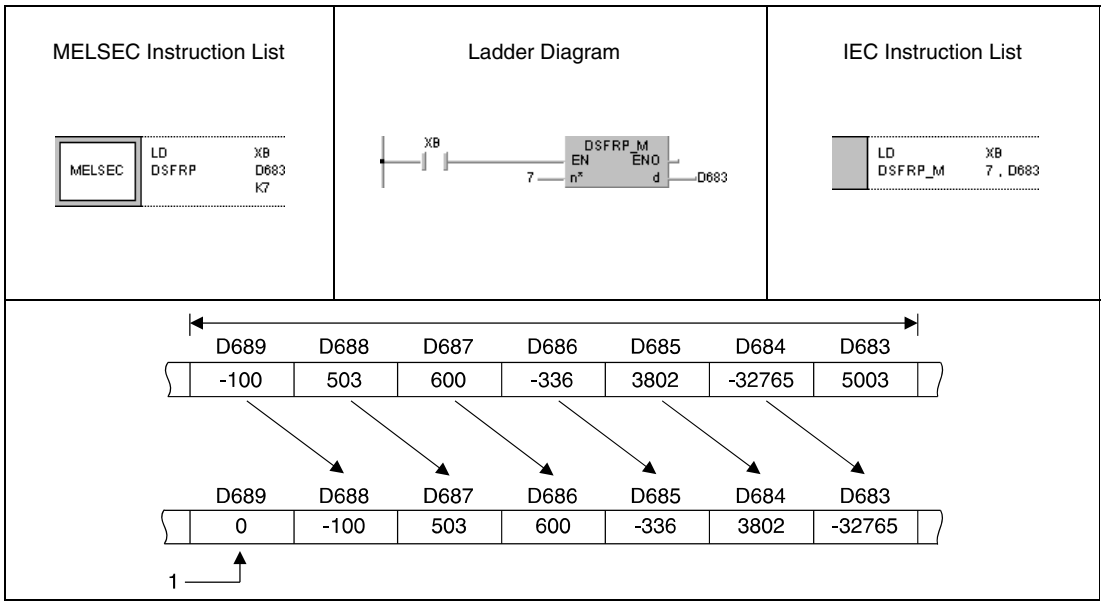
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value in n exceeds the available number of points in the device designated by d. (Error code 4101)

Program Example 1

DSFRP

With leading edge from XB, the following program shifts data in the data registers D683 through D689 by one address to the right. D683 retains the value of D684, D684 that of D685 etc. The contents of the last data register (D689) retains the value 0.

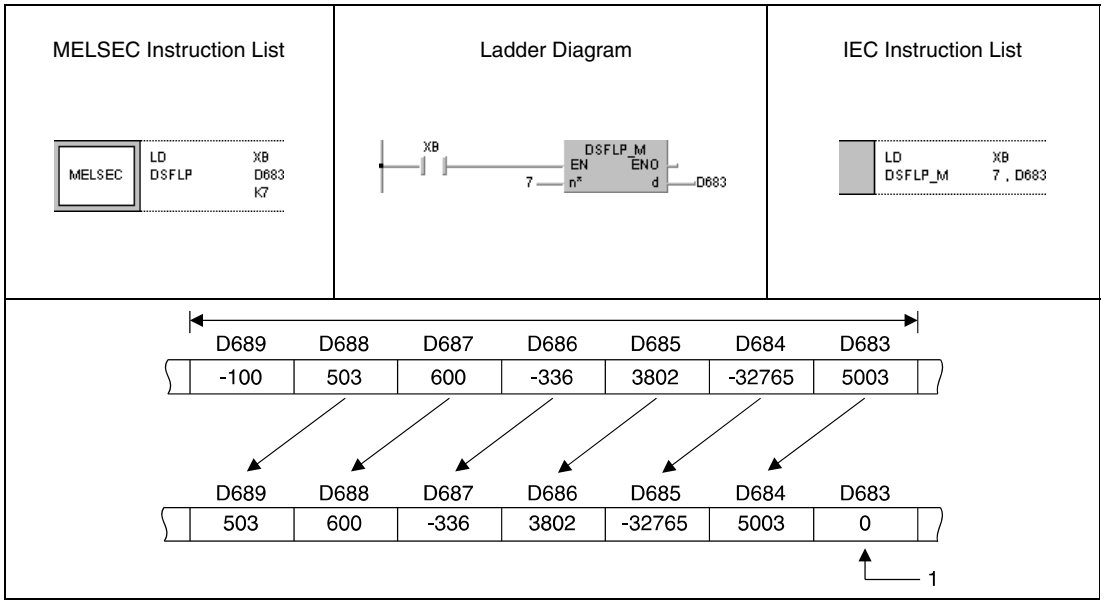


[1] This device is set to 0.

Program Example 2

DSFLP

With leading edge from XB, the following program shifts data in the data registers D683 through D689 by one address to the left. D689 retains the value of D688, D688 that of D687 etc. The contents of the first data registers (D683) retains the value 0.



[1] This device is set to 0.

7.3.5 SFTWR, SFTWRP, SFTWL, SFTWLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

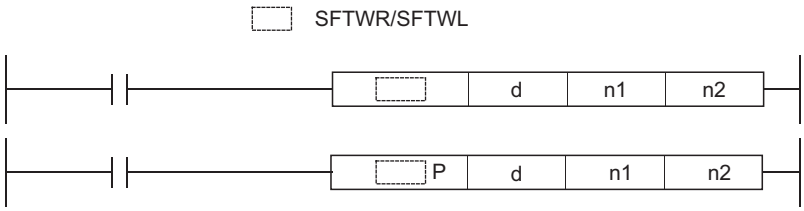
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
d	—	●	●	—	—	—	—	—	—
n1	—	●	●	●	●	●	●	●	—
n2	—	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



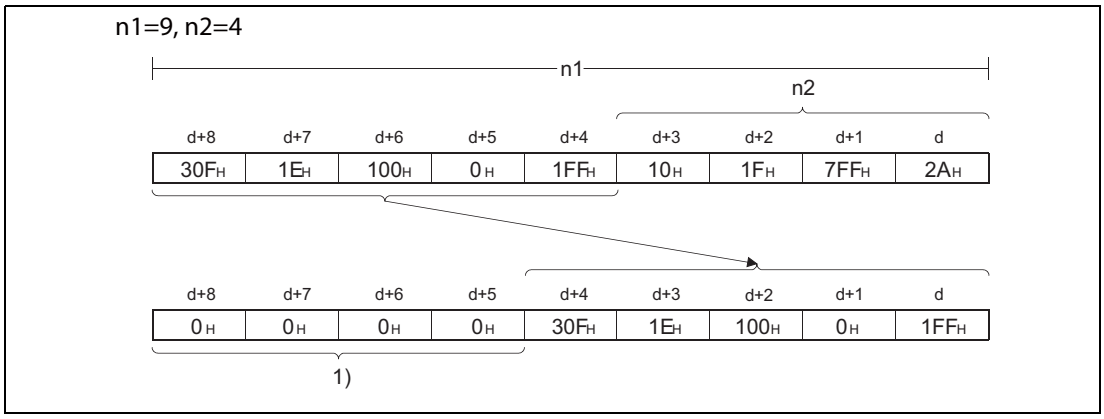
Variables

Set Data	Meaning	Data Type
d	First number of device to be shifted	BIN 16-bit
n1	Number of words to be shifted	BIN 16-bit
n2	Number of shifts	

Functions **Shifting n word devices by n words**

SFTWR **Shifting to the right**

This instruction shifts n1 words data in the devices starting from the device specified by d to the right by n2 words.



[1] Set to 0H

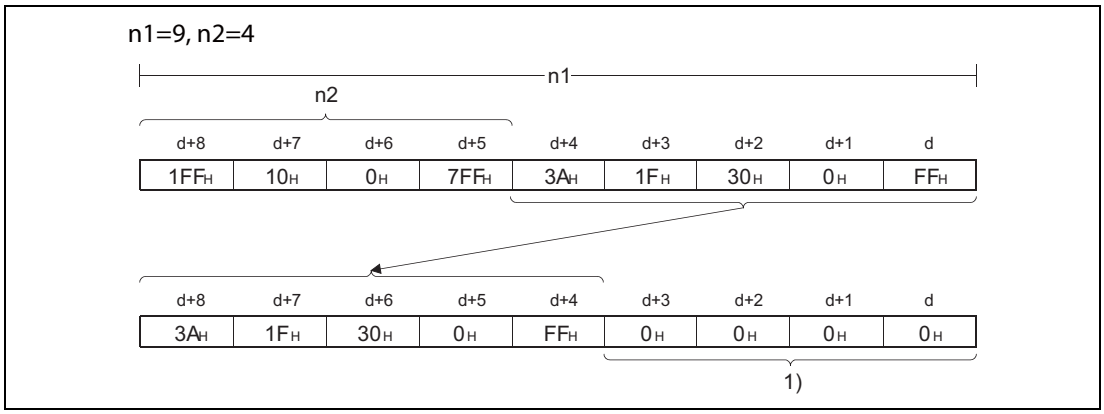
The n2 words data in the devices starting from the highest device are filled with 0s.

If the value specified by n1 or n2 is 0, the instruction will be not processed.

If the value of n2 is equal to or larger than the value of n1, the n1 words data in the devices starting from the device specified by d will be filled with 0s.

SFTWL **Shifting to the left**

This instruction shifts the n1 words data in the devices starting from the device specified by d to the left by n2 words.



[1] Set to 0H

The n2 words data in the devices starting from the lowest device are filled with 0s.

If the value specified by n1 or n2 is 0, the instruction will be not processed.

If the value of n2 is equal to or larger than the value of n1, the n1 words data in the devices starting from the device specified by d will be filled with 0s.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

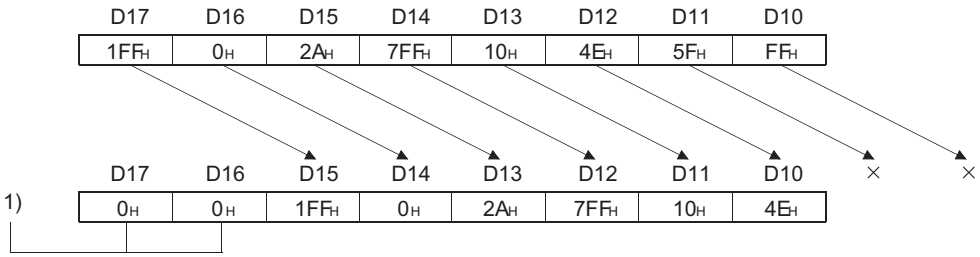
- The value in n1 or n2 is negative. (Error code 4100)
- The range of devices specified by n1 exceeds the range of devices specified by d. (Error code 4101)

Program Example 1

SFTWRP

The following program shifts the 8 words (n1 = 8) data stored in the devices starting from D10 specified by d to the right by 2 words (n2 = 2), when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



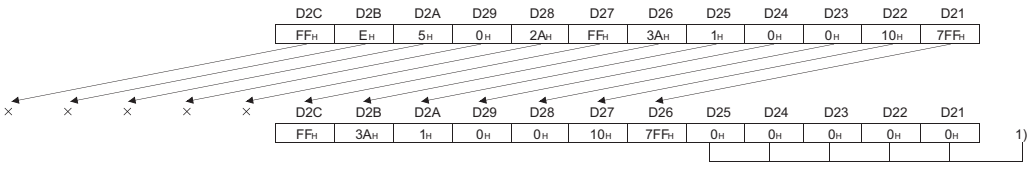
[1] Set to 0H

Program Example 2

SFTWLP

The following program shifts the 12 words (n1 = 12) data stored in the devices starting from D21 specified by d to the left by 5 words (n2 = 5), when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



[1] Set to 0H

7.4 Bit processing instructions

The bit processing instructions change the condition (set and reset) of single bits or entire sections of bits. The condition of bits in data words can as well be tested with the bit processing instructions.

The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Set / reset single bits	BSET	BSET_M
	BSETP	BSETP_M
	BRST	BRST_M
	BRSTP	BRSTP_M
Test condition of single bits in 16-/32-bit data words	TEST	TEST_M
	TESTP	TESTP_M
	DTEST	DTEST_M
	DTESTP	DTESTP_M
Reset sections of bits in a batch	BKRST	BKRST_M
	BKRSTP	BKRSTP_M

7.4.1 BSET, BSETP, BRST, BRSTP

CPU

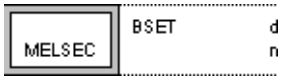
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

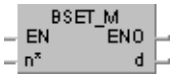
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	●	●	●	●	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

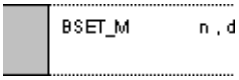
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

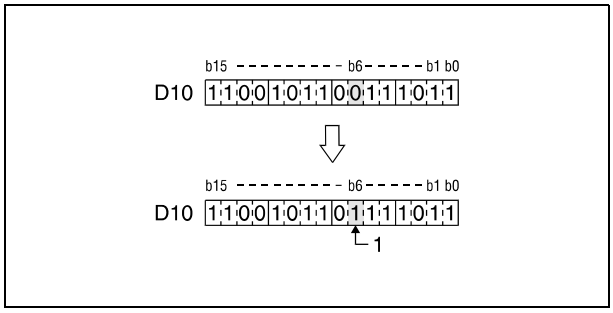
Variables

Set Data	Meaning	Data Type
d	Device storing bits to be set or reset	BIN 16-bit
n	Number of bit to be set or reset	

Functions **Setting / resetting single bits**

BSET Setting single bits of a word device

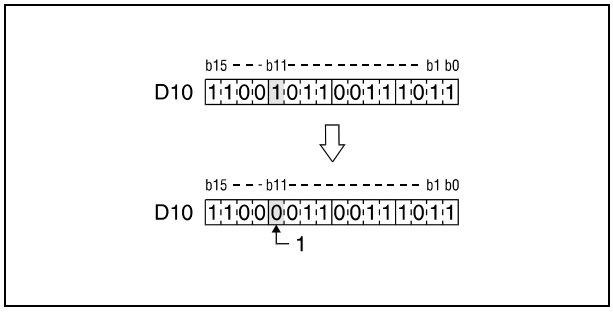
The BSET instruction sets the nth bit of a word device to 1. For n, a value between 0 and 15 (b0 to b15) can be specified. The word device is designated by d. If the value in n exceeds 15, the BSET instruction is executed within the lower 4 bits (b0 to b3). In the following diagram n is set to 6, so bit b6 is set.



[1] This bit is set.

BRST Resetting single bits in a word device

The BRST instruction resets the nth bit of a word device to 0. For n, a value between 0 and 15 (b0 to b15) can be specified. The word device is designated by d. If the value in n exceeds 15, the BRST instruction is executed within the lower 4 bits (b0 to b3). In the following diagram n is set to 11, so bit b11 is reset.



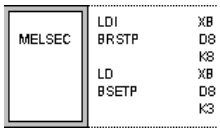
[1] This bit is reset.

Program Example

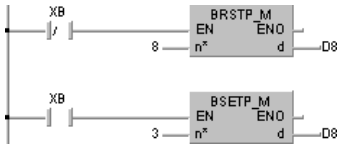
BRSTP/BSETP

The following program resets the 8th bit of D8 (b8) to 0 when the input XB is switched OFF, and sets the 3rd bit of D8 (b3) to 1 when XB is switched ON.

MELSEC Instruction List

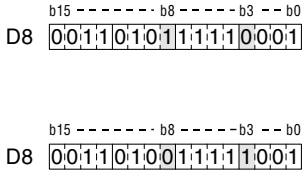


Ladder Diagram



IEC Instruction List

```
LDN     XB
BRSTP_M 8 , D8
LD      XB
BSETP_M 3 , D8
```



NOTE *Single bits in bit devices can be set or reset via a SET or an RST instruction as well. In this case the bits of the word device must be specified. For example, the bit (b8) in data word D5 is addressed as D5.8.*

7.4.2 TEST, TESTP, DTEST, DTESTP

CPU

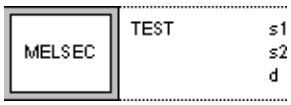
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

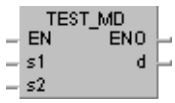
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	—	—
s2	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	—	—	—

GX IEC Developer

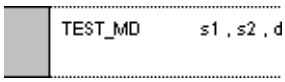
MELSEC Instruction List



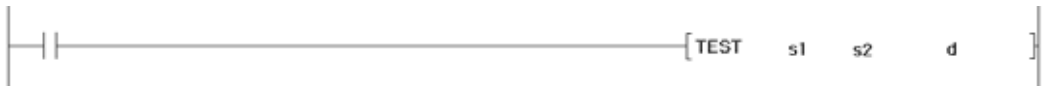
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s1	Number of device storing bits to be tested	Word
s2	Number of bit to be tested	Word
d	Number of bit device storing condition of tested bit	Bit

Functions **Test of single bits in 16- / 32-bit data words**

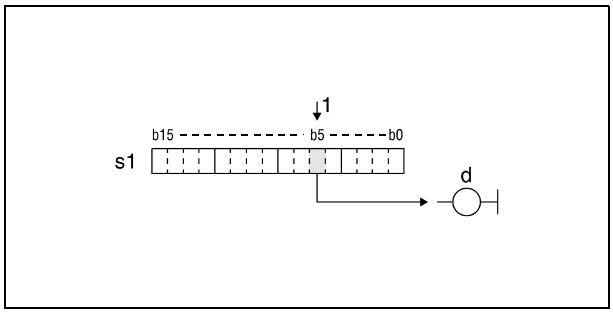
TEST Bit test 16-bit

The TEST instruction checks the condition of a bit s2 in a word device s1. The test result is stored in a bit device designated by d.

The device designated by d is set, if the tested bit is in condition 1, and reset, if the tested bit is in condition 0.

The bit specified by s2 can be any bit between b0 and b15 in a 16-bit data word. When 16 or more is designated at s2, the target is the bit data at the position indicated by the remainder of s2 / 16. For example, when s2 = 18, the target is the data at b2 since the remainder of 18 / 16 is "2".

In the following diagram s2 is set to 5, so the condition of bit b5 in s1 is tested.



[1] Tested bit

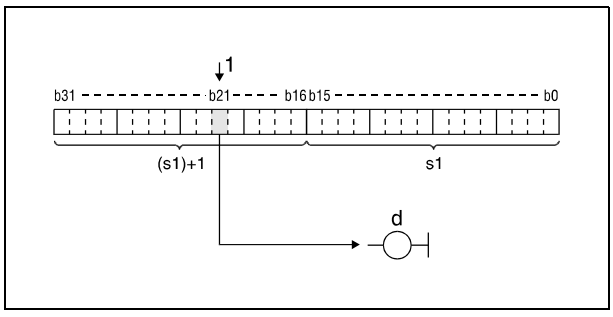
DTEST Bit test 32-bit

The DTEST instruction checks the condition of a bit s2 in a word device s1 and (s1)+1. The test result is stored in a bit device designated by d.

The device designated by d is set, if the tested bit is in condition 1, and reset, if the tested bit is in condition 0.

The bit specified by s2 can be any bit between b0 and b31 in a 32-bit data word. When 32 or more is designated at s2, the target is the bit data at the position indicated by the remainder of s2 / 32. For example, when s2 = 34, the target is the data at b2 since the remainder of 34 / 16 is "2".

In the following diagram s2 is set to 21, so the condition of bit b21 in s1 is tested.

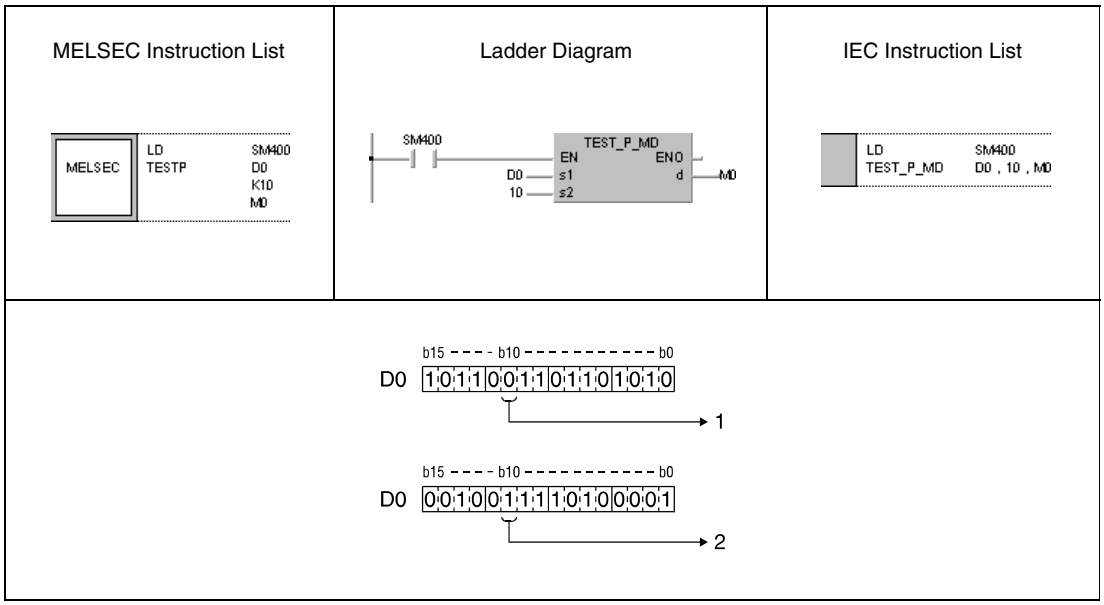


[1] Tested bit

Program Example 1

TESTP

Depending on the test result of the bit (b10) in the 16-bit data word in D0, the following program either resets or sets relay M0.



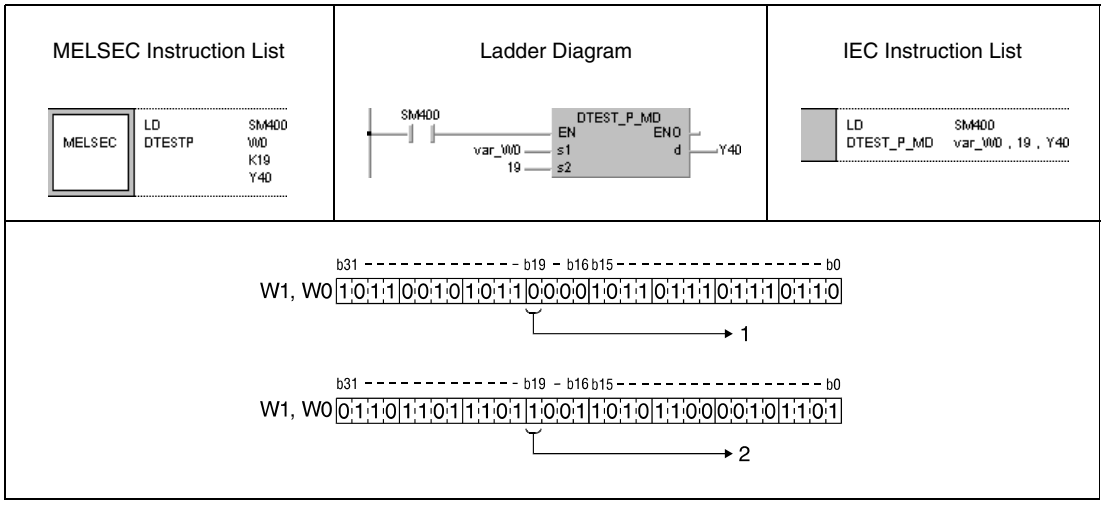
[1] Reset M0

[2] Set M0

Program Example 2

DTESTP

Depending on the test result of the bit (b19) in the 32-bit data word in W0 and W1, the following program either resets or sets output Y40.



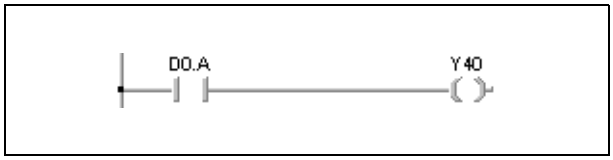
[1] Reset Y40

[2] Set Y40

NOTE

The program example 2 will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

Instead of applying the TEST instruction, a bit to be tested can also be specified as an input contact (see diagram).



7.4.3 BKRST, BKRSTP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

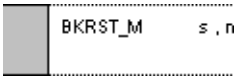
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
s	First number of device to be reset	Bit
n	Number of devices to be reset	BIN 16-bit

Functions

Batch reset of bits

BKRST Reset instruction

The BKRST instruction resets n bits in the device designated by s.

For annunciators (F), the number n of annunciators stored in s is reset and the contents of the registers SD64 through SD79 is cleared according to the reset annunciators. The remaining data are shifted forward. Moreover, the number of annunciator entries in registers SD64 through SD79 is stored in register SD63.

For timers (T) and counters (C), after the execution of this instruction the setting values of n timers and counters are reset to 0 and the inputs and outputs are reset.

For all other bit devices the number n of the devices starting from the device designated by s are reset.

If the according device is already reset, its condition remains unchanged after execution of the instruction.

Operation Errors

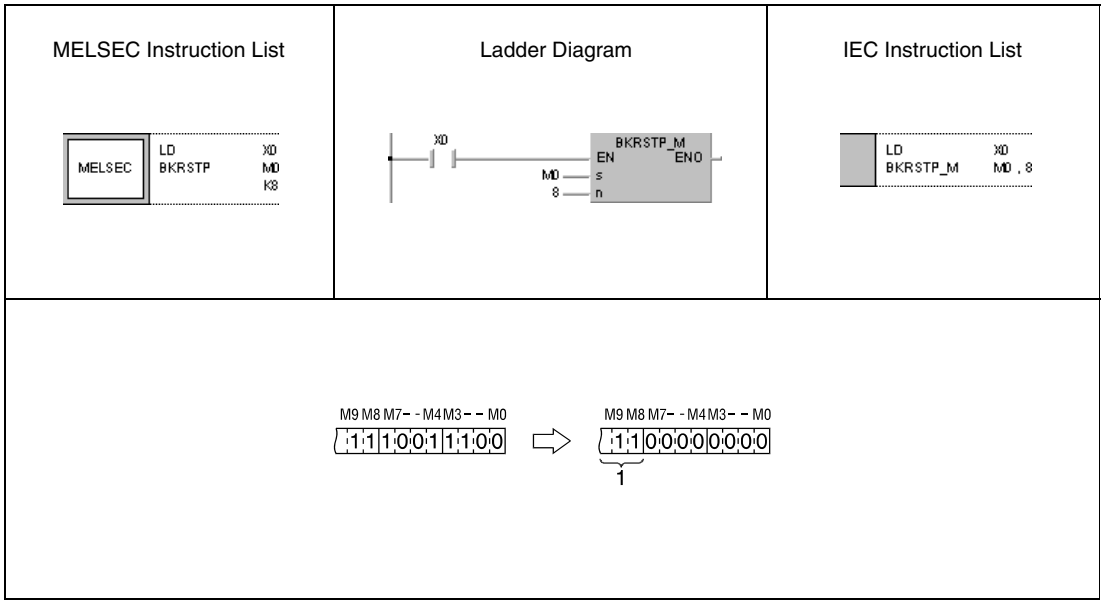
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value in n exceeds the number of bits of the devices designated by s. (Error code 4101)

Program Example 1

BKRSTP

With leading edge from X0, the following program resets the relays M0 through M7.



[1] These bits remain unchanged.

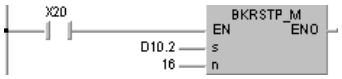
Program Example 2

BKRSTP

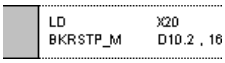
With leading edge from X20, the following program resets bits from the bit (b2) in D10 to the bit (b1) in D11.

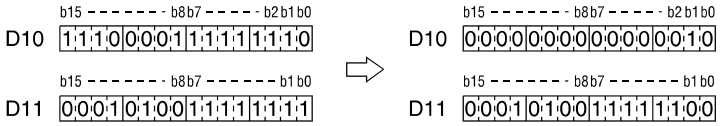
MELSEC Instruction List

Ladder Diagram



IEC Instruction List





7.5 Data processing instructions

Data processing instructions search data in specified devices, check the number of set bits, encode and decode data (e.g. for 7-segment displays), disunite and unite data, search maximum and minimum values, sort data, and calculate the totals of 16-/32-bit BIN data blocks.

The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Search 16-/32-bit data	SER	SER_M
	SERP	SERP_M
	DSER	DSER_M
	DSERP	DSERP_M
Check data bits (16-/32-bit)	SUM	SUM_M
	SUMP	SUMP_M
	DSUM	DSUM_M
	DSUMP	DSUMP_M
Encode/decode data	DECO	DECO_M
	DECOP	DECOP_M
	ENCO	ENCO_M
	ENCOP	ENCOP_M
7-segment decoding	SEG	SEG_M
	SEGP	SEGP_M
Disunite/unite 16-bit data words (4-bit units)	DIS	DIS_M
	DISP	DISP_M
	UNI	UNI_M
	UNIP	UNIP_M
Disunite/unite 16-bit data values (variable bit units)	NDIS	NDIS_M
	NDISP	NDISP_M
	NUNI	NUNI_M
	NUNIP	NUNIP_M
Disunite/unite 16-bit data values (byte units)	WTOB	WTOB_MD
		WTOB_K_MD
	WTOBP	WTOB_P_MD
		WTOB_K_P_MD
	BTOW	BTOW_MD
		BTOW_K_MD
	BTOWP	BTOW_P_MD
		BTOW_K_P_MD
Search maximum values in 16-/32-bit data	MAX	MAX_M
	MAXP	MAXP_M
	DMAX	DMAX_M
	DMAXP	DMAXP_M
Search minimum values in 16-/32-bit data	MIN	MIN_M
	MINP	MINP_M
	DMIN	DMIN_M
	DMINP	DMINP_M

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Sort 16-/32-bit data	SORT	SORT_M
	DSORT	DSORT_M
Calculate totals of 16-/32-bit BIN data blocks	WSUM	WSUM_M
	WSUMP	WSUMP_M
	DWSUM	DWSUM_M
	DWSUMP	DWSUMP_M
Calculation of averages	MEAN	
	MEANP	
	DMEAN	
	DMEANP	

7.5.1 SER, SERP, DSER, DSERP

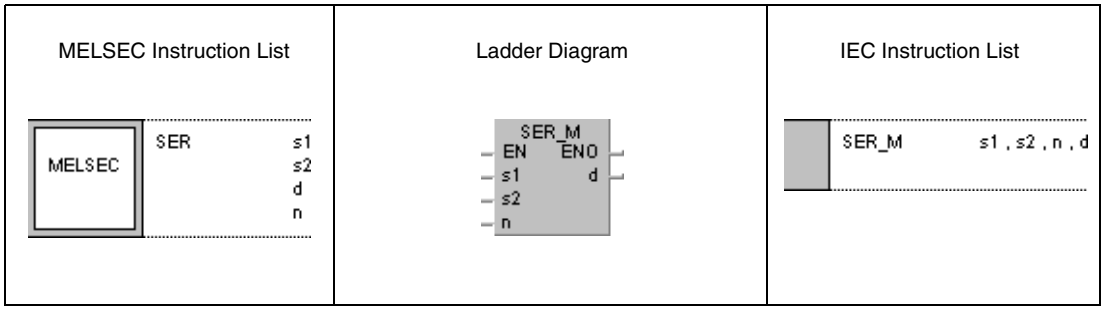
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

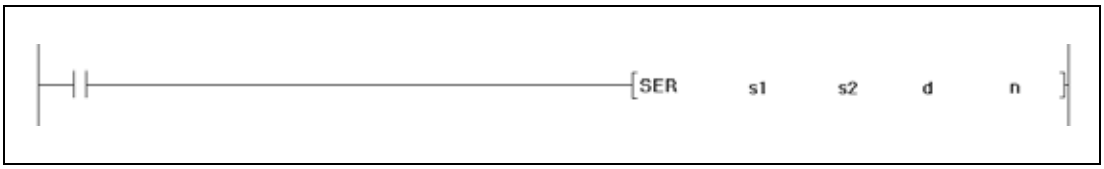
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	—	●	●	—	—	—	—	—	—
d	—	●	●	—	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s1	Data value to be searched, or first number of device storing this value	Word	ANY16
s2	Data to be searched through, or first number of device storing such data		ANY16/ANY32
d	First number of device storing result of search		Array [1..2] of ANY16/ANY32
n	Number of devices to be searched through		ANY16

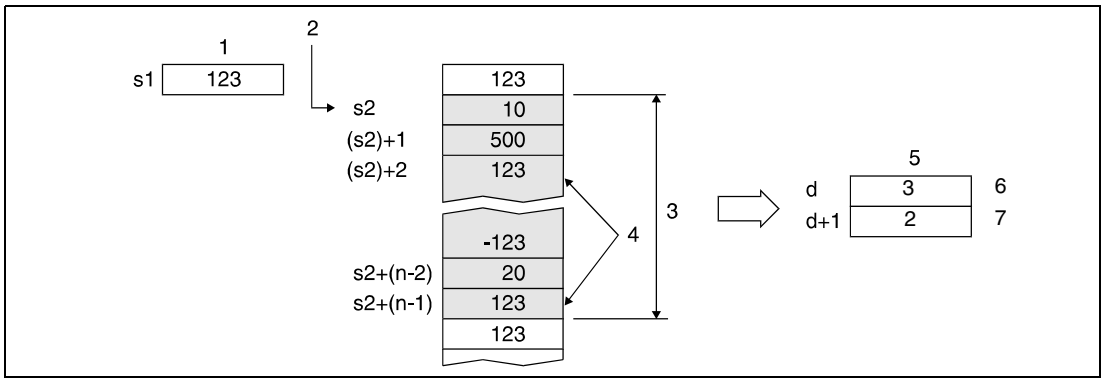
Functions **Search data**

SER / SERP Search 16-bit data

The SER instruction enables searching specified data in a specified search range. The search operation starts from the first number of device designated by s2. The entry code being searched for is specified by s1. The digit designation, i.e. the number of devices is specified by n.

A CPU stores the result of the search in d and d+1 as array_d[1..2] of ANY16.

After finishing the search operation the position of the first device storing the data value is stored in array_d[1] in d. Array_d[2] in d+1 stores the number of data values matching the entry code.



[1] Entry code
[2] Start of search
[3] Search range (n blocks)
[4] Matching data
[5] Search results
[6] Position of match
[7] Number of matches

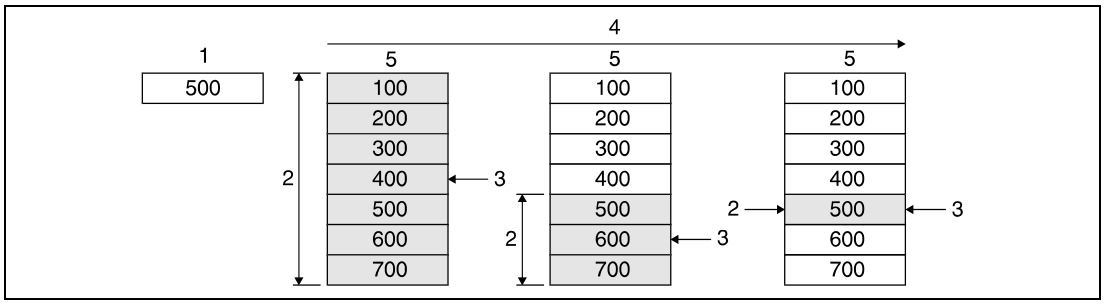
If the value in n is less than or equal to 0, the search operation will not be executed.

If no matching data is found, the content of d and d+1 (array_d[1] and array_d[2]) is 0.

NOTE *Provided the data to be searched through is stored in ascending order, the searching time can be shortened by setting the special relay SM702.*

SM702 ON:

The search range is halved and the size of the entry code determines in what half the code must be stored. This half is devided once again for another decision. This operation is proceeded until the matching value is found.



[1] Entry code
[2] Search range
[3] Comparison to entry code
[4] Processing sequence
[5] Search data

SM702 OFF:

The data search comparing the entry code to each data value starts from the beginning of the search range.

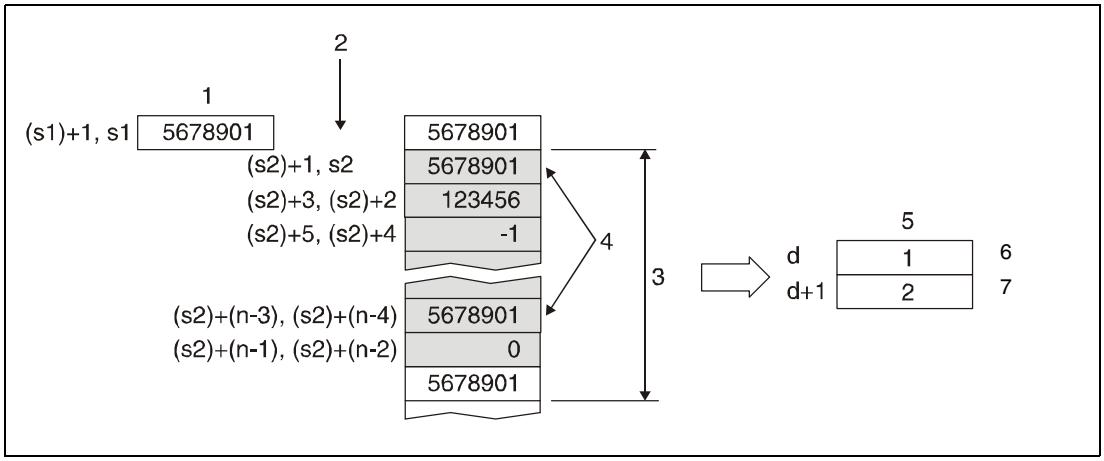
If the search range is not sorted in ascending order, there will be no accurate result with SM702 set.

DSER / DSERP Search 32-bit data

The DSER instruction enables searching specified data in a specified search range. The search operation starts from the first number of device designated by s2 (2 x n-devices). The entry code being searched for is specified by s1 and (s1)+1. The digit designation, i.e. the number of devices is specified by n.

The result of the search is stored in d and d+1 as array [1..2] of ANY16.

After finishing the search operation the position of the first device storing the data value is stored in the least significant array_d[1] (d). The most significant array_d[2] (d+1) stores the number of data values matching the entry code.



[1] Entry code
[2] Start of search
[3] Search range (2 x n)
[4] Matching data
[5] Search results
[6] Position of match
[7] Number of matches

If the value in n is less than or equal to 0, the search operation will not be executed.

If no matching data is found, the content of d and d+1 is 0.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

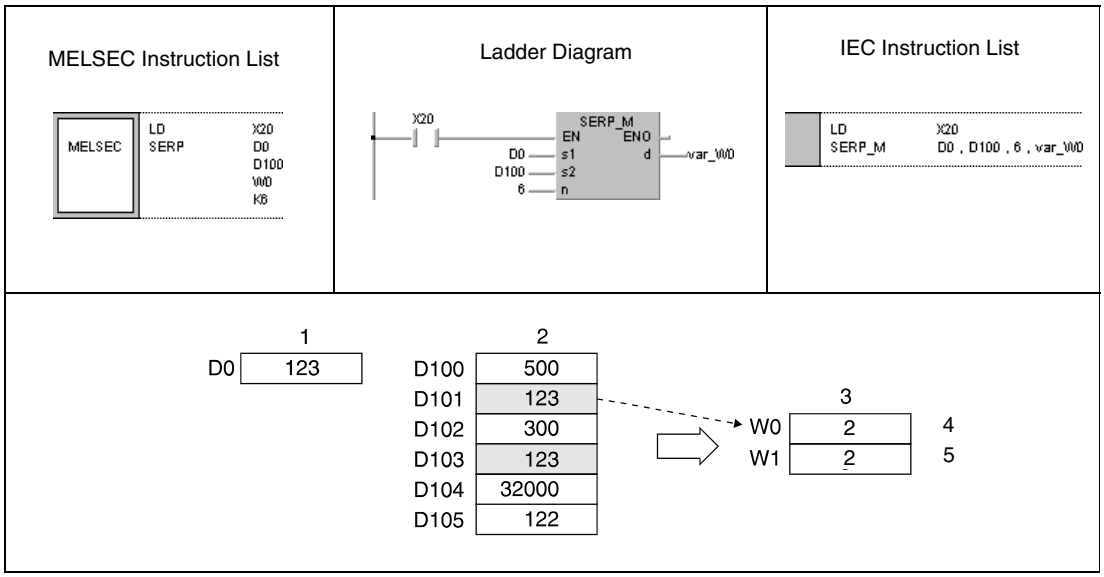
- The search range designated by n beginning from s2 exceeds the relevant device range. (Error code 4101)
- The device specified by d exceeds the range of the corresponding device. (For the Universal model QCPU, LCPU) (Error code 4101)

For details on index qualification refer to section 3.6.

Program Example 1

SERP

With leading edge from X20, the following program compares data in D100 through D105 to the data value in D0. The first matching position is stored in W0. The number of matches is stored in W1.

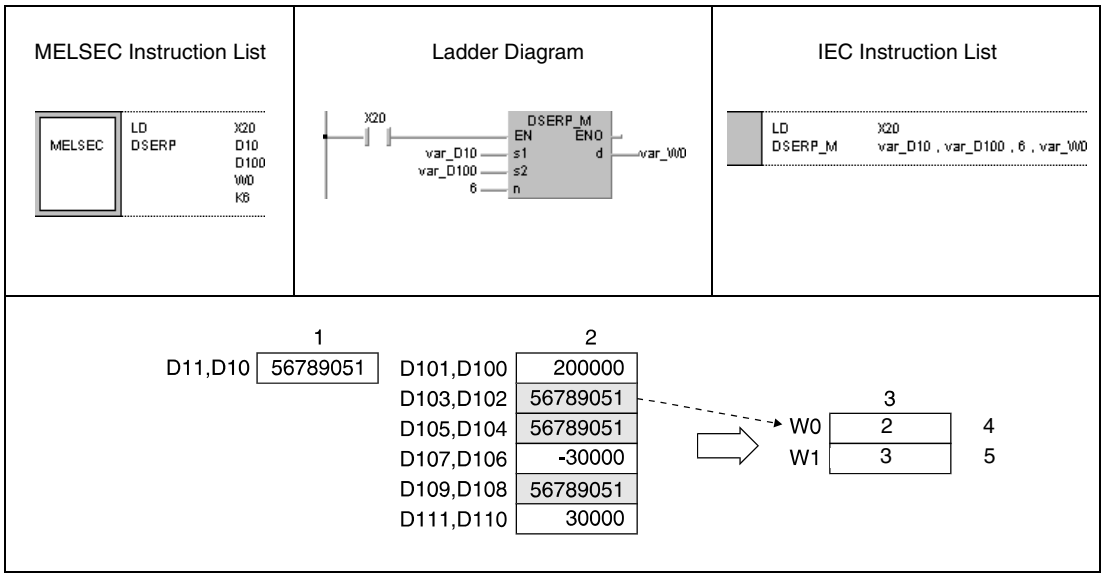


[1] Entry code
[2] Search range
[3] Search results
[4] Position of first match
[5] Number of matches

Program Example 2

DSERP

With leading edge from X20, the following program compares data in D100 through D111 to the data value in D11 and D10. The first matching position is stored in W0. The number of matches is stored in W1.



[1] Entry code
[2] Search range
[3] Search results
[4] Position of first match
[5] Number of matches

NOTE *These programs will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.5.2 SUM, SUMP, DSUM, DSUMP

CPU

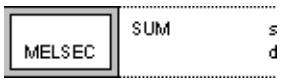
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

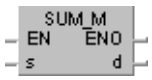
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer

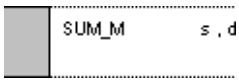
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

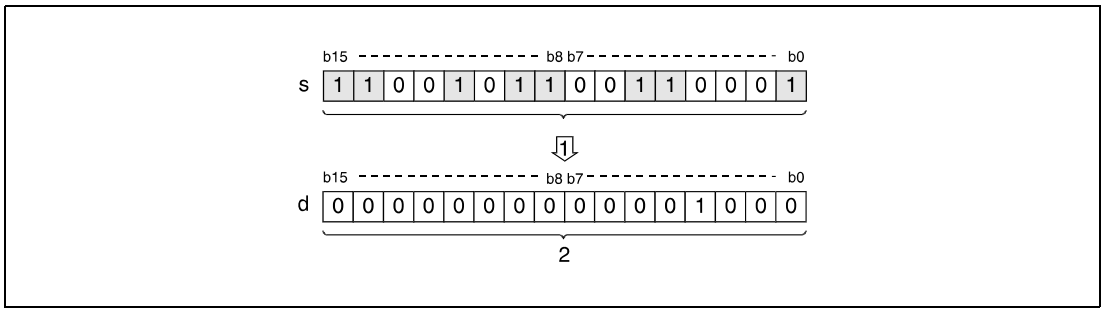
Set Data	Meaning	Data Type
s	First number of device storing data of which set bits are counted	BIN 16-/32-bit
d	First number of device storing number of set bits	

Functions

Check data bits

SUM 16-bit

The SUM instruction determines the number of bits set in a 16-bit data word. The device range to be checked is specified by s. The number of set bits is stored in d.

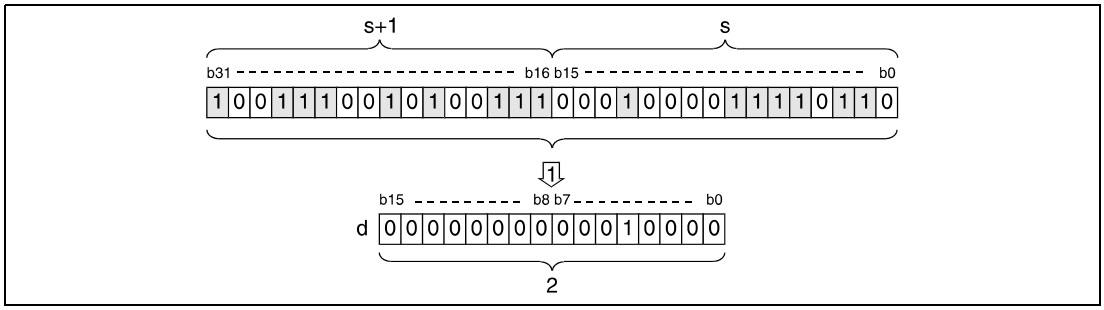


[1] Counting set bits

[2] Binary coded number of bits (In this example 8 bits are set.)

DSUM 32-bit

The DSUM instruction determines the number of bits set in a 32-bit data word. The device range to be checked is specified by s. The number of set bits is stored in d.



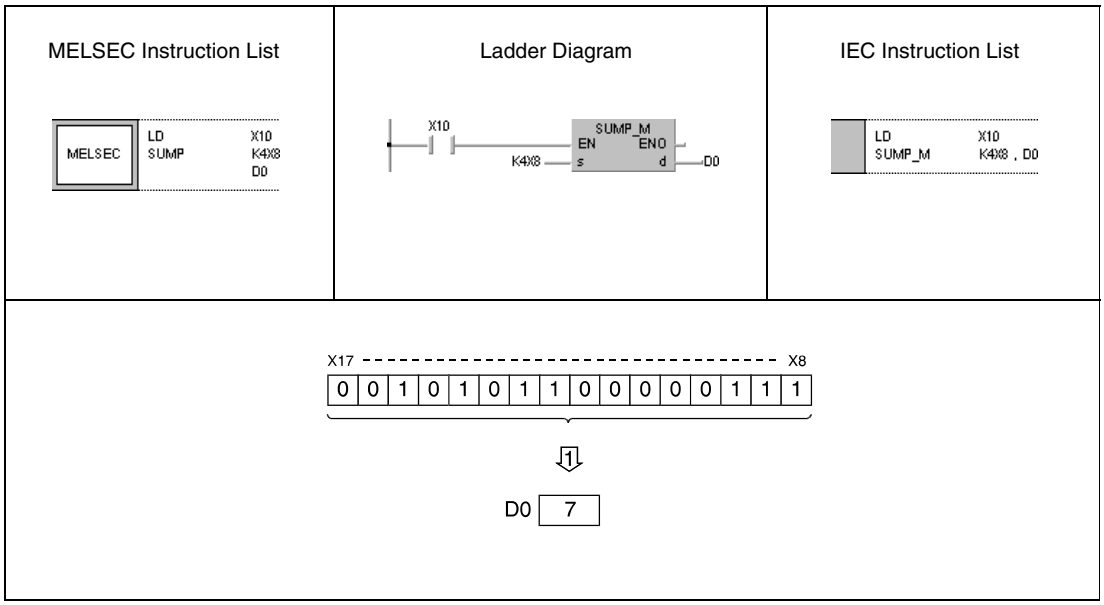
[1] Counting set bits

[2] Binary coded number of bits (In this example 16 bits are set.)

Program Example 1

SUMP

With leading edge from X10, the following program determines the number of set inputs within X8 through X10. The result is stored in D0.

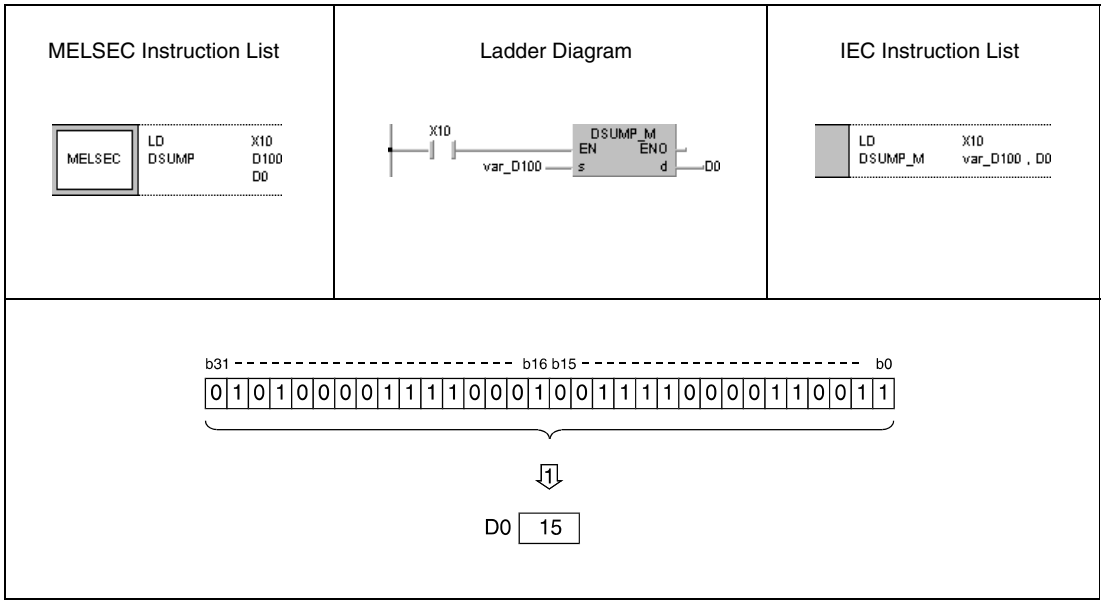


1 Storing the number of set bits in D0

Program Example 2

DSUMP

With leading edge from X10, the following program determines the number of set bits in D100 and D101. The result is stored in D0.



1 Storing the number of set bits in D0

NOTE *The program example 2 will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.5.3 DECO, DECOP

CPU

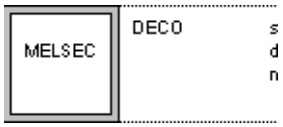
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

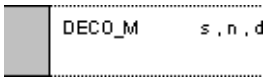
GX IEC Developer

MELSEC Instruction List

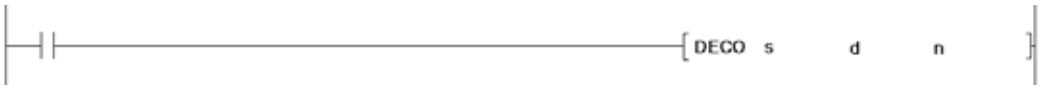


Ladder Diagram

IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	Data to be decoded or device storing such data	BIN 16-bit
d	First number of device storing decoded value	Address
n	Number of bits containing coded data	BIN 16-bit

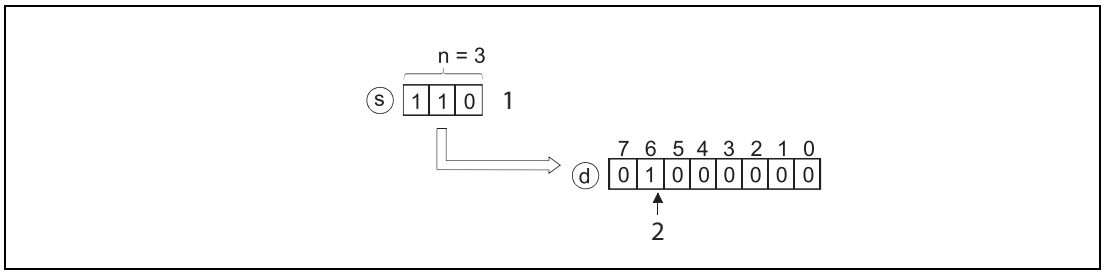
Functions

Decoding from 8 to 256 bits

DECO Decoding data

The DECO instruction decodes data in a device specified by s. The binary coded data is decoded as decimal number. This decimal number (≤ 256) indicates bit x (bx) to be set of a device specified by d.

The number of device addresses in s containing the coded data is specified by n.



[1] Binary value of s: 6

[2] Bit b6 in d is set.

The variable n must be set between 1 and 8.

If n = 0, the instruction is not executed and the specified device addresses remain unchanged.

A bit device is processed as single bit and a word device as 16-bit data value.

Operation Errors

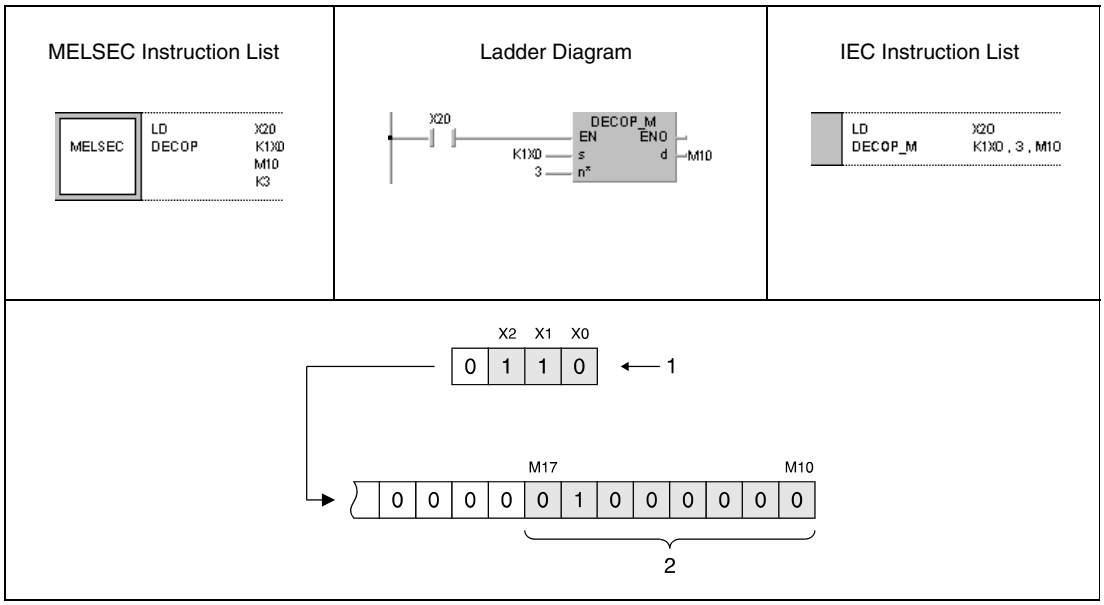
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The variable n is not set between 0 and 8. (Error code 4100)
- The bit x exceeds the relevant device range. (Error code 4101)

Program Example

DECOP

With leading edge from X20, the following program decodes data at X0 through X2. The result is stored in M10 through M17. The binary coded number 6 is contained in X0 through X2, so bit b6 (M16) in M10 through M17 is set.



[1] Binary coded value 6

[2] If the binary coded value is specified as 3 bits, 8 bits are occupied.

NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.5.4 ENCO, ENCOP

CPU

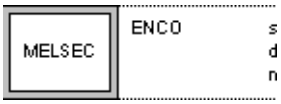
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

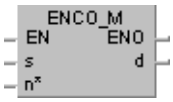
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	—	—	—	—	—	—
d	●	●	●	●	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

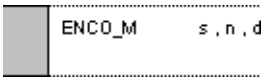
MELSEC Instruction List



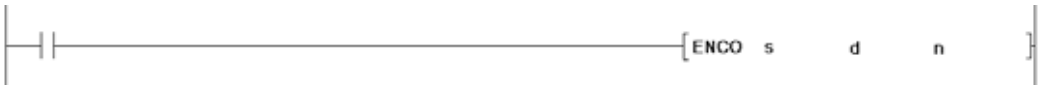
Ladder Diagram



IEC Instruction List



GX Works2



Variables

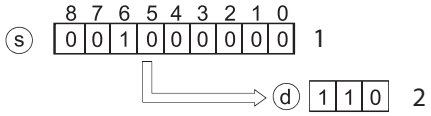
Set Data	Meaning	Data Type
s	Decoded data or device storing such data	BIN 16-bit
d	First number of device storing coded data	
n	Number of bits containing coded value	

Functions

Encoding from 256 to 8 bits

ENCO Encoding data

The ENCO instruction encodes data of a data record of up to 256 bits to a binary 8-bit data sequence. The initial number of device storing data to be encoded is specified by s. The bit x specified by s indicates the decimal value that will be stored binary encoded in d. The number of bits in d containing the encoded data is specified by n.



[1] Bit b6 In s is set.

[2] Binary value of d: 6

The variable n must be set between 1 and 8.

If n = 0, the instruction is not executed and the specified device addresses remain unchanged.

A bit device is processed as single bit and a word device as 16-bit data value.

If more than one bit is set processing starts with the highest bit.

Operation Errors

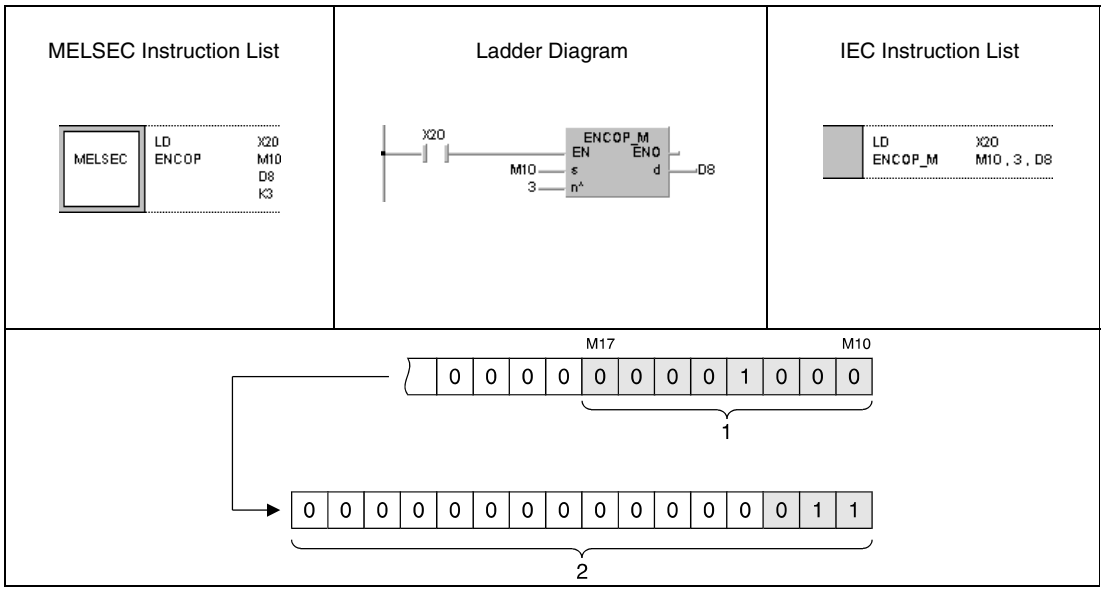
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The variable n is not set between 0 and 8. (Error code 4100)
- All data 2^n bits from s are "0". (Error code = 4100)
- The range 2^n bits from s exceeds the range of the relevant device. (Error code 4101)

Program Example

ENCOP

With leading edge from X20, the following program reads data in M10 through M17 and stores it binary encoded in D8.



[1] If the encoded value is stored in 3 bits in d, 8 bits are occupied in s.

[2] Binary encoded number 3 for set bit 3 (M13)

NOTE *This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.5.5 SEG, SEGP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

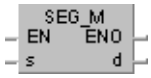
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

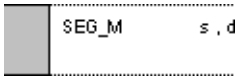
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
s	Data to be decoded, or first number of device storing such data	BIN 16-bit
d	First number of device storing 7-segment data	

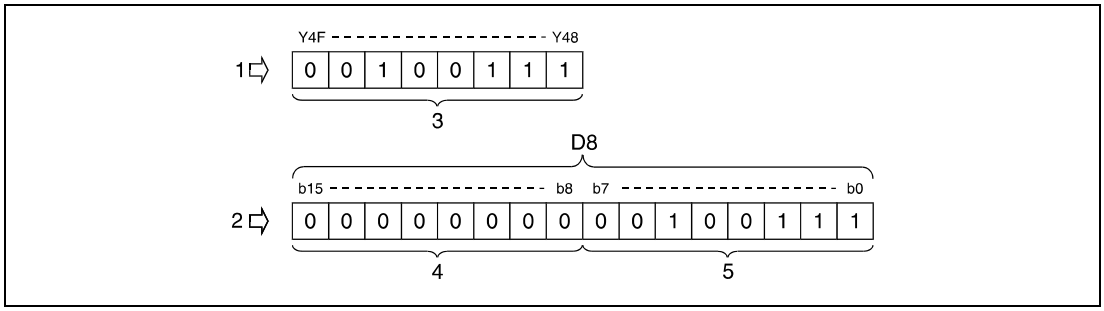
Functions **7-Segment decoding**

SEG / SEGP **Decoding a 4-digit binary value**

The SEG instruction converts a 4-digit binary value into 7-segment code in order to display the values 0 to F. The data value or the initial number of data to be encoded is specified by s. The 7-segment data is stored in d.

If the encoded 7-segment data are output to bit devices, the initial device number and the digit designation must always be specified in d. If a word device is specified by d, only the device number is required.

Storage of data in several bit devices or in a word device applies to the following scheme:



[1] Bit device

[2] Word device

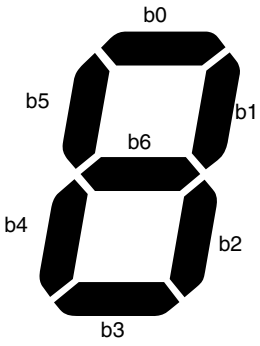
[3] 8 bits

[4] These bits are always reset to 0.

[5] 7-segment data

7-segment data

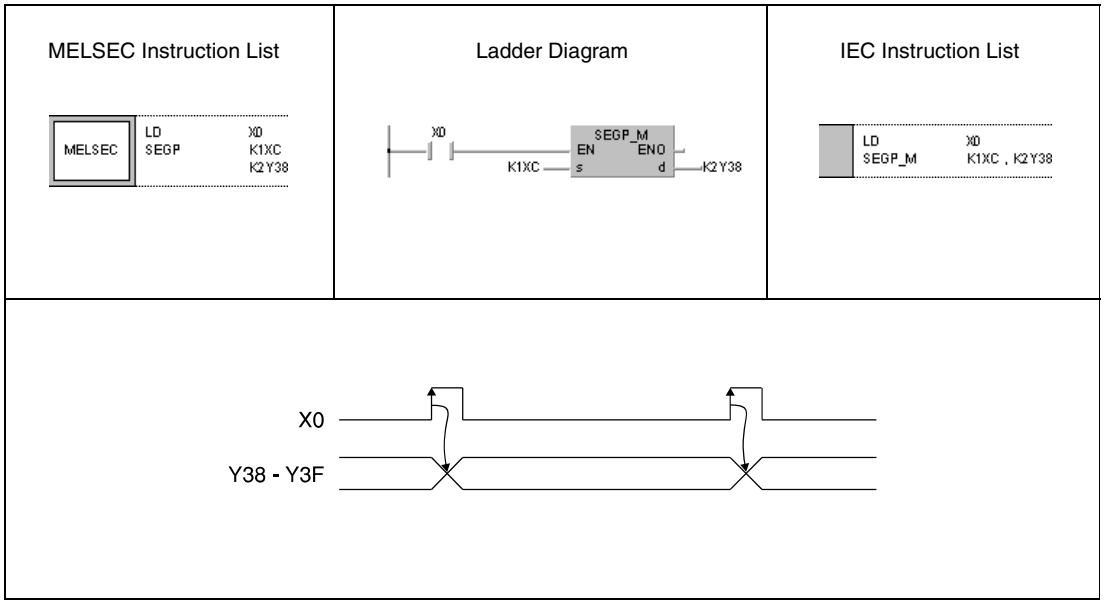
The following table contains an overview of 7-segment data in relation to the bit pattern of the source data. The first bit (b0) of 7-segment data either represents the status of the first bit device or the status of the least significant bit in a word device respectively.

s		Assignment of Segments	d								Display
HEX	Bit Pattern		B7	B6	B5	B4	B3	B2	B1	B0	
0	0000		0	0	1	1	1	1	1	1	0
1	0001		0	0	0	0	0	1	1	0	1
2	0010		0	1	0	1	1	0	1	1	2
3	0011		0	1	0	0	1	1	1	1	3
4	0100		0	1	1	0	0	1	1	0	4
5	0101		0	1	1	0	1	1	0	1	5
6	0110		0	1	1	1	1	1	0	1	6
7	0111		0	0	1	0	0	1	1	1	7
8	1000		0	1	1	1	1	1	1	1	8
9	1001		0	1	1	0	1	1	1	1	9
A	1010		0	1	1	1	0	1	1	1	A
B	1011		0	1	1	1	1	1	0	0	B
C	1100		0	0	1	1	1	0	0	1	C
D	1101		0	1	0	1	1	1	1	0	D
E	1110		0	1	1	1	1	0	0	1	E
F	1111		0	1	1	1	0	0	0	1	F

Program Example

SEGP

With leading edge from X0, the following program outputs the condition of inputs XC through XF as 7-segment code to the outputs Y38 through Y3F. The conditions of outputs Y38 through Y3F are maintained until they are overwritten with new data.



7.5.6 DIS, DISP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

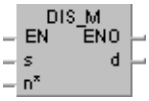
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

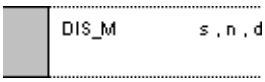
GX IEC Developer

MELSEC Instruction List

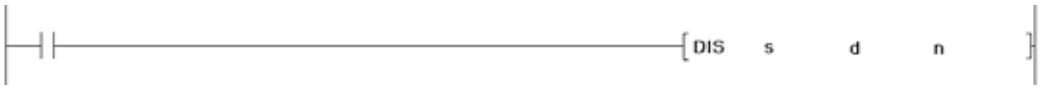
Ladder Diagram



IEC Instruction List



GX Works2



Variables

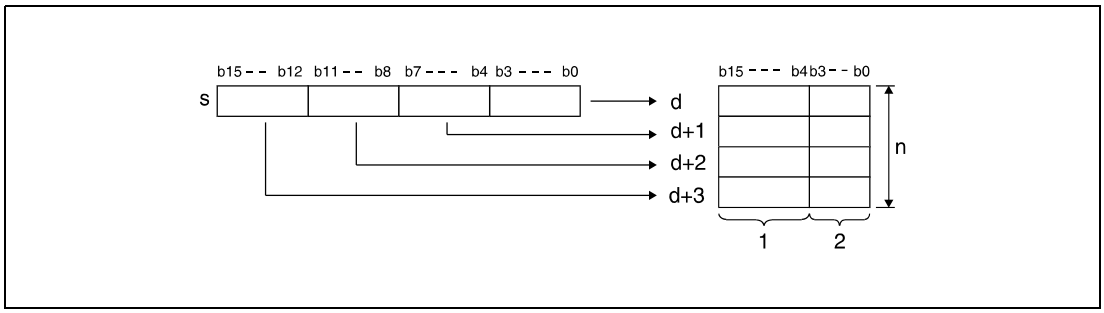
Set Data	Meaning	Data Type
s	First number of device storing data to be disunited	BIN 16-bit
d	First number of device storing disunited data	
n	Number of 4-bit groupings to be disunited. No processing for n = 0.	

Functions

Disuniting 16-bit data

DIS Disuniting 16-bit data values

The DIS instruction disunites a 16-bit data value to groupings of 4 bits and stores their conditions successively in up to 4 destination devices. For this instruction, the data value to be disunited in s, the number of 4-bit groupings in n, and the first number of destination device in d must be specified. Further 4-bit groupings are stored in d+n.



[1] These bits are reset to 0.

[2] Storage area

The upper 12 bits of the destination devices beginning from device number in d, are reset to 0.

The variable n can be set from 1 to 4 (corresponding 4 to 16 bits).

For n = 0 no operation is executed and the specified number of device remains unchanged.

Operation Errors

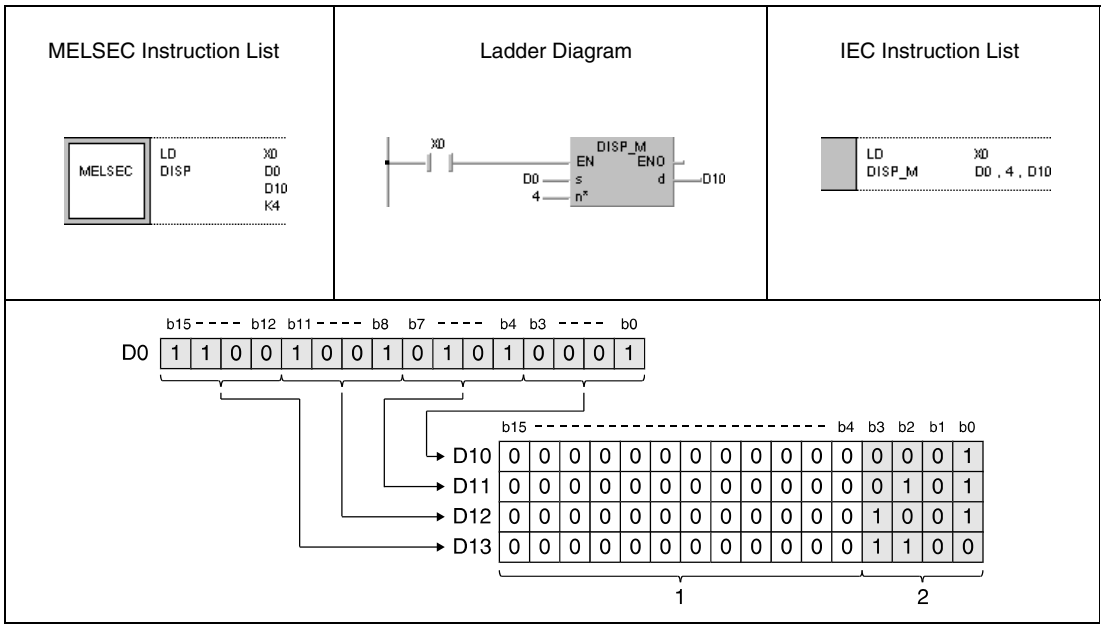
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value in n is not set between 0 and 4.
  (Error code 4100)
- The storage range d specified by n exceeds the relevant device range.
  (Error code 4101)

Program Example

DISP

With leading edge from X0, the following program disunites the 16-bit data value in D0 and stores the bit pattern in groupings of 4 bits in series in D10 through D13.



[1] These bits are reset to 0.

[2] Storage area

7.5.7 UNI, UNIP

CPU

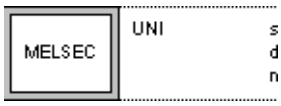
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●☆	●

Devices

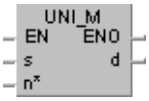
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	●	●	●	●	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List

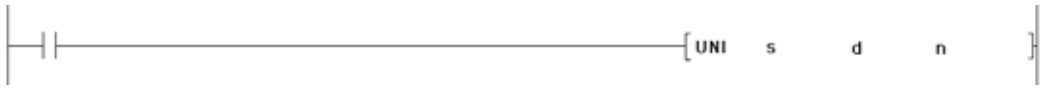


Ladder Diagram



IEC Instruction List

GX Works2



Variables

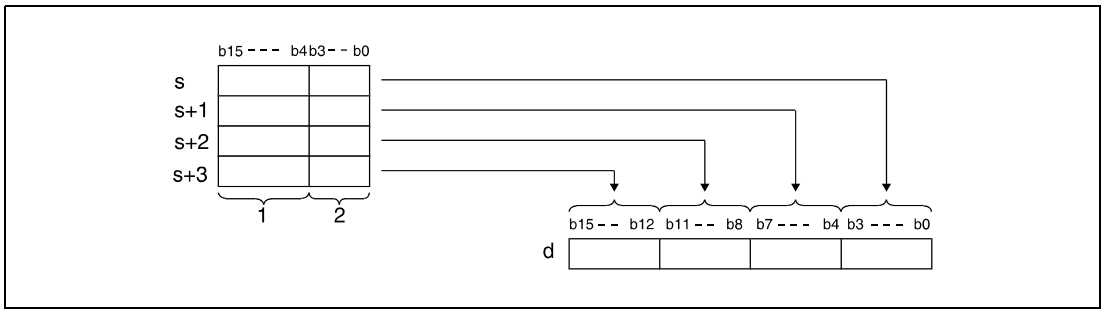
Set Data	Meaning	Data Type
s	First number of device storing data to be united	BIN 16-bit
d	First number of device storing united data	
n	Number of 4-bit groupings to be united. No processing for n = 0.	

Functions

Uniting 16-bit data

UNI Uniting 16-bit data values

The UNI instruction separates each 4 lowest bits of up to four 16-bit data values and unites their conditions in one 16-bit data value. For this instruction, the first number of device storing the data values in s to be united, the number of successive devices n, and the destination address in d must be specified.



[1] These bits are ignored.

[2] 4-bit groupings to be stored in d

The lower 4 bits of the source devices beginning from device number in d, are reset to 0.

The variable n can be set from 1 to 4.

For n = 0 no operation is executed and the specified number of device remains unchanged.

Operation Errors

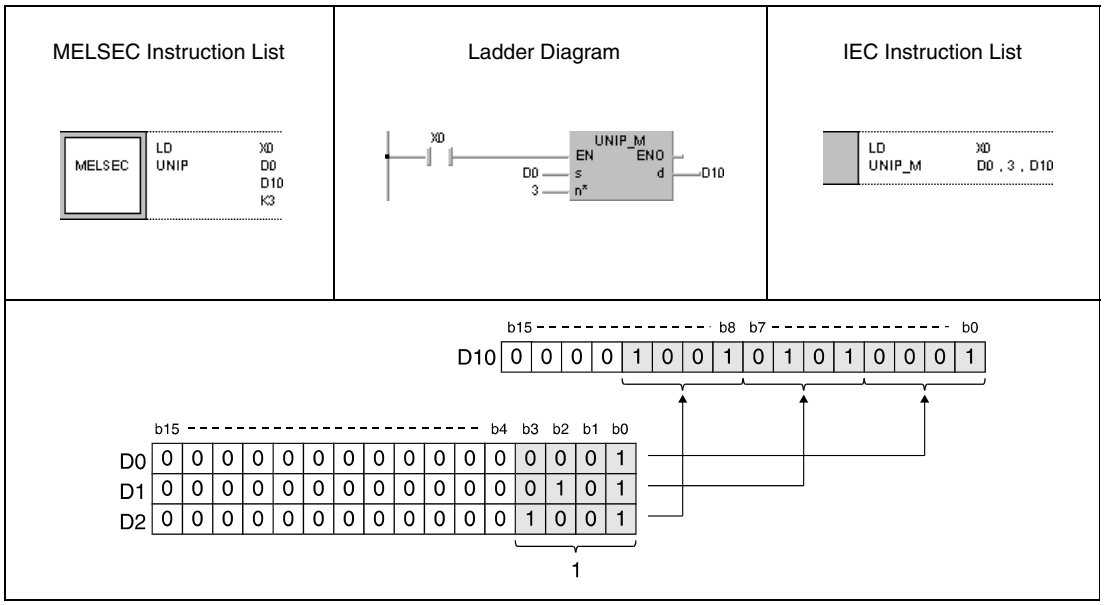
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value in n is not set within 0 and 4.
  (Error code 4100)
- The storage range s specified by n exceeds the relevant device range.
  (Error code 4101)

Program Example

UNIP

With leading edge from X0, the following program unites each lowest 4 bits (b0 through b3) of data registers D0 through D2 successively to one 16-bit data value (the highest 4 digits are "0") in D10.



[1] 4-bit groupings to be stored in D10

7.5.8 NDIS, NDISP, NUNI, NUNIP

CPU

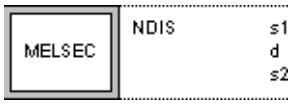
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

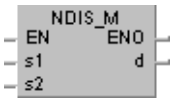
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—

GX IEC Developer

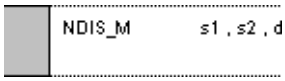
MELSEC Instruction List



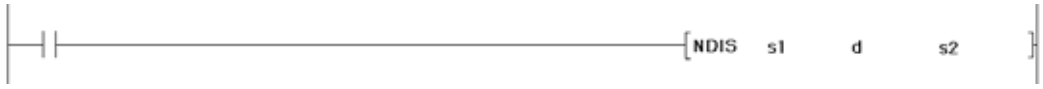
Ladder Diagram



IEC Instruction List



GX Works2



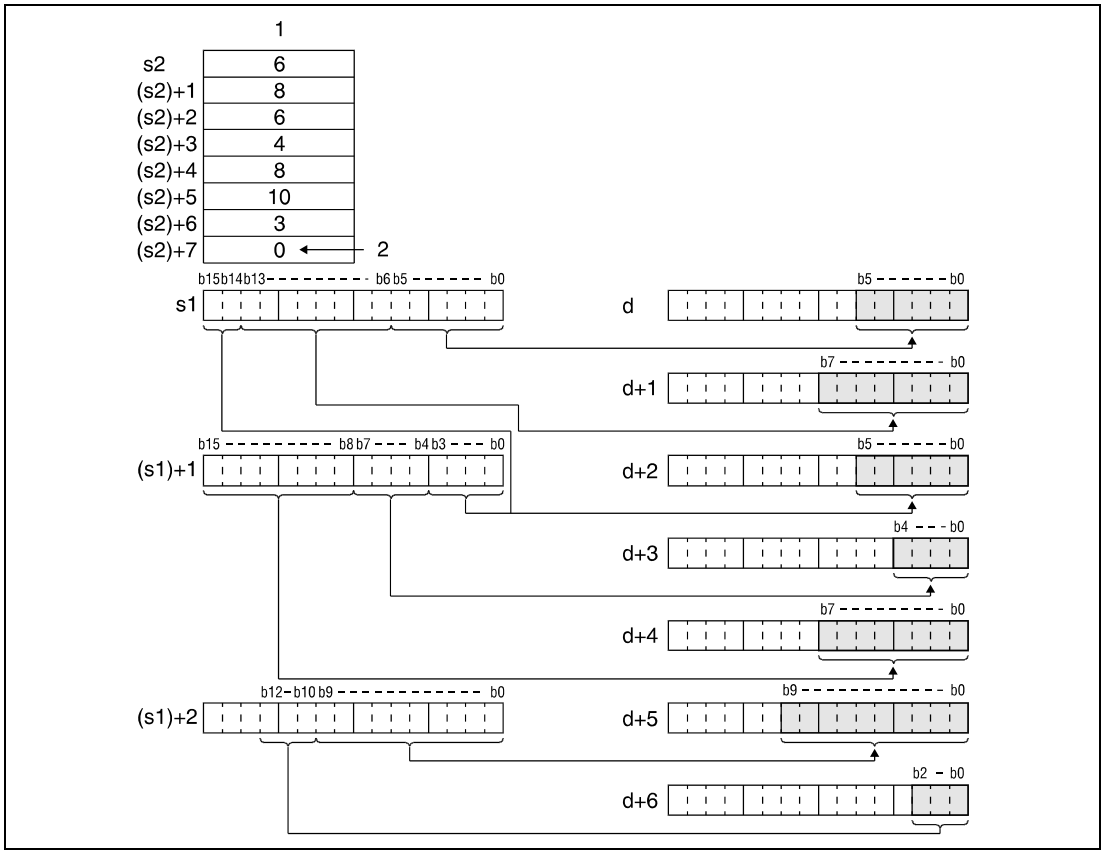
Variables

Set Data	Meaning	Data Type
s1	First number of device storing data to be disunited/united	BIN 16-bit
d	First number of device storing disunited/united data	
s2	Number of bits to be disunited/united in bit groupings	

Functions **Disuniting or uniting of data in random bit groupings**

NDIS Disuniting data

The NDIS instruction disunites data in devices specified from s1 on to bit groupings with a number of bits specified by s2. The disunited bit groupings are stored separately in the device specified by d onwards.



[1] Size of bit grouping

[2] The 0 indicates the end of processing.

The size of bit groupings specified by s2 can be set within 1 and 16 bits.

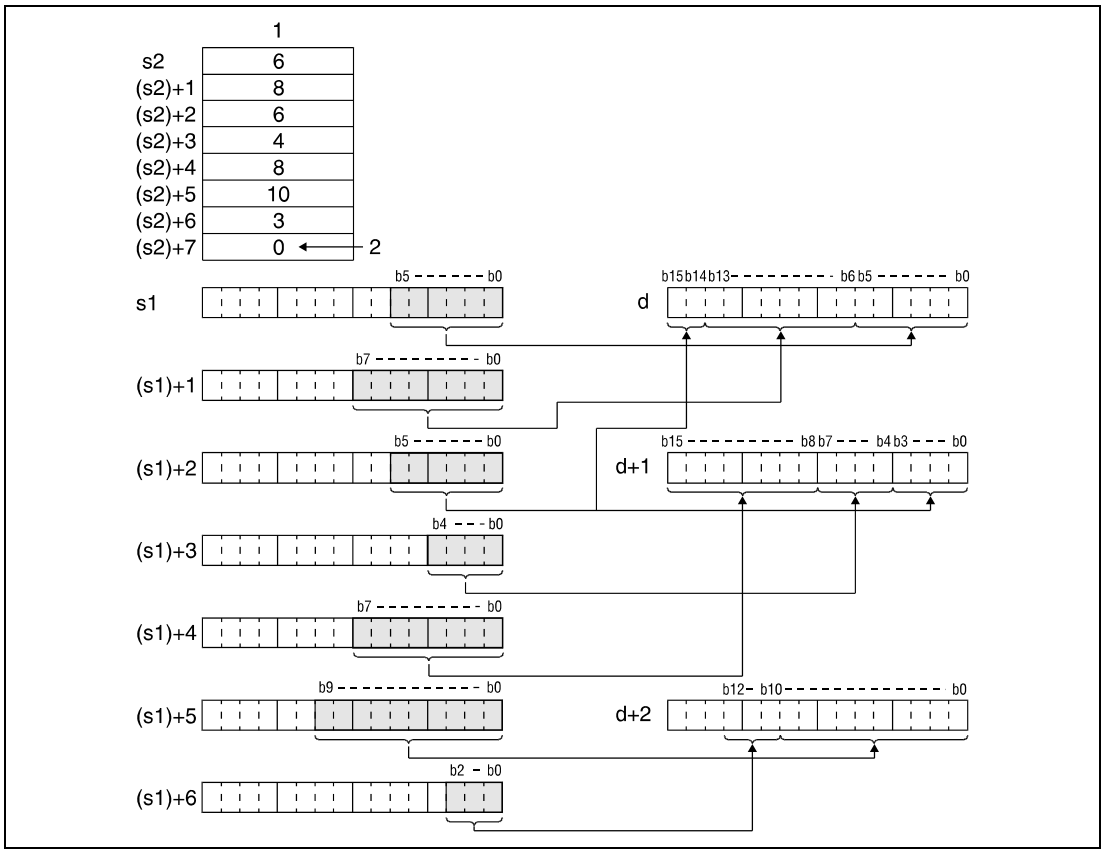
Values in s2 are processed from the first device address in s2 on and up to the address with the entry 0.

Do not overlap the device range for data to be dissociated (s1 to end range of s1) with the device range which stores the dissociated data (d to end range of d). If overlapped, the correct operation result may not be obtained.

Do not specify the same device number for s1, s2, and d. In this case the operation does not work correctly.

NUNI Uniting data

The NUNI instruction separates bit groupings of a size specified by s2 from devices specified by s1 and unites these bit groupings in one data value. The bit groupings are stored successively from the device specified by d on.



[1] Size of bit groupings

[2] The 0 indicates the end of processing.

The size of bit groupings specified by s2 can be set within 1 and 16 bits.

Values in s2 are processed from the first device address in s2 on and up to the address with the entry 0.

Do not overlap the device range for data to be linked (s1 to end range of s1) with the device range which stores the linked data (d to end range of d). If overlapped, the correct operation result may not be obtained.

Do not overlap the device numbers to be designated at s1, s2, and d. In this case the operation does not work correctly.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

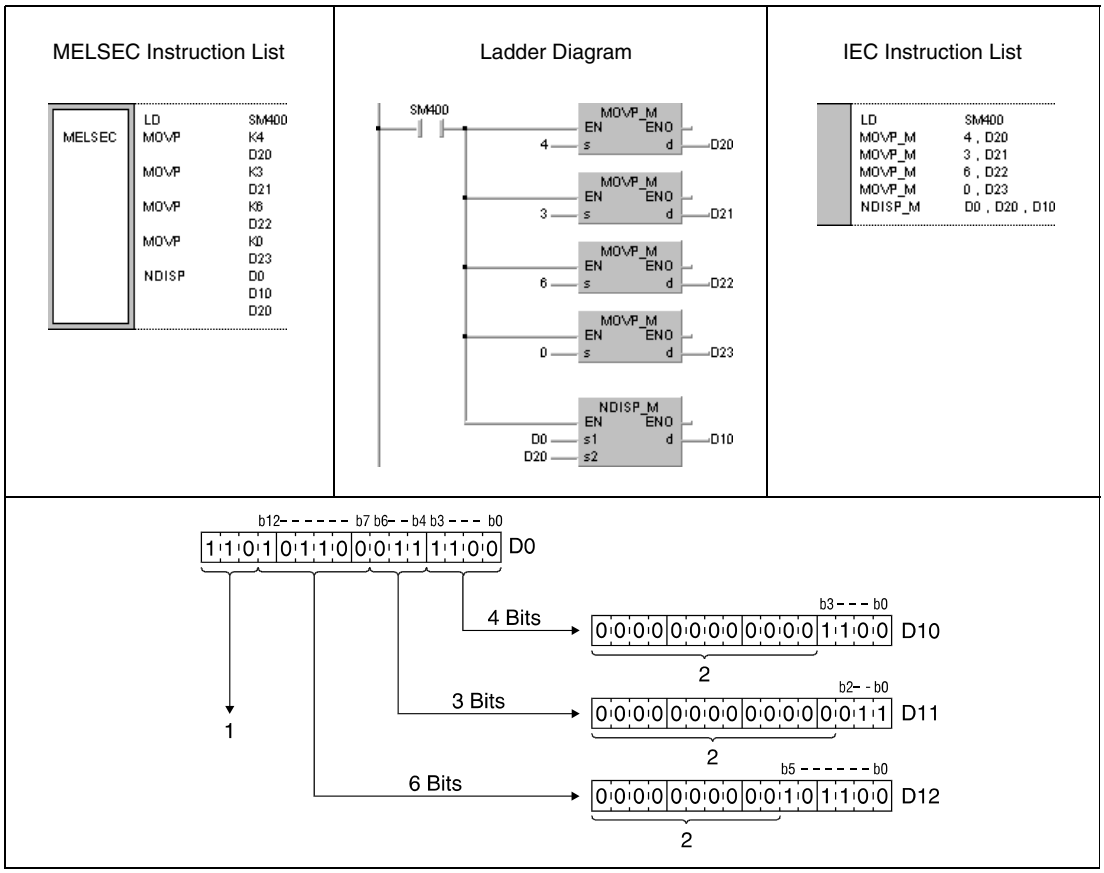
- The bit groupings of a size specified by s2 in the devices specified by s1 or d exceed the relevant storage device range.
  (Error code 4101)
- The size of bit groupings specified by s2 exceeds the valid range of 1 to 16 bits.
  (Error code 4100)

Program Example 1

NDISP

The following program separates the bit groupings b0 to b3 (4), b4 to b6 (3), and b7 to b12 (6) from D0 and stores each single bit grouping beginning from bit grouping b0 to b3 in D10 through D12.

The values in brackets indicate the size of bit groupings in D20 through D22. D23 must store the value 0 (see functions).



[1] These bits are ignored.

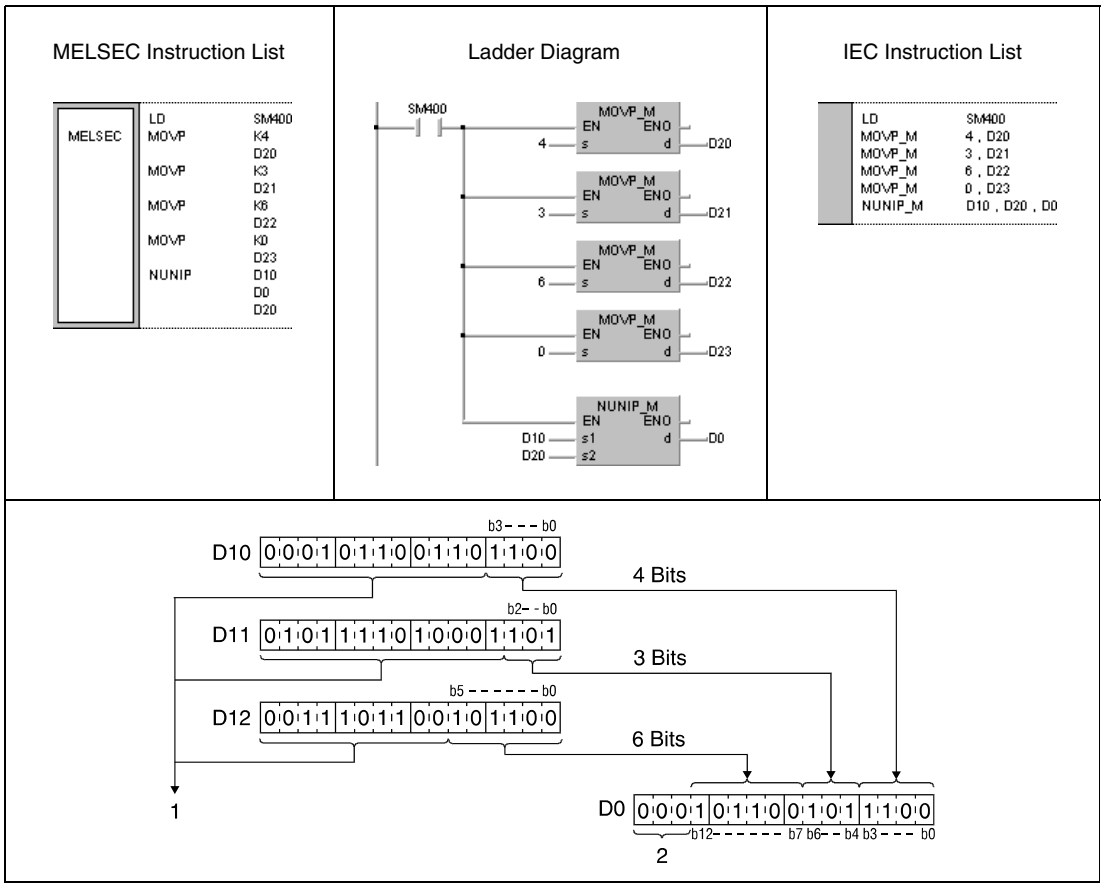
[2] These bits are reset to 0.

Program Example 2

NUNIP

The following program separates the bit groupings b0 to b3 (4), b0 to b2 (3), and b0 to b5 (6) from D10 through D12 and stores the bit groupings successively in D0 beginning from bit grouping b0 to b3.

The values in brackets indicate the size of bit groupings in D20 through D22. D23 must store the value 0 (see functions).



[1] These bits are ignored.

[2] These bits are reset to 0.

7.5.9 WTOB, WTOBP, BTOW, BTOWP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

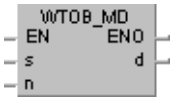
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

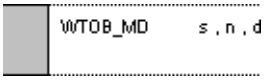
GX IEC Developer

MELSEC Instruction List

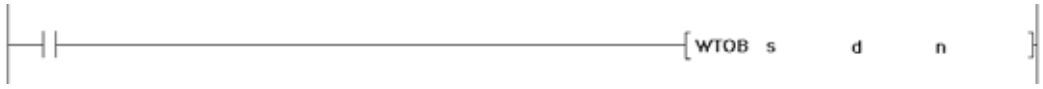
Ladder Diagram



IEC Instruction List



GX Works2



Variables

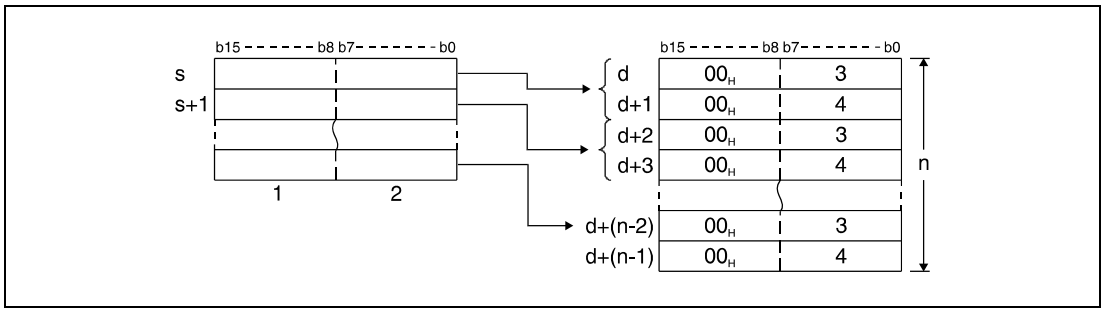
Set Data	Meaning	Data Type
s	First number of device storing data to be disunited/united in byte units	BIN 16-bit
d	First number of device storing disunited/inited bytes	
n	Number of byte units to be disunited/united	

Functions

Disuniting and uniting data in byte units

WTOB Disuniting data

The WTOB instruction disunites a 16-bit data value to byte units and stores their conditions successively in destination devices. For this instruction the data values in s to be disunited, the number of byte units in n, and the first number of destination device in d must be specified. Further byte units are stored in d+n. For storage only the lowest bytes of the devices specified by d are used.



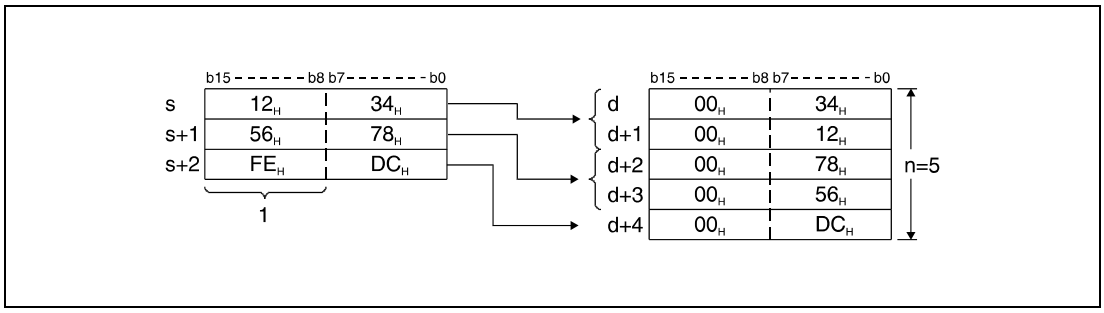
[1] Highest bytes

[2] Lowest bytes

[3] Data of the according lowest bytes

[4] Data of the according highest bytes

For example, if n = 5, 5 bytes are disunited from the device specified by s through s+2 and stored successively in the lowest bytes of the devices specified by d through d+4.

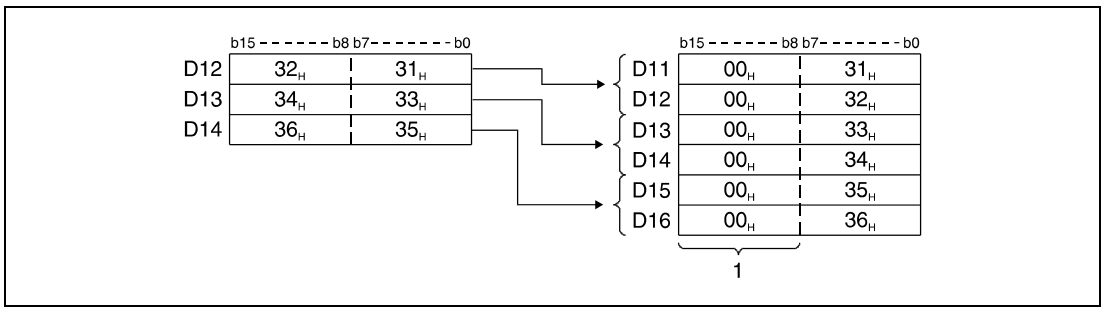


[1] These bytes are ignored.

The number of byte units specified by n automatically determines the range of 16-bit data in s and the storage range of the byte units in d.

If n = 0, the instruction is not executed and the specified device addresses remain unchanged.

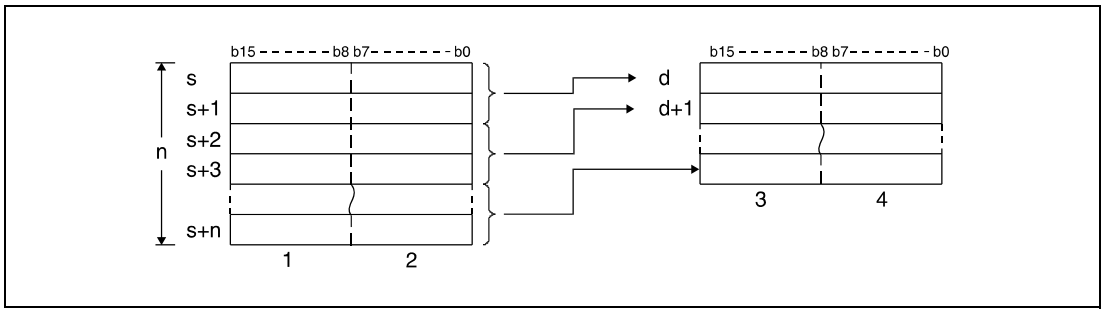
The highest bytes in the devices specified by d are set to the value "00H".



[1] These bytes are set to "00H".

BTOW Uniting data

The BTOW instruction separates any lowest bytes of 16-bit data values and stores their conditions in 16-bit data values. For this instruction, the initial number of data value in s to be united, the number of byte units n, and destination device in d must be specified.



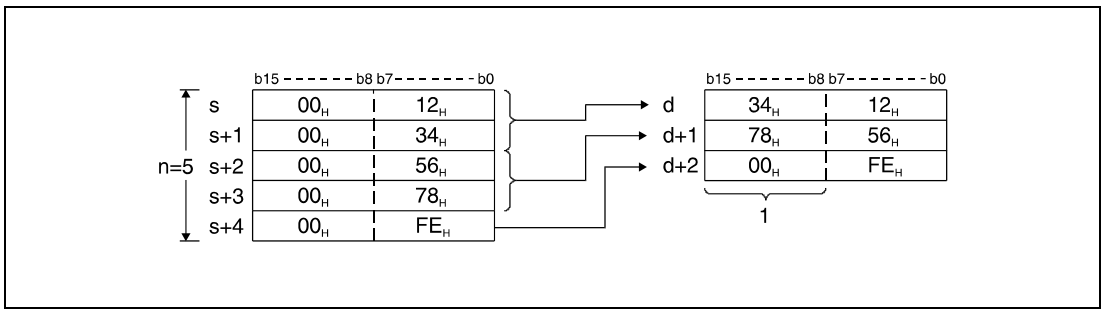
[1] These bytes are ignored.

[2] Data of 1st through nth byte

[3] Data of 2nd, 4th, and nth byte

[4] Data of 1st, 3rd, and (n-1)th byte

For example, if n = 5, the 5 lowest bytes are disunited from the device specified by s through s+4 and stored successively in the devices specified by d through d+2.



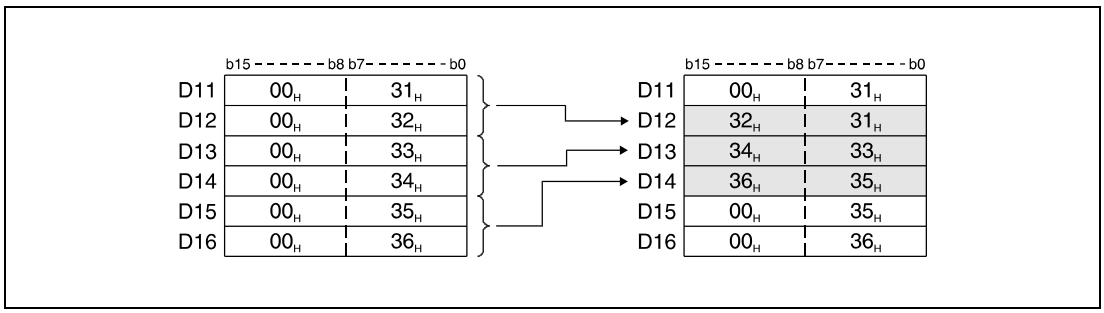
[1] This byte is set to "00H".

The number of byte units specified by n automatically determines the range of byte data in s and the storage range of the byte data in d.

If n = 0, the instruction is not executed and the specified device addresses remain unchanged.

The highest bytes in the devices specified by s are ignored on processing.

The operation is even processed correctly in cases where the storage ranges of s through s+n and d through d+n overlap. The following diagram shows a case where the lowest bytes are separated from D11 through D16 and stored again succcessively in D12 through D14.



Operation Errors

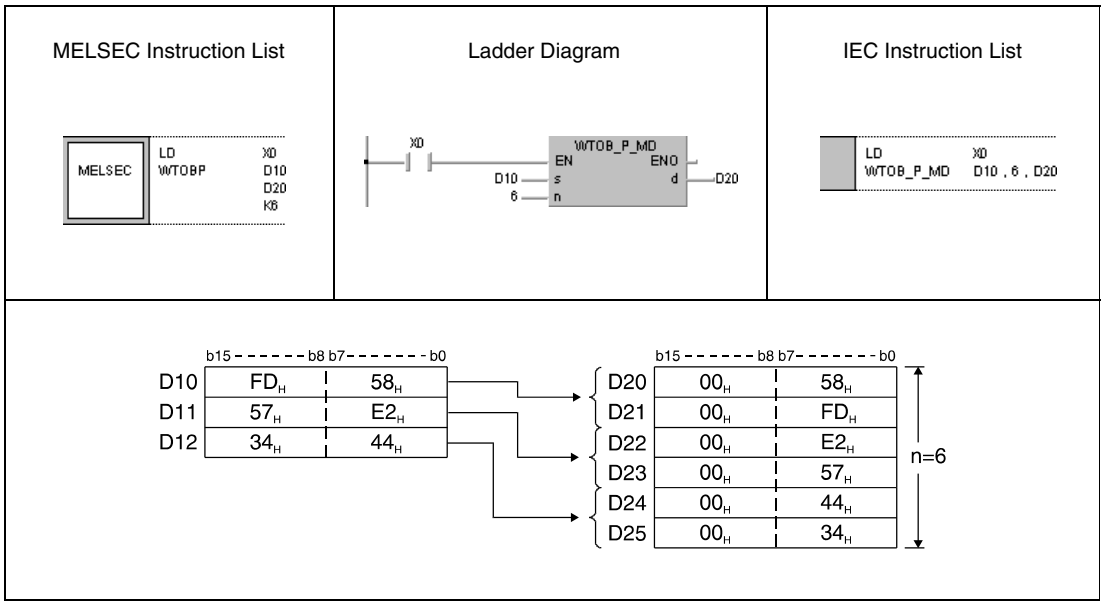
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of byte units specified by n, that are stored in the device specified by s, exceeds the relevant storage device range.
  (Error code 4101)
- The number of byte units specified by n, that are stored in the device specified by d, exceeds the relevant storage device range.
  (Error code 4101)

Program Example 1

WTOBP

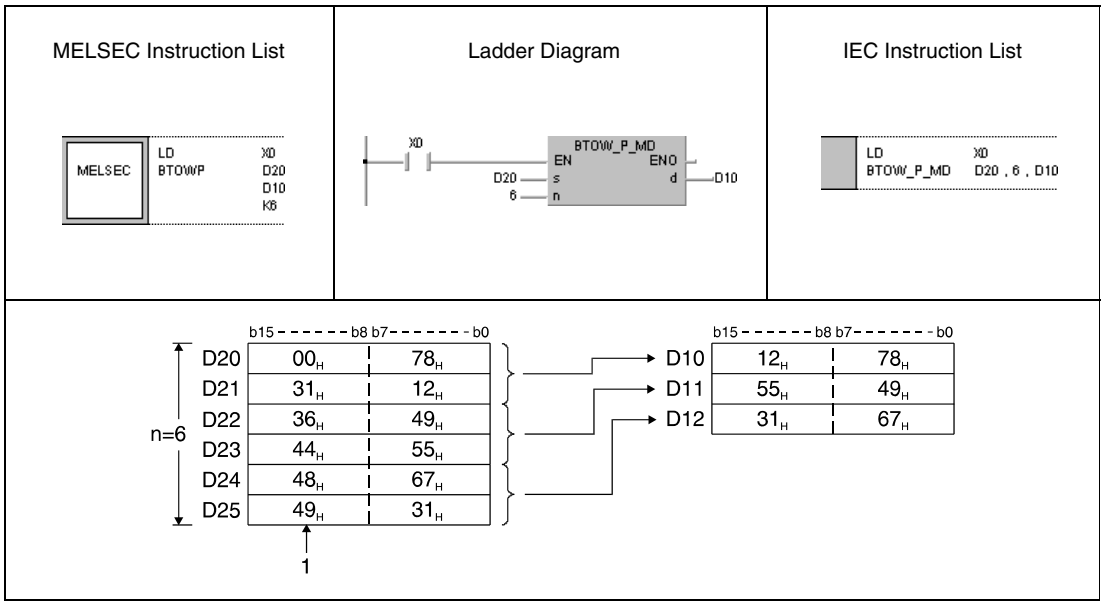
With leading edge from X0, the following program separates 6 bytes in D10 through D12 successively and stores these bytes in the lowest bytes in D20 through D25.



Program Example 2

BTOWP

With leading edge from X0, the following program separates the 6 lowest bytes in registers D20 through D25 and unites these bytes successively in D10 through D12.



[1] These bytes are ignored.

7.5.10 MAX, MAXP, DMAX, DMAXP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

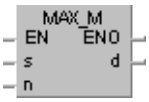
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

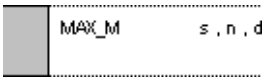
GX IEC Developer

MELSEC Instruction List

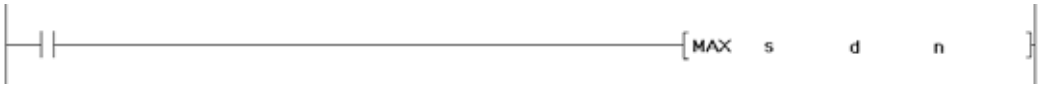
Ladder Diagram



IEC Instruction List



GX Works2



Variables

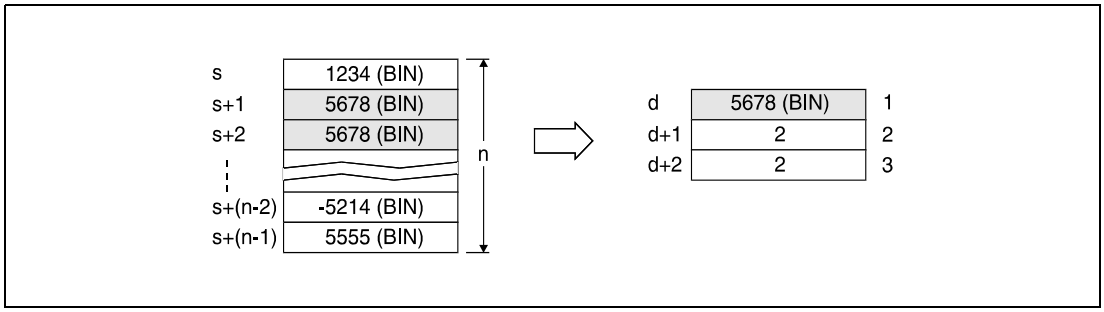
Set Data	Meaning	Data Type
s	First number of device storing data to be searched through for maximum values	BIN 16-/32-bit
d	First number of device storing search result	
n	Number of data blocks to be searched through	BIN 16-bit

Functions **Searching maximum values in 16-/32-bit data**

MAX Searching maximum values in 16-bit data

The MAX instruction searches for maximum values in 16-bit data blocks. The number of data blocks to be searched through is specified by n. The greatest value found in s through s+(n-1) is stored in d.

The first position in s through s+(n-1) where the maximum value is found is counted beginning from s = 1 and stored in d+1. The number of existing identical maximum values is stored in d+2.



[1] Found maximum value

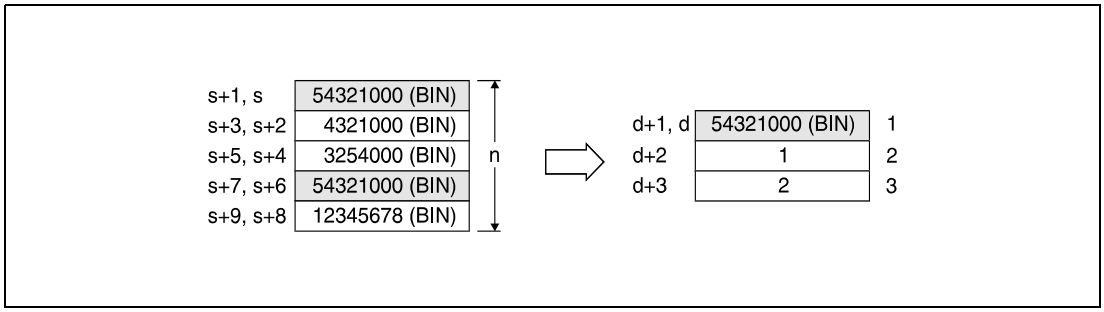
[2] First position the value has been found at

[3] Number of identical maximum values

DMAX Searching maximum values in 32-bit data

The DMAX instruction searches for maximum values in 32-bit data blocks. The number of data blocks to be searched through is specified by n. The greatest value found in s through s+(n-1) is stored in d.

The first position in s through s+(n-1) where the maximum value is found is counted beginning from s = 1 and stored in d+2. The number of existing identical maximum values is stored in d+3.



[1] Found maximum value

[2] First position the value has been found at

[3] Number of identical maximum values

Operation Errors In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

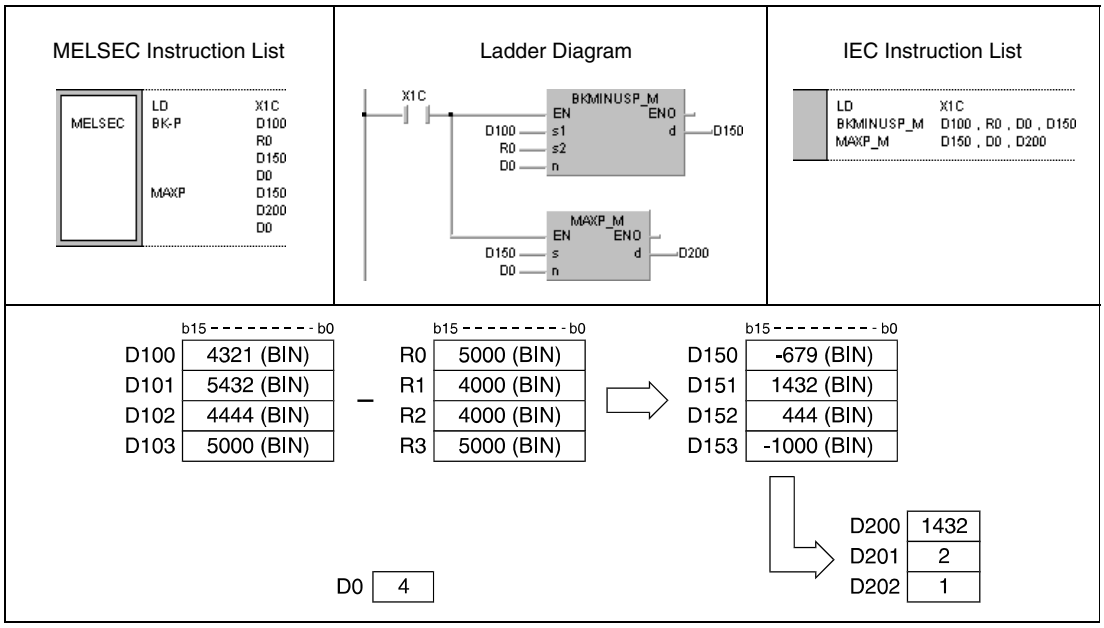
- The number of data blocks specified by n stored in the devices specified by s exceeds the relevant storage device range.
  (Error code 4101)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example 1

MAXP

With leading edge from X1C, the following program subtracts data in R0 through R3 from data in D100 through D103 and stores the result in D150 through D153. The number of 16-bit data blocks (4) is specified in D0.

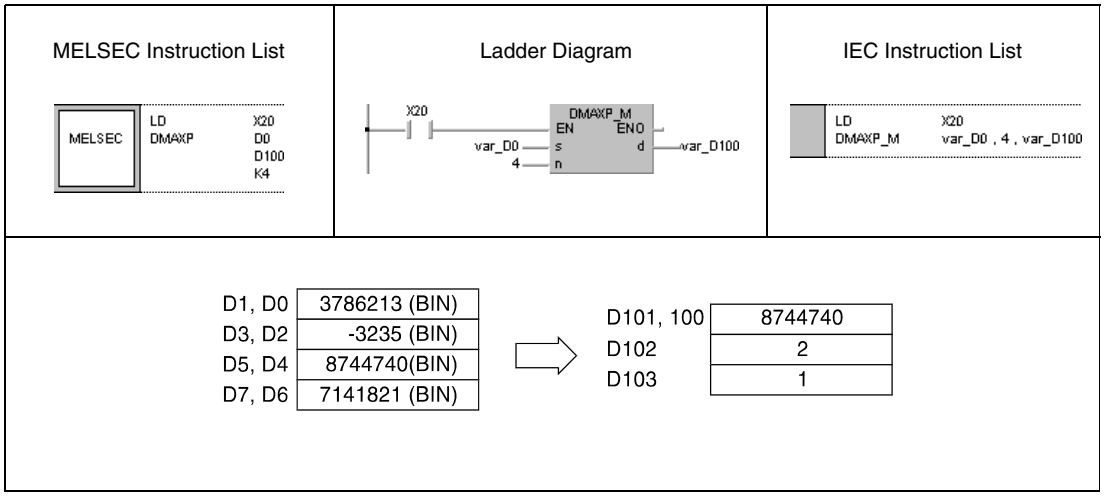
In the following step, as well with leading edge from X1C, the registers D150 through D153 are searched through for the maximum value. The value found is stored in D200, its position is stored in D201, and the number of identical maximum values is stored in D202.



Program Example 2

DMAXP

With leading edge from X20, the following program searches for the maximum value of 32-bit data in D100 and D101. The position of the value is stored in D102, the number of identical maximum values is stored in D103.



NOTE

The program example 2 will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.5.11 MIN, MINP, DMIN, DMINP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

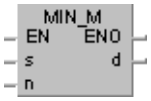
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

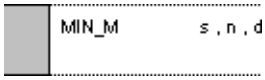
GX IEC Developer

MELSEC Instruction List

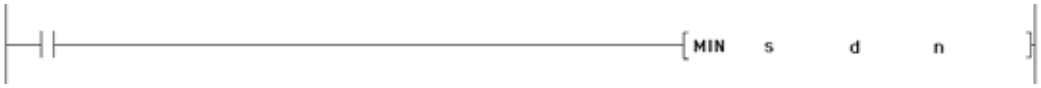
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	First number of device storing data to be searched through for minimum values	BIN 16-/32-bit
d	First number of device storing search result	
n	Number of data blocks to be searched through	BIN 16-bit

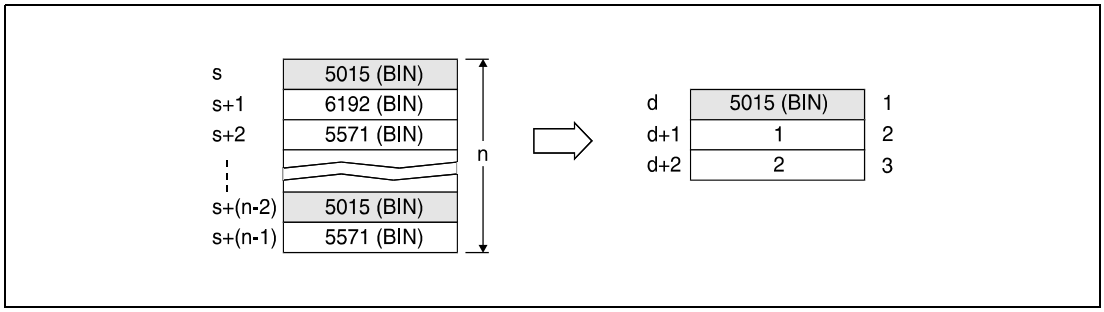
Functions

Searching minimum values in 16-/32-bit data

MIN Searching minimum values in 16-bit data

The MIN instruction searches for minimum values in 16-bit data blocks. The number of data blocks to be searched through is specified by n. The smallest value found in s through s+(n-1) is stored in d.

The first position in s through s+(n-1) where the minimum value is found is counted beginning from s = 1 and stored in d+1. The number of existing identical minimum values is stored in d+2.



[1] Found minimum value

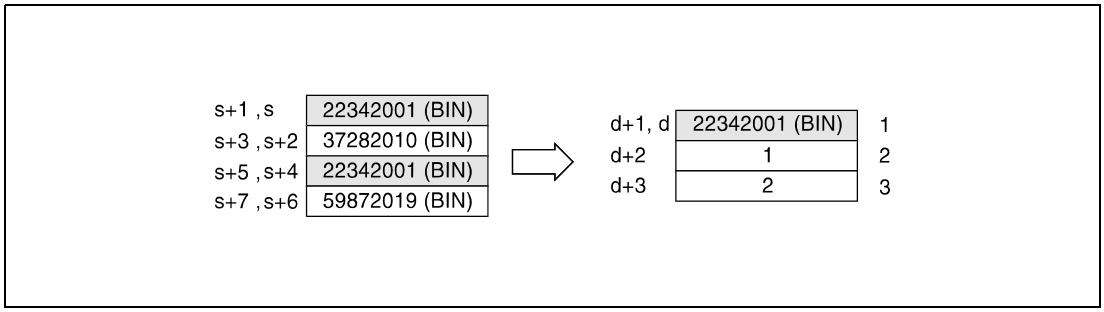
[2] First position the value has been found at

[3] Number of identical minimum values

DMIN Searching minimum values in 32-bit data

The DMIN instruction searches for minimum values in 32-bit data blocks. The number of data blocks to be searched through is specified by n. The smallest value found in s through s+(n-1) is stored in d and d+1.

The first position in s through s+(n-1) where the minimum value is found is stored in d+2. The number of existing identical minimum values is stored in d+3.



[1] Found minimum value

[2] First position the value has been found at

[3] Number of identical minimum values

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

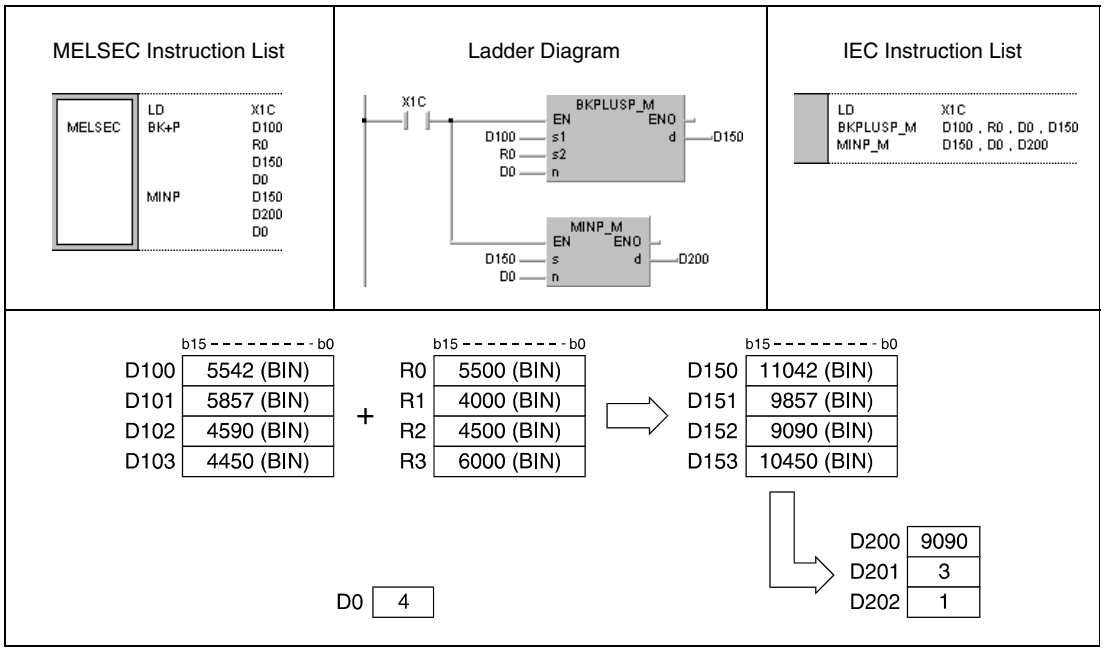
- The number of data blocks specified by n stored in the devices specified by s exceeds the relevant storage device range.
  (Error code 4101)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example 1

MINP

With leading edge from X1C, the following program adds data in D100 through D103 to data in R0 through R3 and stores the result in D150 through D153. The number of 16-bit data blocks (4) is specified in D0.

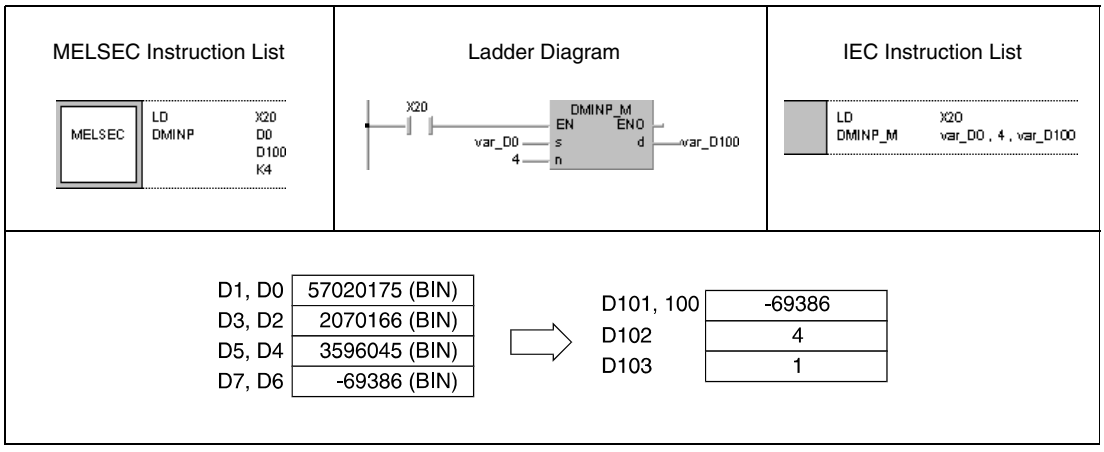
In the following step, as well with leading edge from X1C, the registers D150 through D153 is searched through for the minimum value. The value found is stored in D200, its position is stored in D201, and the number of identical minimum values is stored in D202.



Program Example 2

DMINP

With leading edge from X20, the following program searches for the minimum value of 32-bit data in D0 through D7 and stores the value in D100 and D101. The position of the value is stored in D102, the number of identical minimum values is stored in D103.



NOTE *The program example 2 will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.5.12 SORT, DSORT

CPU

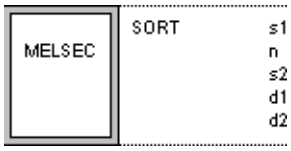
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

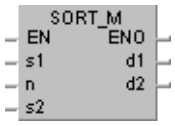
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
d1	●	—	—	—	—	—	—	—	—
d2	—	●	●	—	—	—	—	—	—

GX IEC Developer

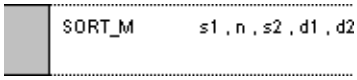
MELSEC Instruction List



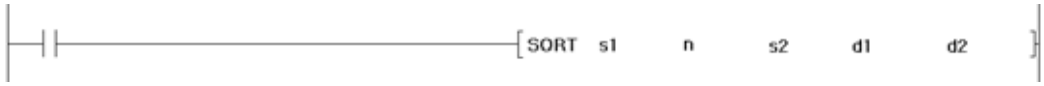
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s1	First number of device storing data to be sorted	BIN 16-/32-bit
n	Number of data blocks to be sorted	BIN 16-bit
s2	Number of data blocks to be compared each sort operation	BIN 16-bit
d1	Number of bit to be set after finishing sort operation	Bit
d2	For system use only	BIN 16-bit

Functions **Sorting 16-/32-bit data**

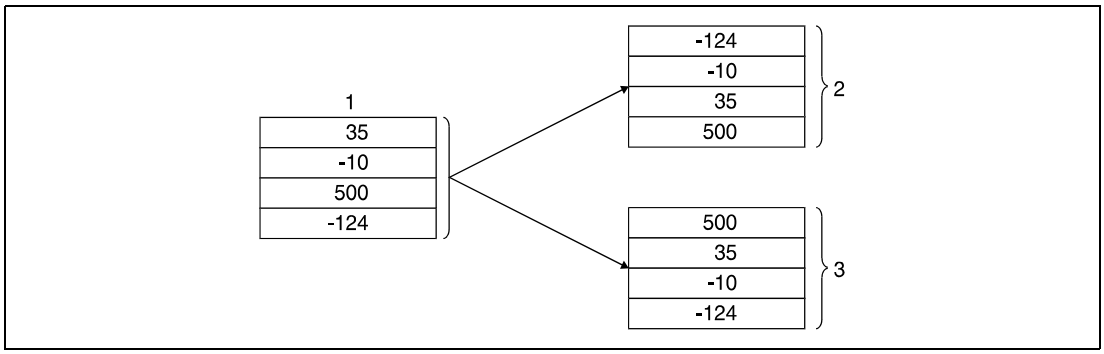
SORT Sorting 16-bit data

The SORT instruction sorts 16-bit data specified by s1 in ascending or descending order. The number of data to be sorted is specified by n.

The sorting order is set via the special relay SM703:

– SM703 OFF: Ascending order

– SM703 ON: Descending order



[1] Data to be sorted

[2] Data sorted in ascending order (SM703 = OFF)

[3] Data sorted in descending order (SM703 = ON)

For finishing the SORT instruction several scans are required. The number of required scans can be calculated by the division of the maximum number of scans by the number of 16-bit data specified in s2, to be compared each scan (decimal fractions are rounded up). Increasing the number of 16-bit data specified in s2 reduces the number of required scans for sorting but increases the processing time per scan.

The required number of sorting scans until finishing the sort operation is calculated via the following equation:

Required number of sorting scans = ((n) x (n-1)) / (2 x (s2))

For example, for n = 10 and s2 = 1 the result is 45 sort scans until finishing the sort operation.

For n = 10 and s2 = 2 the result is 22.5. Rounded up, 23 sort scans are required.

The bit specified in d1 is reset during the sort operation and will be set again when the sort operation is finished. This bit remains set and must be reset by appropriate programming.

The devices specified in d2 and (d2)+1 are used for internal system processing during the sort operation. So, these devices must not be changed by programming.

If the value in n is changed during the operation, the operation is processed with the currently set number of 16-bit data.

By resetting the execution condition, the operation will be terminated. Upon setting the execution condition again, the sort operation will be restarted.

To execute another sort operation immediately after the completion of the previous sort, turn OFF the execution command once, then turn it ON.

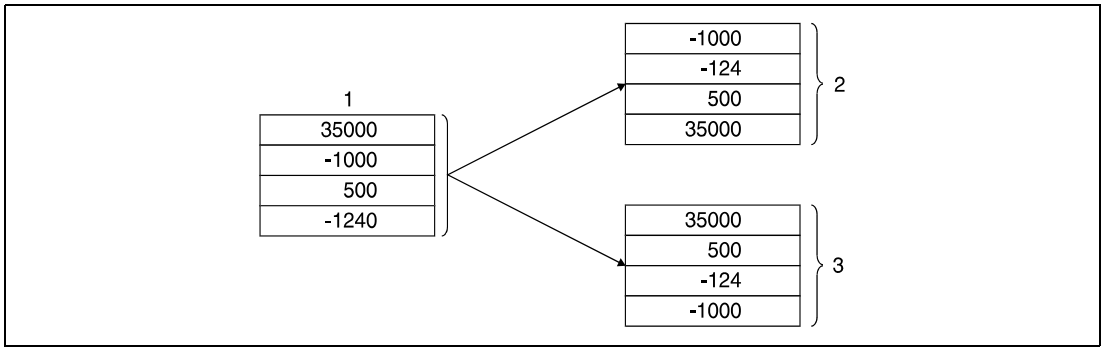
DSORT Sorting 32-bit data

The DSORT instruction sorts 32-data specified by s1 in ascending or descending order. The number of data to be sorted is specified by n.

The sorting order is set via the special relay SM703:

– SM703 OFF: Ascending order

– SM703 ON: Descending order



[1] Data to be sorted

[2] Data sorted in ascending order (SM703 = OFF)

[3] Data sorted in descending order (SM703 = ON)

For finishing the DSORT instruction several scans are required. The number of required scans can be calculated by the division of the maximum number of scans by the number of 32-bit data specified in s2, to be compared each scan (decimal fractions are rounded up). Increasing the number of 32-bit data specified in s2 reduces the number of required scans for sorting but increases the processing time per scan.

The required number of sorting scans until finishing the sort operation is calculated via the following equation:

Required number of sorting scans = ((n) x (n-1)) / (2 x (s2))

For example, for n = 10 and s2 = 1 the result is 45 sort scans until finishing the sort operation.

For n = 10 and s2 = 2 the result is 22.5. Rounded up, 23 sort scans are required.

The bit specified in d1 is reset during the sort operation and will be set again when the sort operation is finished. This bit remains set and must be reset by appropriate programming.

The devices specified in d2 and (d2)+1 are used for internal system processing during the sort operation. So, these devices must not be changed by programming.

If the value in n is changed during the operation, the operation is processed with the currently set number of 32-bit data.

By resetting the execution condition, the operation will be terminated. Upon setting the execution condition again, the sort operation will be restarted.

To execute another sort operation immediately after the completion of the previous sort, turn OFF the execution command once, then turn it ON.

Operation Errors

In the following cases an operation occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The range specified by n (SORT, SORTP) or 2 x n (DSORT, DSORTP) in the device specified by s1 exceeds the relevant storage device range.
  (Error code 4101)
- s2 is 0 or is a negative value.
  (Error code: 4100)
- The device range of the (n/2 x n) points starting from the device designated by s1 overlaps with the device range of the 2 points starting from the device designated by d2.
  (Error code: 4101)

Program Example

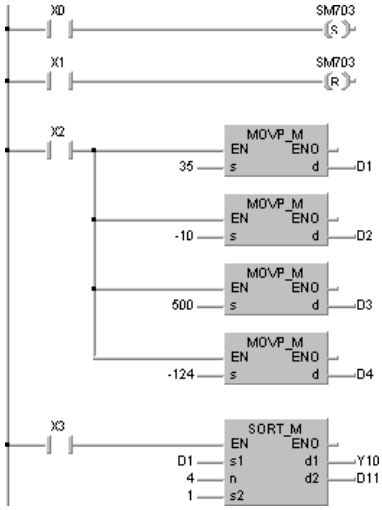
SORT

While X3 is set, the following program sorts 16-bit data in D1 through D4. In a first step with leading edge from X2, the values 35, -10, 500, and -124 are written to the registers D1 through D4. Then sorting starts. The sorting order is determined via X0 (set SM703) and X1 (reset SM703). After finishing the sort operation the output Y10 is set.

MELSEC Instruction List

```
MELSEC
        LD      X0
        SET     SM703
        LD      X1
        RST     SM703
        LD      X2
        MOVP    K35
                D1
        MOVP    K-10
                D2
        MOVP    K500
                D3
        MOVP    K-124
                D4
        LD      X3
        SORT    D1
                K4
                K1
                Y10
                D11
```

Ladder Diagram



IEC Instruction List

```
        LD      X0
        S       SM703
        LD      X1
        R       SM703
        LD      X2
        MOVP_M  35 , D1
        MOVP_M  -10 , D2
        MOVP_M  500 , D3
        MOVP_M  -124 , D4
        LD      X3
        SORT_M  D1 , 4 , 1 , Y10 , D11
```

7.5.13 WSUM, WSUMP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

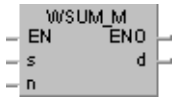
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	●	●	●	●	●	●	●	—	—
n	●	●	●	●	●	●	●	●	—

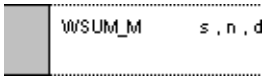
GX IEC Developer

MELSEC Instruction List

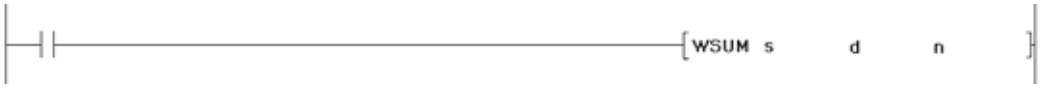
Ladder Diagram



IEC Instruction List



GX Works2



Variables

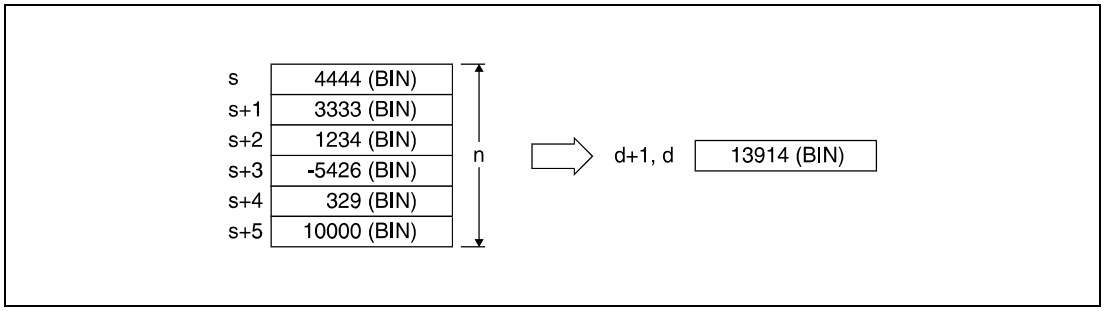
Set Data	Meaning	Data Type
s	First number of device storing data to be added	BIN 16-bit
d	First number of device storing result	BIN 32-bit
n	Number of data blocks to be added	BIN 16-bit

Functions

Calculating totals of 16-bit BIN data blocks

WSUM Calculation of totals

The WSUM instruction calculates the total of 16-bit data blocks in the device specified by s. The number of data blocks to be summed up is specified by n. The result is stored in the device specified by d.



Operation Errors

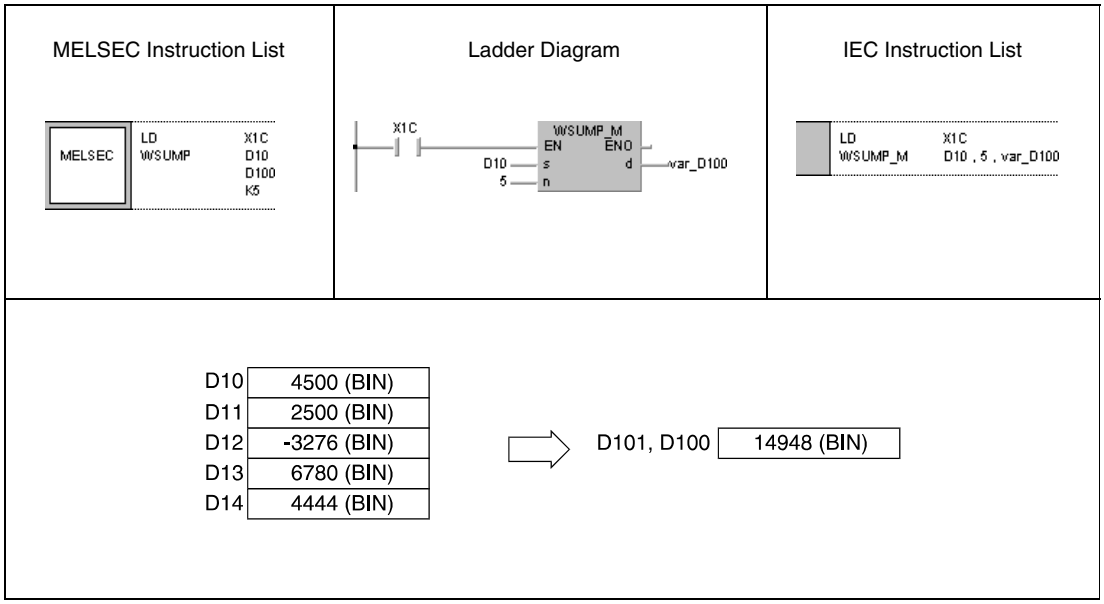
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The range specified by n in the device specified by s exceeds the relevant storage device range.
  (Error code 4101)

Program Example

WSUMP

With leading edge from X1C, the following program adds BIN 16-bit data blocks in D10 through D14 and stores the result in D100 and D101.



7.5.14 DWSUM, DWSUMP

CPU

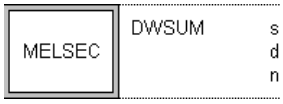
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

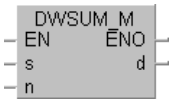
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	●	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

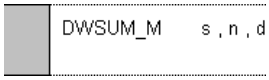
MELSEC Instruction List



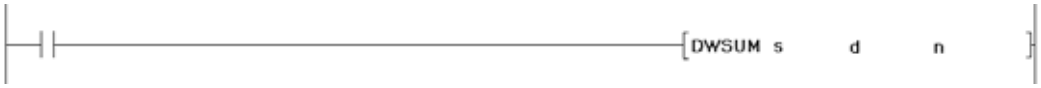
Ladder Diagram



IEC Instruction List



GX Works2



Variables

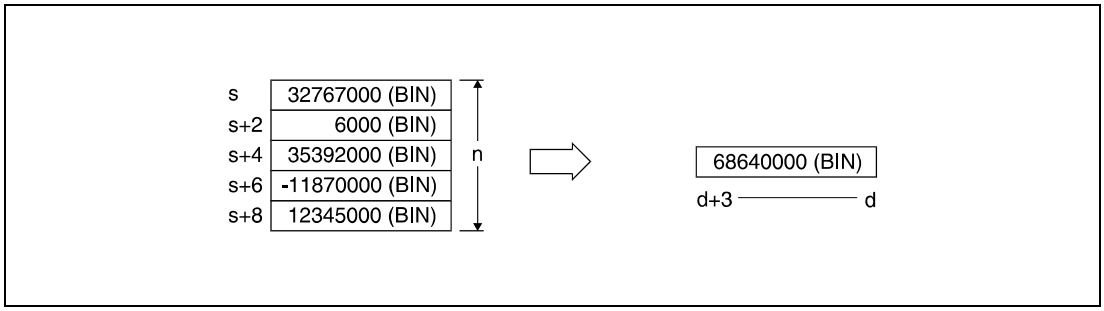
Set Data	Meaning	Data Type	
		MELSEC	IEC
s	First number of device storing data to be added	BIN 32-bit	ANY32
d	First number of device storing result	BIN 64-bit	Array [1..4] of ANY16
n	Number of data blocks to be added	BIN 16-bit	ANY16

Functions

Calculating totals of 32-bit BIN data blocks

DWSUM Calculation of totals

The DWSUM instruction calculates the total of 32-bit data blocks in the device specified by s. The number of data blocks to be summed up is specified by n. The result is stored in array[1] through array[4] in the device specified by d.



Operation Errors

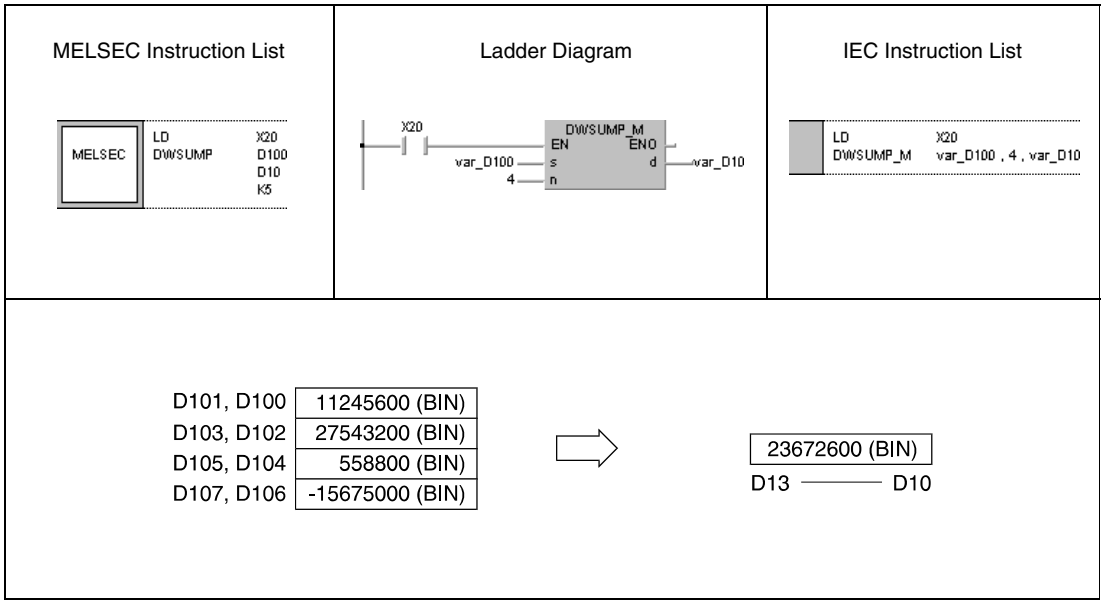
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The range specified by n in the device specified by s exceeds the relevant storage device range.
  (Error code 4101)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

DWSUMP

With leading edge from X20, the following program adds 32-bit BIN data blocks in D100 through D107 and stores the result in D10 through D13.



NOTE

This program will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.5.15 MEAN, MEANP, DMEAN, DMEANP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

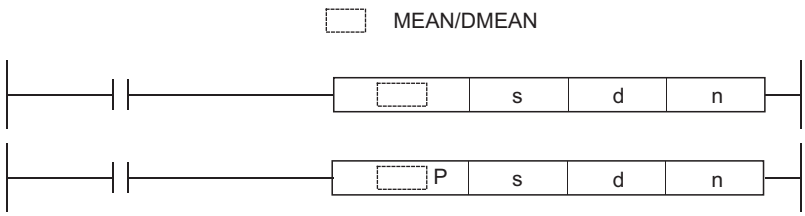
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—
n	—	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



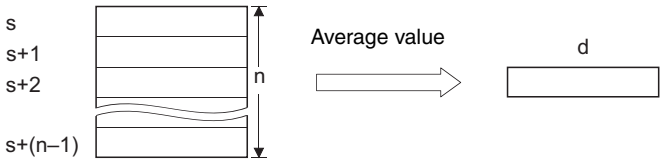
Variables

Set Data	Meaning	Data Type
s	First number of device storing data to be averaged	BIN 16/32-bit
d	First number of device storing result	
n	Number of data or number of devices where the number of data are stored (Setting range: 1 to 32767)	BIN 16-bit

Functions **Calculating averages of 16/32-bit BIN data**

MEAN **Calculation of averages (16-bit data)**

This instruction calculates the mean of 16-bit BIN data in n-point devices starting from the device specified by s. The result is stored in the device specified by d.

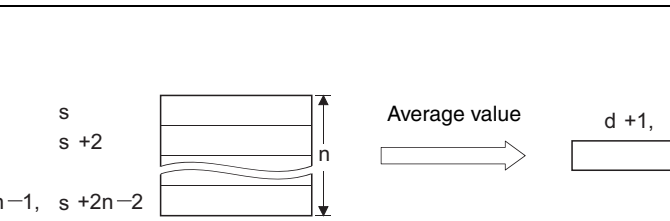


If the value calculated is not integer, this instruction will drop the number of decimal places.

If the value specified by n is 0, the instruction will be not processed.

DMEAN **Calculation of averages (32-bit data)**

This instruction calculates the mean of 32-bit BIN data in n-point devices starting from the device specified by s. The result is stored in the device specified by d.



If the value calculated is not integer, this instruction will drop the number of decimal places.

If the value specified by n is 0, the instruction will be not processed.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

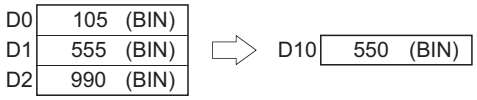
- The value specified by n is other than 0 to 32767.
  (Error code 4100)
- The range of the n-point devices starting from the device specified by s exceeds the range of the devices specified by d.
  (Error code 4101)

Program Example 1

MEAN

The following program stores the average value of 16-bit data stored from D0 to D2 into D10, when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



Program Example 2

DMEAN

The following program stores the average value of 32-bit data stored from D0 to D5 into D10 and D11, when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

7.6 Structured program instructions

Structured program instructions call programs and parts of programs or switch over between them. In addition, instructions for index qualification and program repetitions (loops) are supplied.

The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Repetition instructions	FOR	FOR_M
	NEXT	NEXT_M
	BREAK	BREAK_MD
	BREAKP	BREAK_P_MD
Subroutine program calls	CALL	CALL_M
	CALLP	CALLP_M
	RET	RET_M
	FCALL	FCALL_MD
	FCALLP	FCALL_P_MD
Subroutine calls between program files (only possible with GX Works2)	ECALL	ECALL_M
	ECALLP	ECALLP_M
	EFCALL	EFCALL_M
	EFCALLP	EFCALLP_M
	XCALL	
Select refresh	COM	COM_M
	CCOM	
	CCOMP	
Index qualification of entire ladders	IX	IX_MD
	IXEND	IXEND_MD
Designation of qualification values in index qualification of entire ladders	IXDEV	IXDEV_M
	IXSET	IXSET_M

7.6.1 FOR, NEXT

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

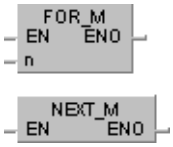
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
n	●	●	●	●	●	●	●	●	—

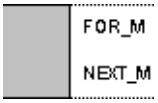
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
n	Number of repetitions of the FOR/NEXT loops (from 1 to 32767)	BIN 16-bit

Functions

FOR/NEXT loop instruction

FOR/NEXT Loop instruction

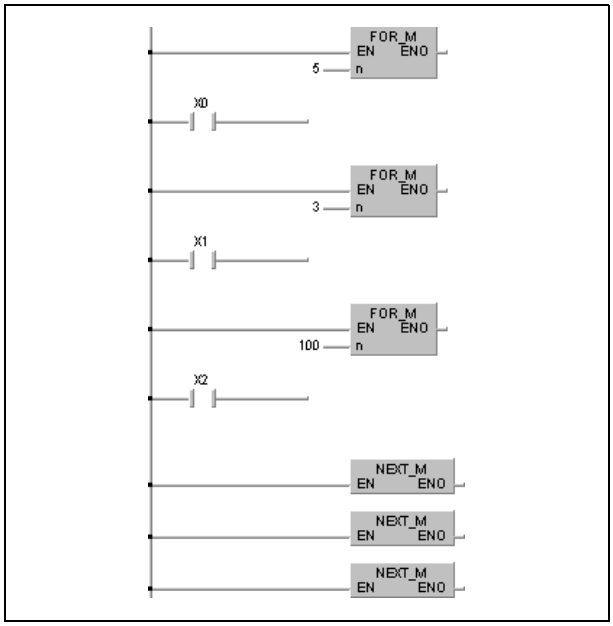
The FOR/NEXT loop repeats single program sequences without setting an input condition. The program sequence located between the FOR and the NEXT command is repeated for n times.

After executing the FOR/NEXT loop for n times, the next program step following the NEXT command is executed.

The variable n can be specified from 1 to 32767. If n is less than or equal to 0, it is processed as 1. Thus, the FOR/NEXT loop will be executed at least once.

If a program sequence between the FOR/NEXT loop is not intended to be executed, it can be skipped by a jump instruction (CJ or SCJ).

In total, up to 16 levels of FOR/NEXT loops can be nested up. The following diagram illustrates the principle of nesting:



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The END/FEND or GOEND instruction is executed after a FOR instruction and before the NEXT instruction.
  (Error code 4200)
- The NEXT instruction is executed before the FOR instruction.
  (Error code 4201)
- A STOP instruction is programmed within a FOR/NEXT loop.
  (Error code 4200)
- The maximum number of nesting levels is exceeded.
  (Error code 4202)

NOTES

In order to terminate the execution of a FOR/NEXT loop before it is finished, a BREAK instruction must be programmed.

Apply the EGP/EGF instruction, to connect a switch condition to the FOR/NEXT instruction.

Branching into a FOR to NEXT loop using a JMP or other branch instruction from the outside of the FOR to NEXT loop is not possible.

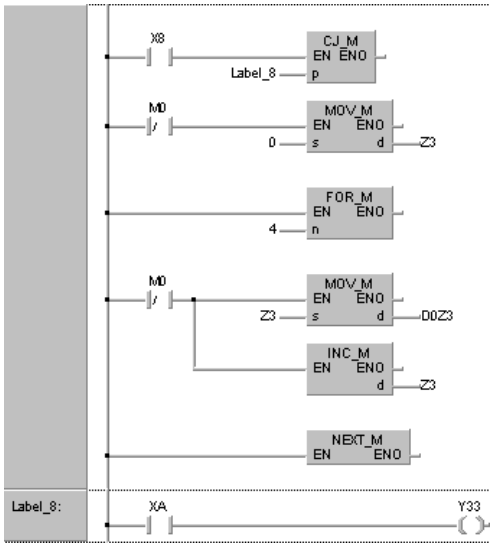
Program Example

The following program processes the program sequence between FOR and NEXT for four times, if X8 is OFF. The FOR/NEXT loop is skipped, if X8 is ON.

MELSEC Instruction List

```
MELSEC   LD    X8
         CJ    Label_8
         LDI   M0
         MOV   K0
               Z3
         FOR   K4
         LDI   M0
         MOV   Z3
               D0Z3
         INC   Z3
         NEXT
Label_8: LD    XA
MELSEC   OUT   Y33
```

Ladder Diagram



IEC Instruction List

```
         LD    X8
         JMPC  Label_8
         LDN   M0
         MOV_M 0 , Z3
         FOR_M 4
         LDN   M0
         MOV_M Z3 , D0Z3
         INC_M Z3
         NEXT_M
Label_8: LD    XA
         ST    Y33
```

7.6.2 BREAK, BREAKP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

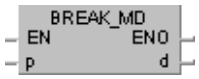
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				P
d	●	●	●	●	●	●	●	—	—
p	—	—	—	—	—	—	—	—	●

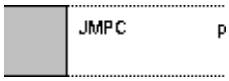
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List (IEC Instruction)



GX Works2

Variables

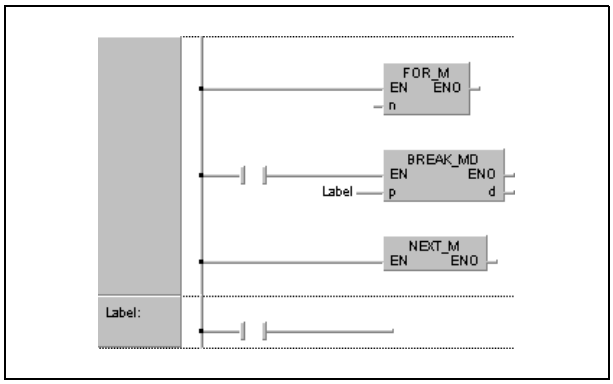
Set Data	Meaning	Data Type
d	Device storing the remaining number of FOR/NEXT loops	BIN 16-bit
p	Destination address (Pointer/Label) to be jumped to after executing the BREAK instruction	Pointer/label

Functions

Terminating a FOR/NEXT loop

BREAK Terminating the FOR/NEXT execution

The BREAK instruction terminates a FOR/NEXT loop execution and jumps to the pointer/label specified by p. Only a pointer within the same program file can be assigned to Pn. If a pointer of the other program file is used, an operation error will be returned.



The number of remaining FOR/NEXT loops to be executed is stored in the device specified by d. Note that the remaining number includes the operation when the BREAK instruction is executed.

The BREAK instruction can only be applied during the execution of a FOR/NEXT loop.

The BREAK instruction can only be applied to one nesting level. For several nesting levels the appropriate number of BREAK instructions must be executed.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The BREAK instruction was executed without a FOR/NEXT loop.
  (Error code 4203)
- The jump destination for the pointer designated by Pn does not exist.
  (Error code: 4210)
- The pointer of another program file is designated for Pn.
  (Error code: 4210)

Program Example

BREAKP

The following program terminates the execution in the 30th FOR/NEXT loop and jumps to the program part specified with label_0. The number of remaining FOR/NEXT loops (70) is stored in D1.

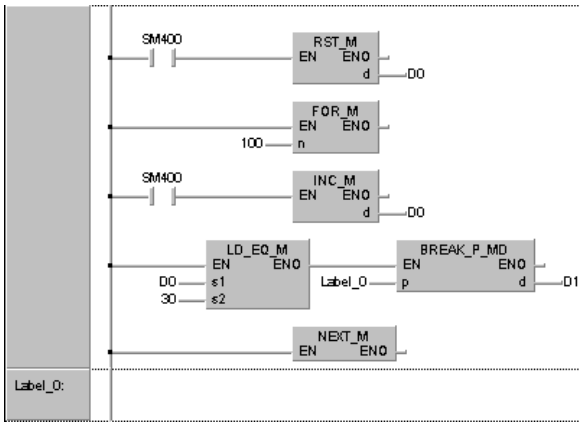
MELSEC Instruction List

```
MELSEC   LD       SM400
         RST      D0
         FOR      K100
         LD       SM400
         INC      D0
         LD=      D0
                  K30
         BREAKP   D1
                  Label_0
         NEXT
Label_0:
MELSEC
```

IEC Instruction List

```
         LD       SM400
         RST_M    D0
         FOR_M    100

         LD       SM400
         INC_M    D0
         LD_EQ_M  D0 , 30
         BREAK_P_MD Label_0 , D1
         NEXT_M
Label_0:
```

Ladder Diagram



7.6.3 CALL, CALLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

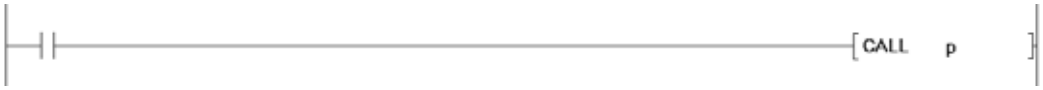
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				P
p	—	●	●	—	—	—	—	—	●
s1 – s5	●[1)]	●	●	●	●	●	●	●	—

[1] Annunciators (F) cannot be used

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
MELSEC CALL p	CALL_M EN ENO p	

GX Works2



Variables

Set Data	Meaning	Data Type
pn	Address number (pointer/label) of subroutine program	Pointer/label
s1 to s5	Number of the device to be passed as an argument to a subroutine program	bits, BIN 16-bit, BIN 32-bit

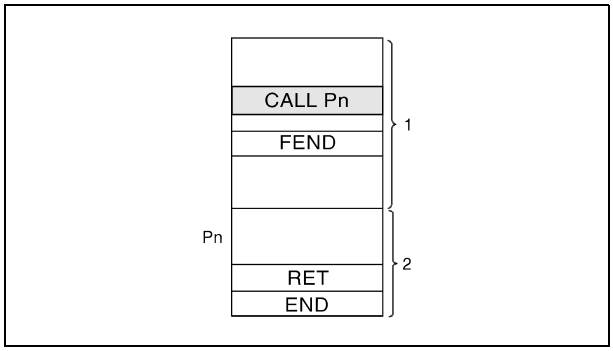
NOTE

The CALL instruction should not be used with the IEC editor because the subroutine structure is generated bx the GX IEC Developer.

Functions **Calling a subroutine program**

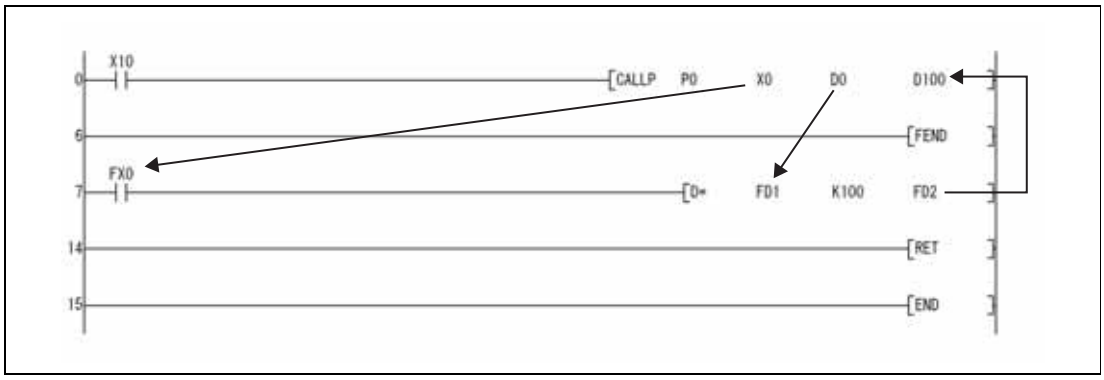
CALL Subroutine program call

The CALL instruction calls a subroutine program specified by a pointer Pxx in the GX Works2 or by a label in the GX IEC Developer, respectively. The pointer (label) addresses range from P(label)0 to P(label)4095. Refer to the notes on programming pointer (label) addresses for the jump instructions (CJ, SCJ, JMP).



[1] Main routine program
[2] Subroutine program

When function devices (FX, FY, FD) are used by a subroutine program, specify a device with s1 to s5 corresponding to the function device. The contents to the devices specified by s1 to s5 are as indicated below.



Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.

After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.

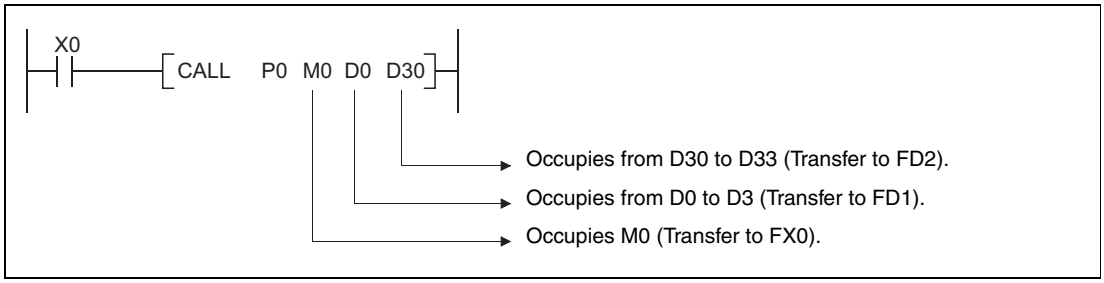
The processing units for the function devices are as follows:

- FX, FY: Bits
- FD: 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function devices	Device	Data Size	Remark
FX FY	Bit device	1 point	—
	When bit designation is made for word device	1 bit	
FD	When digit designation of a bit device is used [1)]	4 words	The data size varies depending on the instruction to be used.
	Word device	4 words	

[1] An error will not occur even when the device number specified by s1 to s5 is not a multiple of 16 at the digit designation of the bit device.



s1 to s5 can be used with the CALL (P) instruction.

The number of function devices to be used by a subroutine program must be identical to the number of arguments in the CALL (P) instruction.
Also, the types of the function device and CALL (P) argument used should be identical.

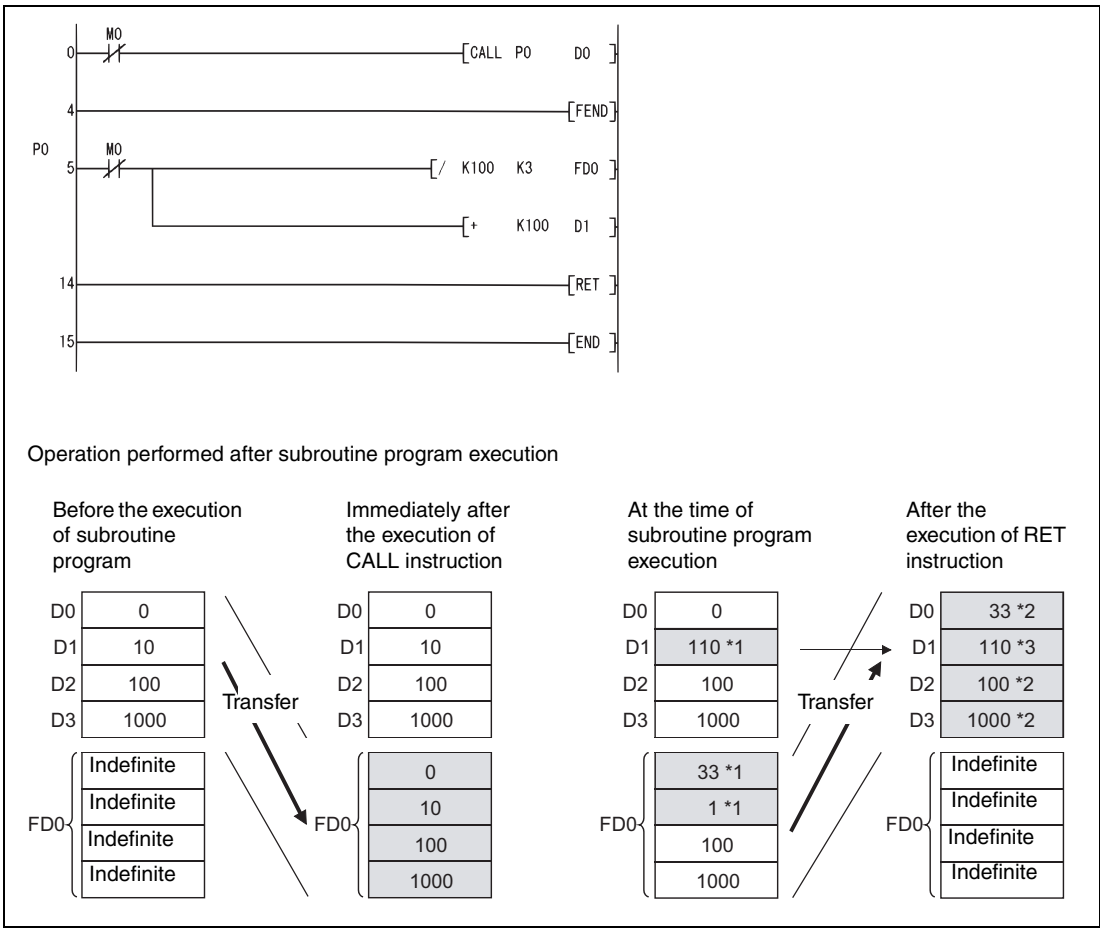
Device numbers specified by the CALL (P) instruction should not overlap. If they do overlap, it will not be possible to obtain accurate calculations.

The device used in the argument of the CALL (P) instruction should not be used in a subroutine program. If used, it will not be possible to obtain accurate calculations. (Refer to the following program example.)

When the device, either timer or counter, is used in the argument of the CALL(P) instruction, only the current value is transmitted/received.

Incorrect operation example

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D1 is used in the subroutine program.



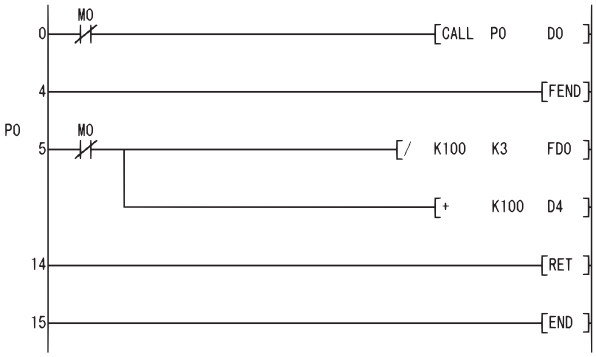
[1] Stores the execution result of the subroutine program.

[2] Replaced by the value of the function device.

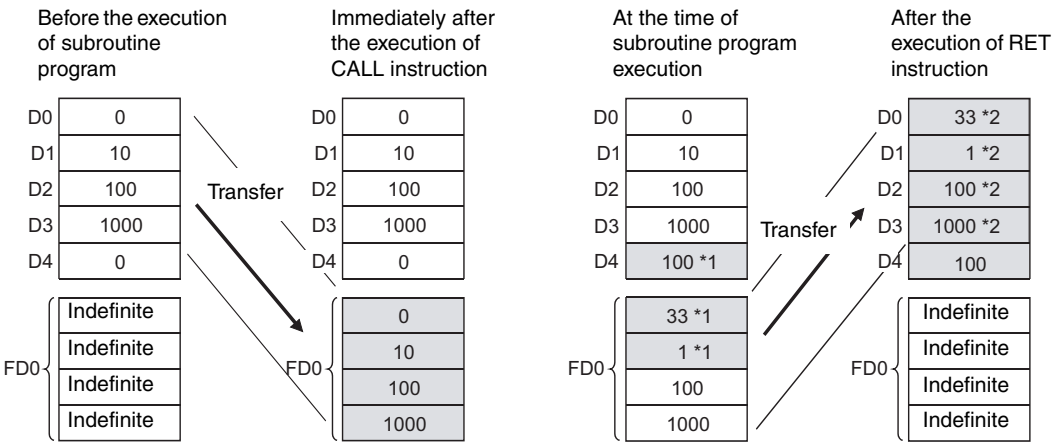
[3] D1 does not reflect the value of the function device.

Correct operation example

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D4 is used in the subroutine program.



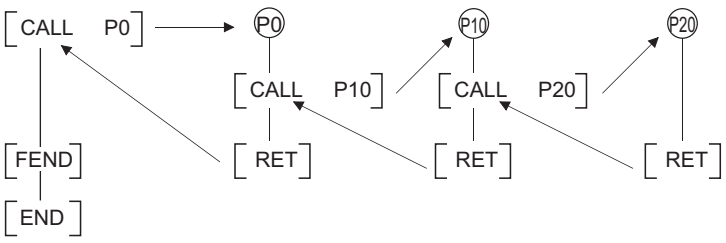
Operation performed after subroutine program execution



[1] Stores the execution result of the subroutine program.

[2] Replaced by the value of the function device.

The CALL instruction calls a subroutine program specified by pointer (label) addresses. In total, up to 16 subprogram nesting levels can be addressed.



Devices that were set during the execution of a subroutine program remain set, even if the routine is not executed any longer. In order to reset these devices the FCALL instruction has to be applied.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

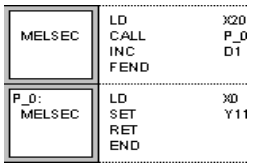
- The device specified for the argument cannot be secured for the data size. (Error code 4101)
- After execution of a CALL instruction an END, FEND, GOEND, or STOP instruction is executed, without a prior RET instruction. (Error code 4211)
- A RET instruction is executed before a CALL instruction. (Error code 4212)
- More than 16 nesting levels are executed. (Error code 4213)
- There is no subroutine program stored at the specified pointer/label. (Error code 4210)

Program Example

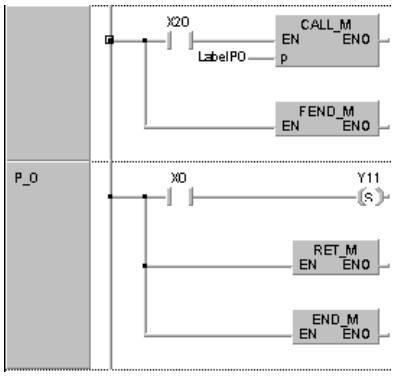
CALL

While X20 is set, the following program executes the subroutine program at pointer/label P_0.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

NOTES

In MELSEC-mode, the FEND, END, and RET instructions have to be programmed by the user. After the program organization unit has been processed no further one will be executed because it would follow the FEND instruction.

Alternatively to this programming, the IEC editor can be used. In that case the FEND instruction would be set by the compiler of the GX IEC Developer automatically.

7.6.4 RET

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

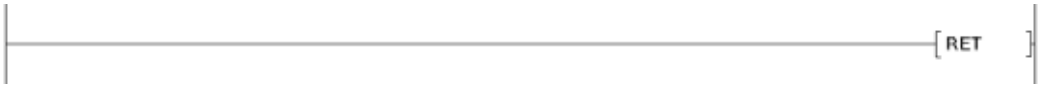
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

GX Works2



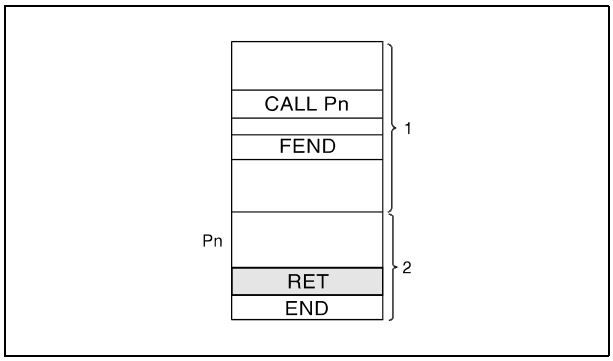
Variables

Set Data	Meaning	Data Type
—	—	—

Functions **End of subroutine program**

RET Return to main program

The RET instruction marks the end of a subroutine program. The program jumps back to the program step, that is specified after the CALL, FCALL, ECALL, EFCALL or XCALL instruction.



[1] Main routine program

[2] Subroutine program

NOTE *In the MELSEC-mode the FEND, END, and RET instructions have to be programmed by the user. After the program organization unit has been processed no further one will be executed because it would follow the FEND instruction.*

Alternatively to this programming, the IEC editor can be used. In that case the FEND instruction would be set by the compiler of the GX IEC Developer automatically.

Operation Errors In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- After execution of a CALL(P), FCALL(P), ECALL(P), EFCALL(P) or XCALL instruction an END, FEND, GOEND, or STOP instruction is executed, without a prior RET instruction. (Error code 4211)
- An RET instruction is executed before a CALL(P), FCALL(P), ECALL(P), EFCALL(P) or XCALL instruction. (Error code 4212)

7.6.5 FCALL, FCALLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				P
pn	—	—	—	—	—	—	—	—	●
s1 – s5	●[1)]	●	●	●	●	●	●	●	—

[1] Annunciators (F) cannot be used

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2

Variables

Set Data	Meaning	Data Type
pn	Address number (pointer/label) of subroutine program	Pointer/label
s1 to s5	Number of the device to be passed as an argument to a subroutine program	Bit BIN16-bit BIN 32-bit

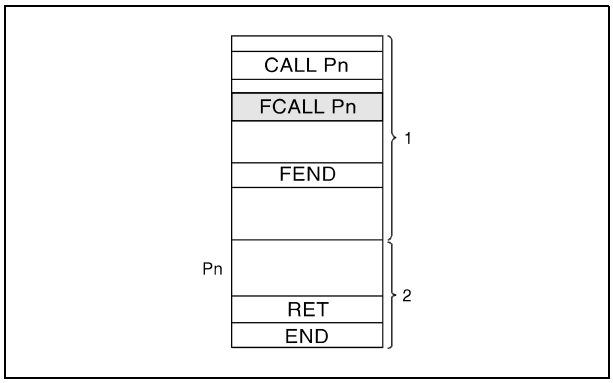
NOTE *These instructions are not available in GX IEC Developer.*

Functions **Resetting outputs in subroutine programs**

FCALL Resetting outputs (in conjunction with CALL instruction)

The FCALL instruction can execute subroutine programs designated by a pointer within the same program file, and subroutine programs designated by common pointers.

On resetting the execution condition for the FCALL instruction, the contacts and coils in the subroutine program specified in p (pointer/label) are treated as if the execution condition of the according instruction was not set.



[1] Main routine program

[2] Subroutine program

The condition of coils and contacts after execution of the FCALL instruction or the respective condition of coils and contacts with the according execution condition not set is listed below:

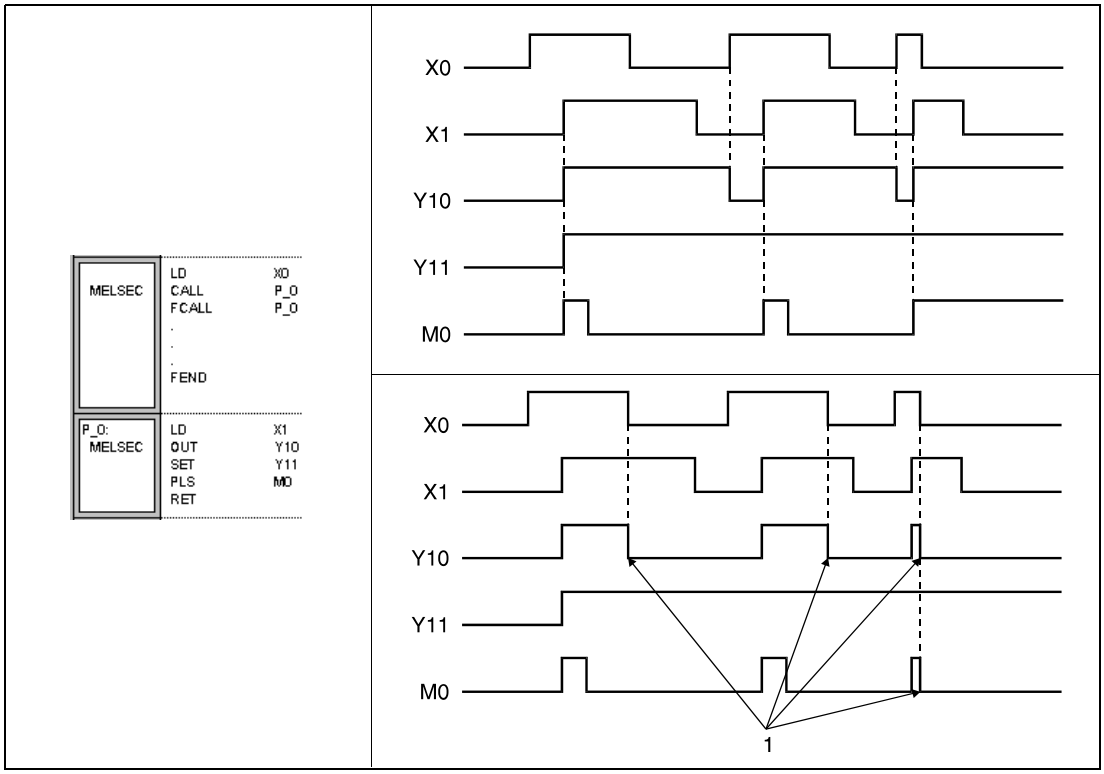
Instruction	Condition of contacts and coils
OUT instruction	All contacts and coils, designated by the OUT instruction are reset.
SET instruction	All contacts and coils, designated by these instructions remain their condition.
RST instruction	
SFT instruction	
Basic instructions	
Application instructions	
PLS instruction	All contacts and coils, designated by these instructions adopt a condition as if the execution conditions of the instructions were not set.
Instructions generating an output pulse	
Setting values of low- and high-speed timers	The setting values are reset to 0.
Setting values of retentive timers	The setting values remain set.
Setting values of counters	

The FCALL instruction is used in conjunction with a CALL instruction.

The following diagrams show a program, applying the CALL and FCALL instructions. The diagrams on the right show the signal condition of several contacts designated by several several instructions. The diagram on the top right shows the contact conditions without applying an FCALL instruction. The diagram on the bottom right shows the contact conditions applying an FCALL instruction.

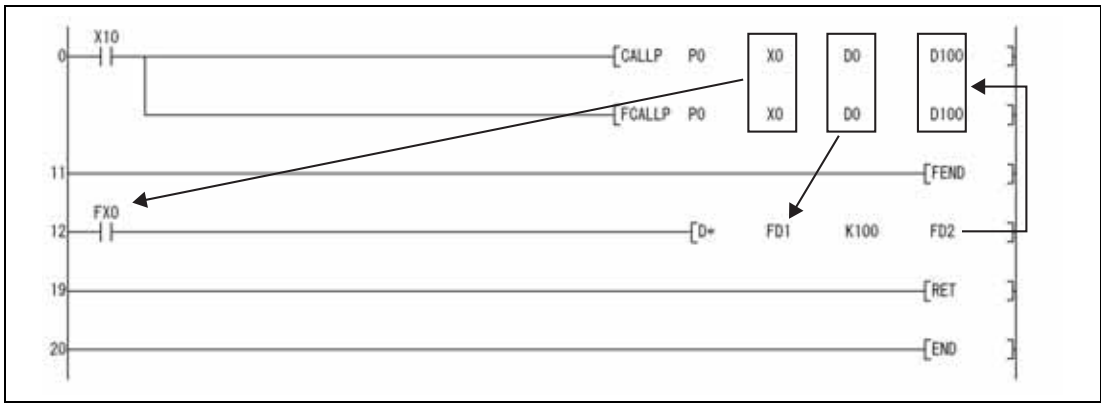
If only the CALL instruction is applied, the conditions of contacts and coils designated in a subroutine program are remained after resetting the execution condition of the CALL instruction (see diagram on top right).

If the FCALL instruction is applied, the conditions of contacts and coils designated in a subroutine program are reset after resetting the execution condition of the FCALL instruction (see diagram on bottom right). The same applies to coils and contacts designated by an OUT or PLS instruction, or by a pulse generating instruction.



[1] Forced OFF by FCALL instruction

When function devices (FX, FY, FD) are used by a subroutine program, specify a device with s1 to s5 corresponding to the function device. The contents to the devices specified by s1 to s5 are as indicated below.



Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.

After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.

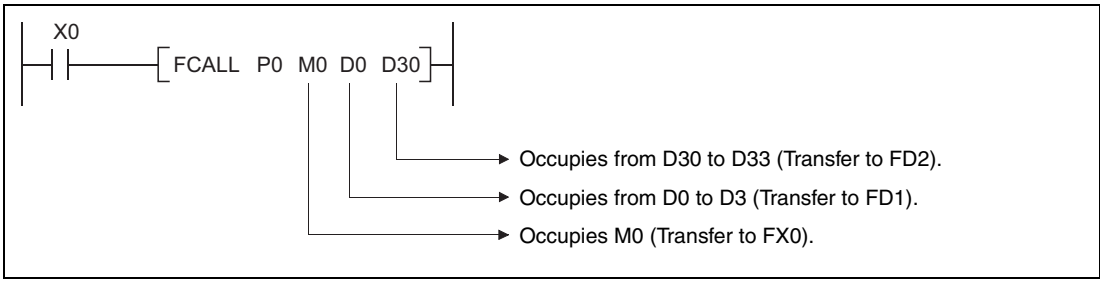
The processing units for the function devices are as follows:

- FX, FY: Bits
- FD : 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

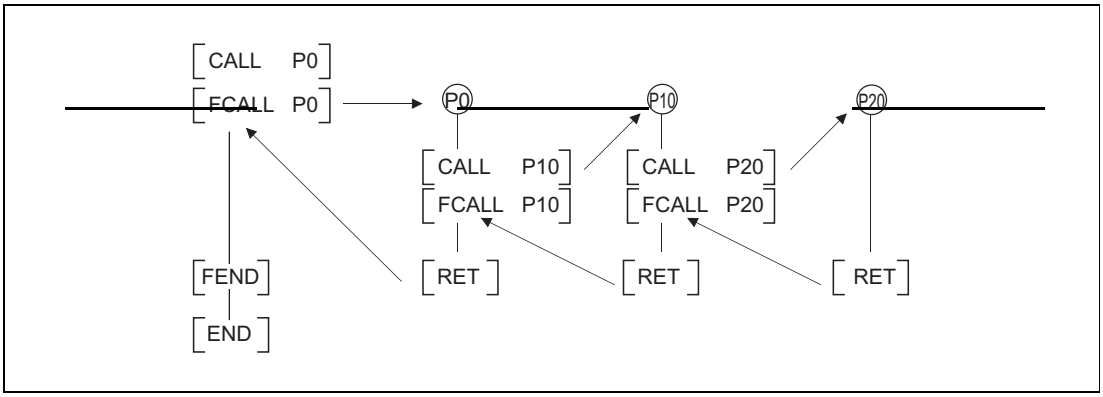
Function Devices	Device	Data Size	Remark
FX FY	Bit device	1 point	—
	When bit designation is made for word device	1 bit	
FD	When digit designation of a bit device is used [1)]	4 words	The upper 2 words of FD become 0
	Word device	4 words	—

[1] An error will not occur even when the device number specified by s1 to s5 is not a multiple of 16 at the digit designation of the bit device.



s1 to s5 can be used with the FCALL (P) instruction.

Up to 16 nesting levels are possible with the FCALL(P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



Operation Errors

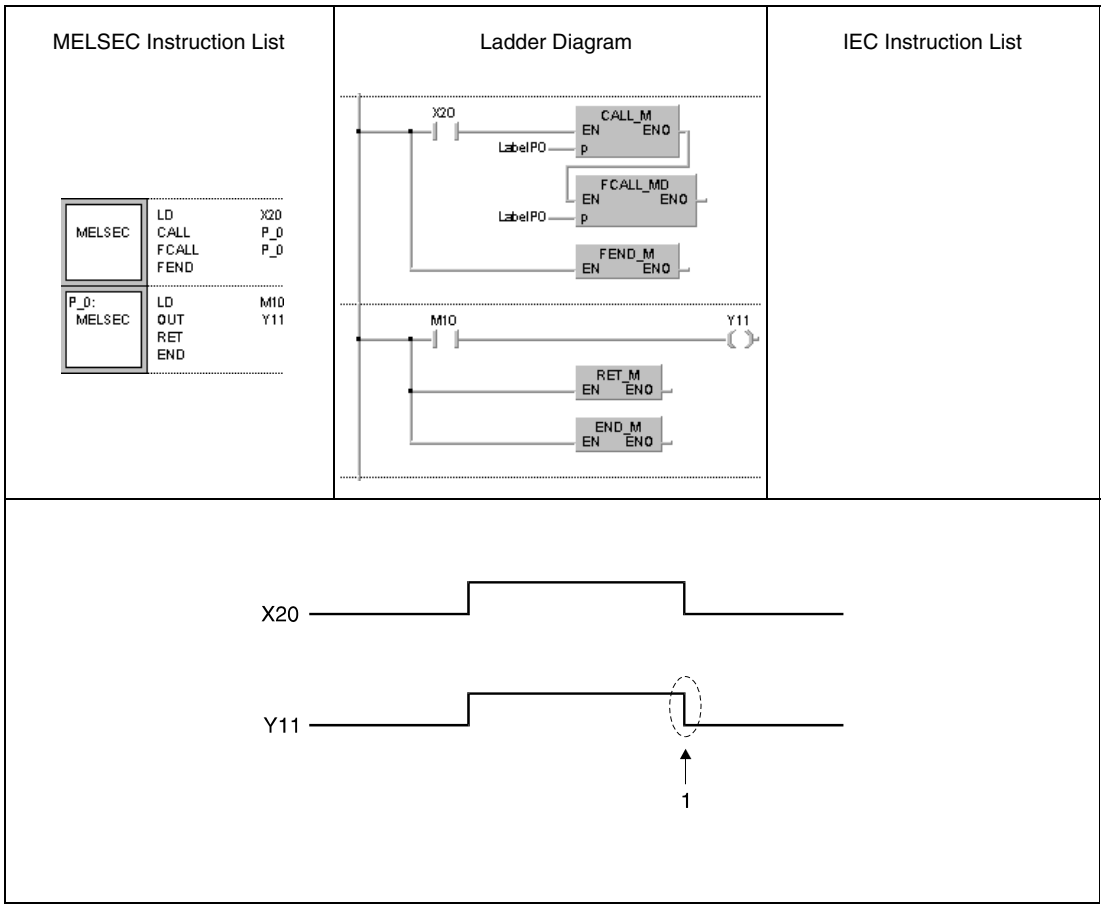
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified for the argument cannot be secured for the data size. (Error code 4101)
- After execution of an FCALL instruction an END, FEND, GOEND, or STOP instruction is executed, without a prior RET instruction. (Error code 4211)
- A RET instruction is executed before an FCALL instruction. (Error code 4212)
- More than 16 nesting levels are executed. (Error code 4213)
- There is no subroutine program stored at the specified pointer/label. (Error code 4210)

Program Example

FCALL

While X20 is set, the following program executes the subroutine program at pointer address (label) P_0. If X20 is reset, the FCALL instruction resets the output Y11 as well (1).



7.6.6 ECALL, ECALLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				P
n [1)]	—	●	●	—	—	—	—	●	—
pn	—	—	—	—	—	—	—	—	●
s1 – s5	●[2)]	●	●	●	●	●	●	—	—

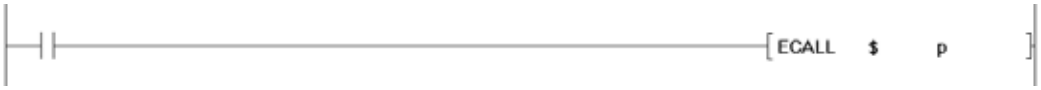
[1] File name

[2] Annunciators (F) cannot be used

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

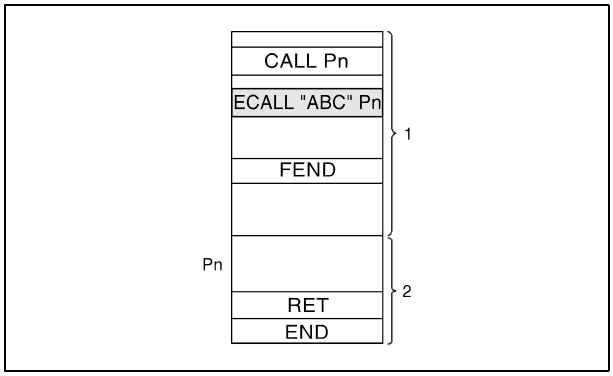
Set Data	Meaning	Data Type
File name	Name of progarm file containing the subroutine program	Character string
pn	Address number (pointer/label) of subroutine program	Pointer/label
s1 to s5	Device number that passes to subroutine	Bit BIN16-bit BIN 32-bit

NOTE *These instructions are not available in GX IEC Developer.*

Functions **Calling a subroutine program in a program file**

ECALL Subroutine program call

The ECALL instruction calls a subroutine program specified by pointer address (label) in a program file specified by a file name. The pointer address (label) ranges from P(label)0 to P(label)4095. Refer to the notes on programming pointer (label) addresses for the jump instructions (CJ, SCJ, JMP).



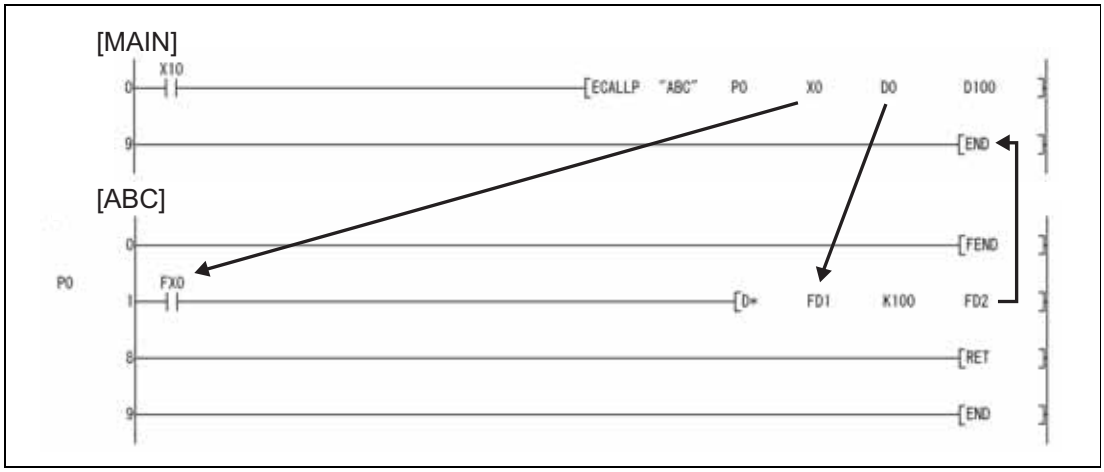
[1] Main routine program (file name: "MAIN")

[2] Subroutine program (file name: "ABC")

Only files stored in internal memory (drive 0) can be specified by the file name.

When calling program files no file extension is required. (Only ".QPG" files will be acted on.)

When function devices (FX, FY, FD) are used by a sub-routine program, specify a device with s1 through s5 corresponding to the function device. The contents to the devices specified by s1 to s5 are as indicated below.



Prior to execution of the sub-routine program, bit data is transmitted to FX, and word data is transmitted to FD.

After the execution of the sub-routine program, the contents of FY and FD are transmitted to the corresponding device.

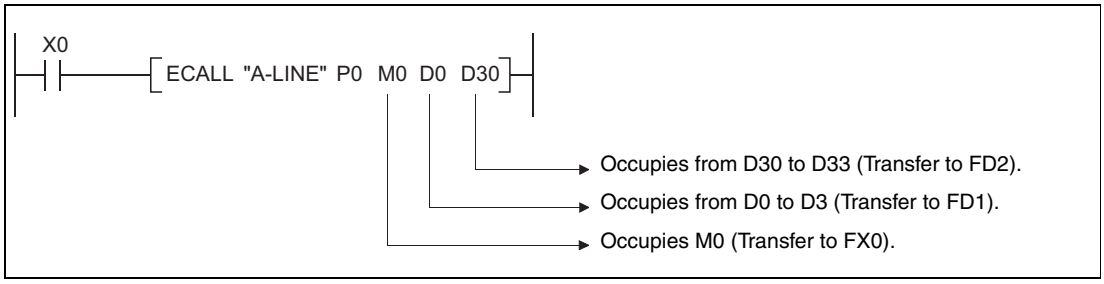
The processing units for the function devices are as follows:

- FX, FY: Bits
- FD : 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function Devices	Device	Data Size	Remark
FX FY	Bit device	1 point	—
	When bit designation is made for word device	1 bit	
FD	When digit designation of a bit device is used [1)]	4 words	The data size varies depending on the instruction to be used.
	Word device	4 words	

[1] An error will not occur even when the device number specified by s1 to s5 is not a multiple of 16 at the digit designation of the bit device.

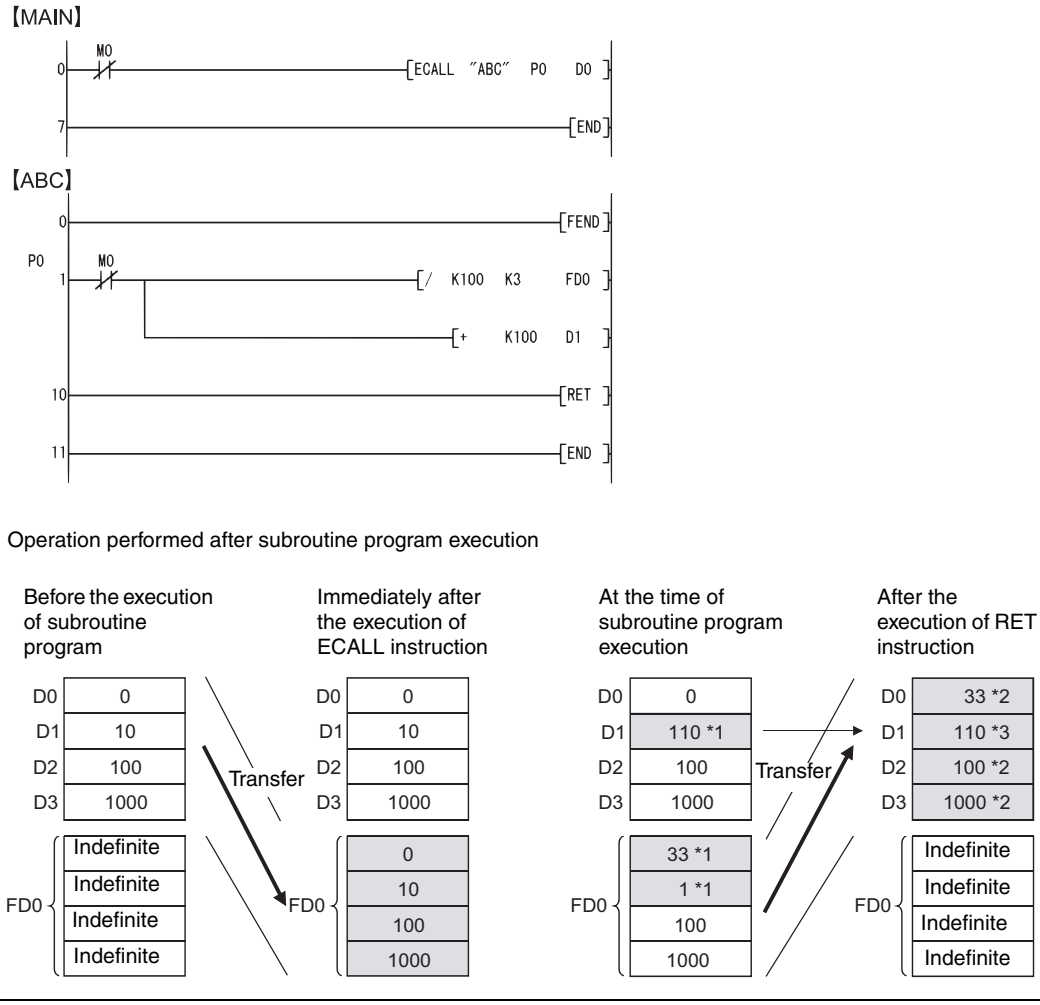


s1 to s5 can be used with the ECALL instruction.

The device used in the argument of the ECALL instruction should not be used in a subroutine program. If used, it will not be possible to obtain accurate calculations. (Refer to the following program example.)

Incorrect operation example

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D1 is used in the subroutine program.



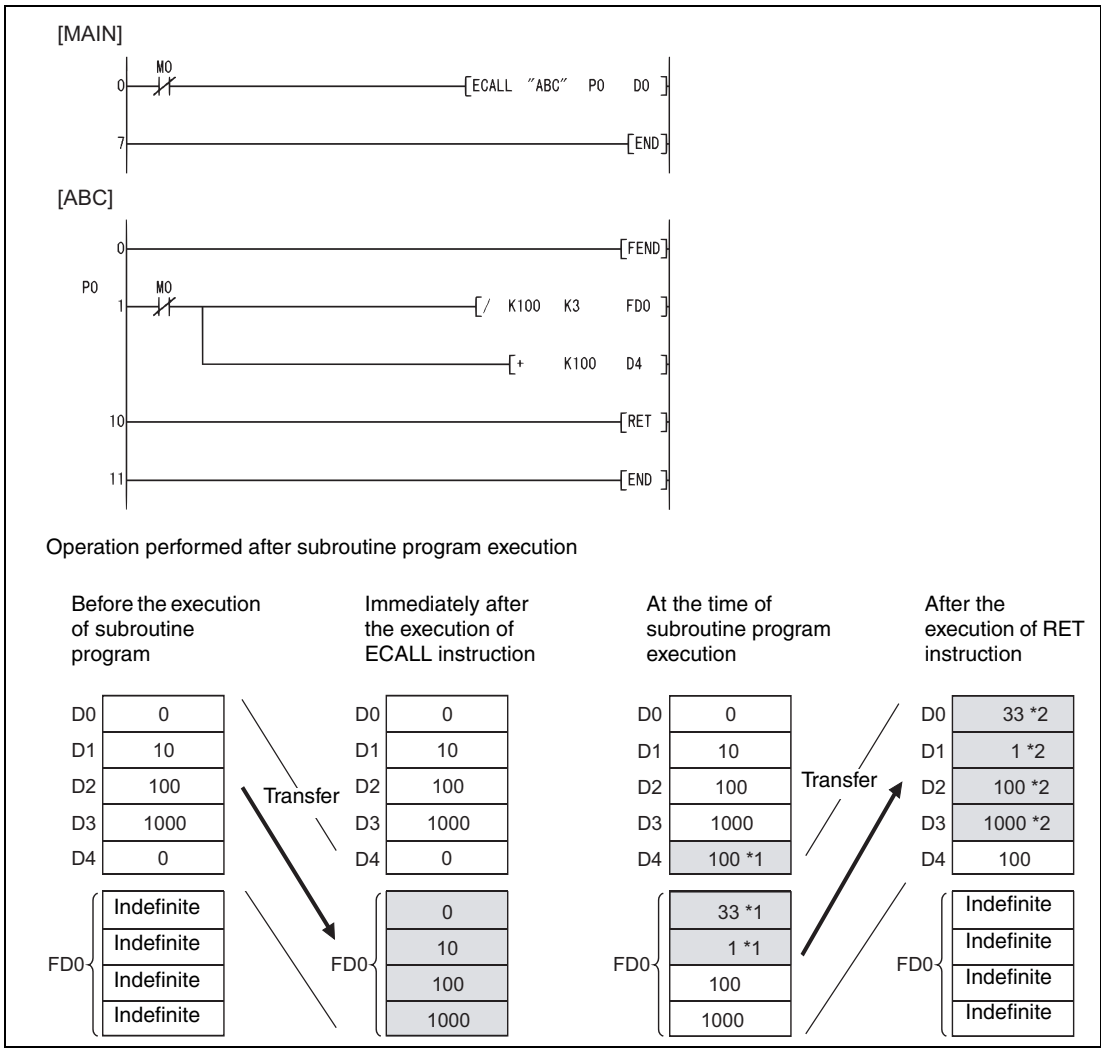
[1] Stores the execution result of the subroutine program.

[2] Replaced by the value of the function device.

[3] D1 does not reflect the value of the function device.

Correct operation example

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D4 is used in the subroutine program.

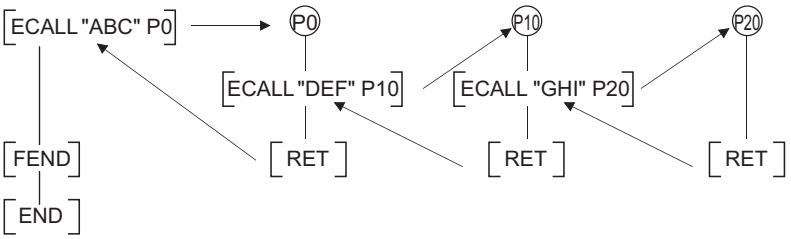


[1] Stores the execution result of the subroutine program.

[2] Replaced by the value of the function device.

The devices specified in s1 through s5 must not overlap. If they do overlap, it will not be possible to obtain accurate calculations.

Up to 16 levels of nesting can be used with the ECALL(P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



Devices which are turned ON within subroutine programs will be latched even if the subroutine program is not executed. Devices turned ON during the execution of a subroutine program can be turned OFF by the EFCALL(P) instruction.

Operation Errors

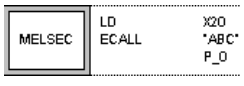
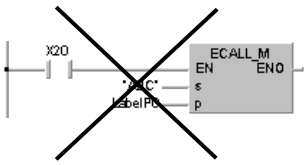
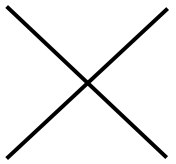
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified for the argument cannot be secured for the data size.
  (Error code 4101)
- After execution of an ECALL instruction an END, FEND, GOEND, or STOP instruction is executed, without a prior RET instruction.
  (Error code 4211)
- An RET instruction is executed before an ECALL instruction.
  (Error code 4212)
- More than 16 nesting levels are executed.
  (Error code 4213)
- There is no subroutine program stored at the specified pointer/label.
  (Error code 4210)
- The specified program file does not exist.
  (Error code 4210)
- The specified program file cannot be executed.
  (Error code 2411)

Program Example

ECALL

While X20 is set, the following program executes the subroutine program at pointer/label P_0 in the program file "ABC".

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

7.6.7 EFCALL, EFCALLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				P
n[1)]	—	●	●	—	—	—	—	●	—
pn	—	—	—	—	—	—	—	—	●
s1 – s5	●[2)]	●	●	●	●	●	●	—	—

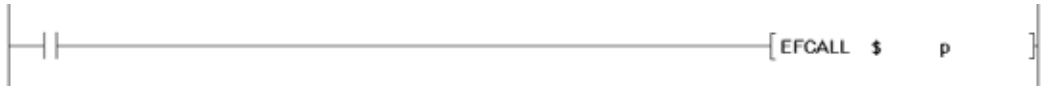
[1] File name

[2] Annunciators (F) cannot be used

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
file name	Name of program file containing the subroutine program	Character string
pn	Address number (pointer/label) of subroutine program	Pointer/label
s1 to s5	Device number that passes to subroutine	Bit BIN16-bit BIN 32-bit

NOTE *These instructions are not available in GX IEC Developer.*

Functions **Resetting outputs in subroutine programs in program files**

EFCALL **Resetting outputs (in conjunction with ECALL)**

On resetting the execution condition for the EFCALL instruction, the contacts and coils in the subroutine program specified in p (pointer/label) are treated as if the execution condition of the according instruction was not set.

The EFCALL instruction executes subroutine programs, that are located within a different program file from that one calling them.

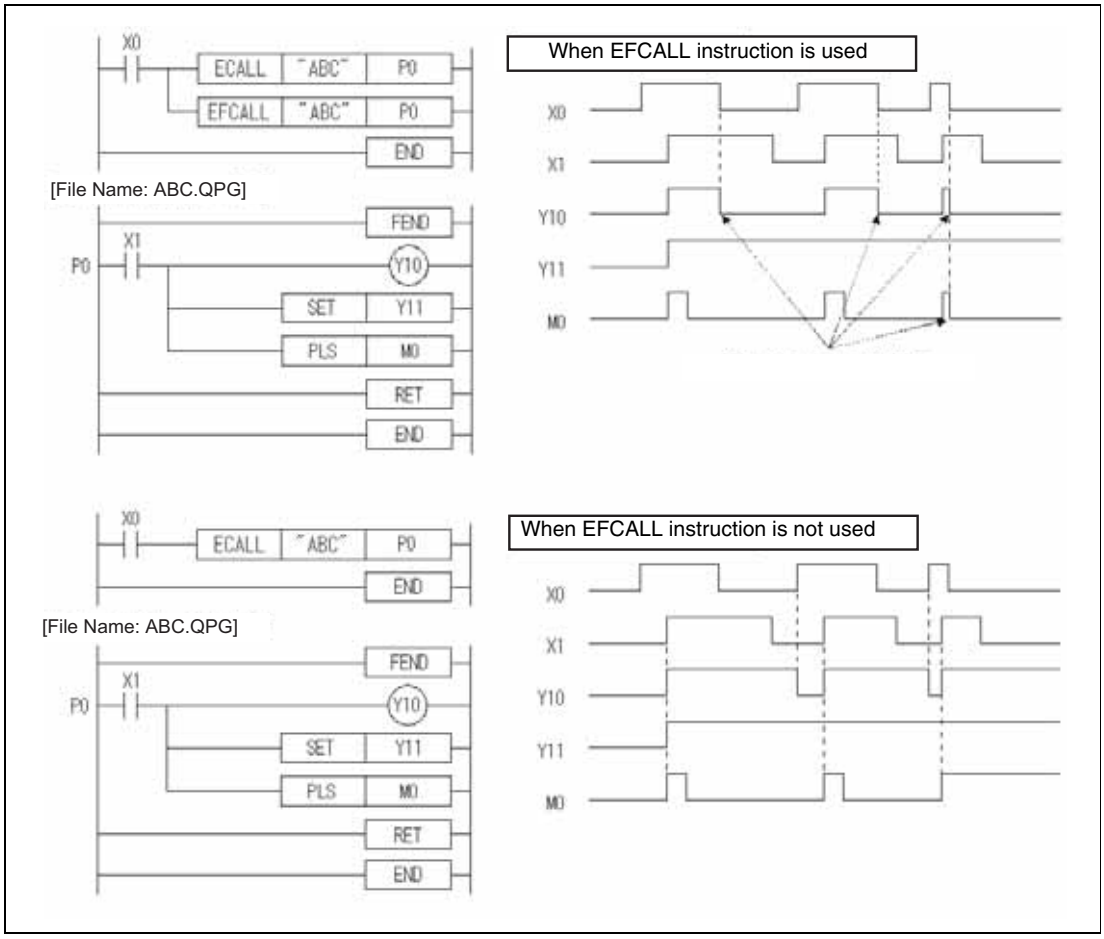
The condition of coils and contacts after execution of the EFCALL instruction or the respective condition of coils and contacts with the according execution condition not set is listed below:

Instruction	Condition of contacts and coils
OUT instruction	All contacts and coils, designated by the OUT instruction are reset.
SET instruction	All contacts and coils, designated by these instructions remain their condition.
RST instruction	
SFT instruction	
Basic instructions	
Application instructions	
PLS instruction	All contacts and coils, designated by these instructions adopt a condition as if the execution conditions of the instructions were not set.
Instructions generating an output pulse	
Setting values of low- and high-speed timers	The setting values are reset to 0.
Setting values of retentive timers	The setting values remain set.
Setting values of counters	

The EFCALL instruction is used in conjunction with an ECALL instruction.

If the EFCALL(P) instruction is used in conjunction with the ECALL(P) instruction, non-execution processing of a subroutine program is performed when the execution command is turned OFF, enabling forcible turning OFF of the OUT instruction and the PLS instruction (including P instructions).

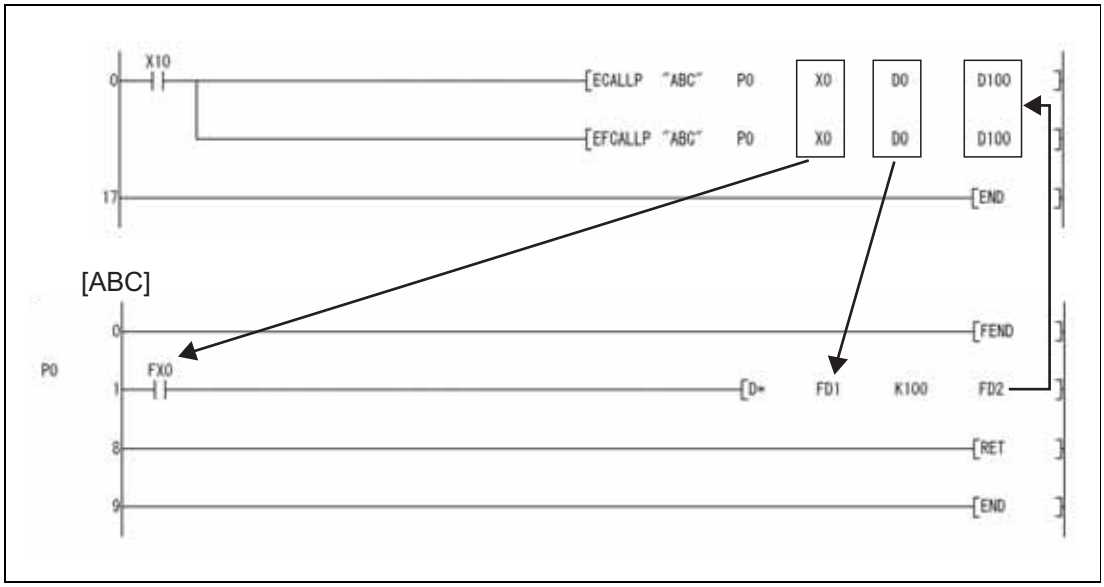
In case the EFCALL(P) instruction is not used in conjunction with the ECALL(P) instruction, non-execution processing of a subroutine program is not performed even if the execution command is turned OFF. Therefore, output status of the individual coil instructions remains unchanged.



Only the file name of a program file stored in the drive 0 (program memory/internal RAM) can be designated for a file name.

It is not necessary to designate the extension (".QPG") with the file name. (Only ".QPG" files will be acted on.)

When function devices (FX, FY, FD) are used by a sub-routine program, specify a device with s1 through s5 corresponding to the function device (see following figure).



Prior to execution of the sub-routine program, bit data is transmitted to FX, and word data is transmitted to FD.

After the execution of the sub-routine, the contents of FY and FD are transmitted to the corresponding device.

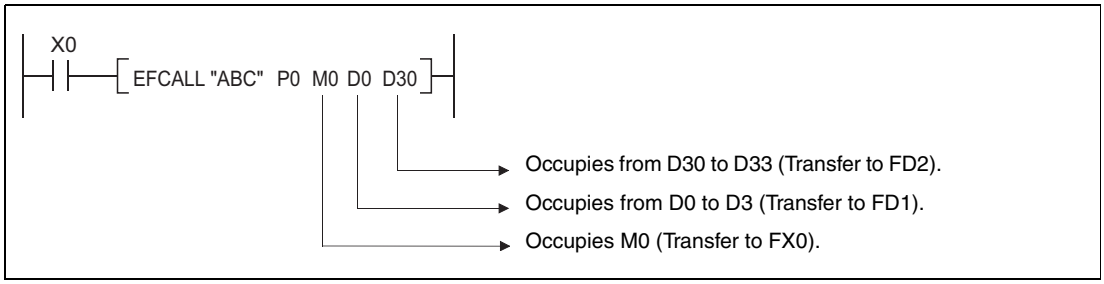
The processing units for the function devices are as follows:

- FX, FY: Bits
- FD : 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function Devices	Device	Data Size	Remark
FX FY	Bit device	1 point	—
	When bit designation is made for word device	1 bit	
FD	When digit designation of a bit device is used [1)]	4 words	The upper 2 words of FD become 0
	Word device	4 words	—

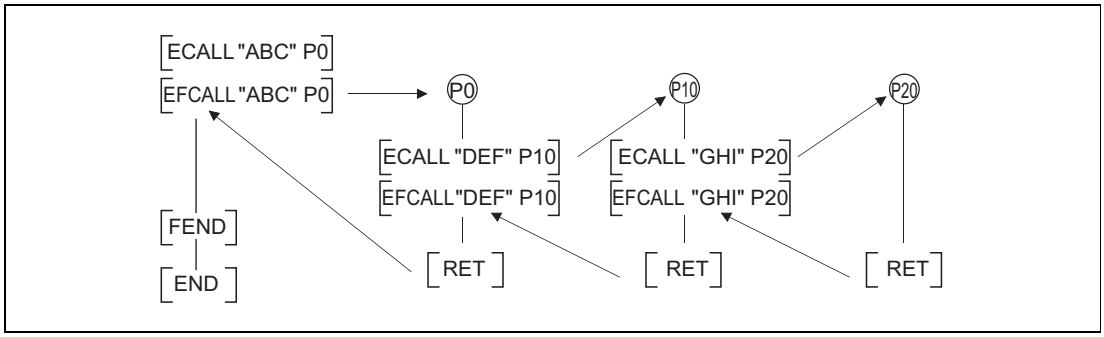
[1] An error will not occur even when the device number specified by s1 to s5 is not a multiple of 16 at the digit designation of the bit device.



s1 to s5 can be used with the EFCALL (P) instruction.

The number of function devices used by sub-routine programs must be identical to the number of devices handed over by the EFCALL(P) instruction in s1 through s5. The function devices must be identical to the types of devices handed over by the EFCALL(P) instruction.

The EFCALL(P) instruction calls a subroutine program specified via the pointer address (label). In total up to 16 nesting levels can be programmed. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P) and XCALL instructions.



Operation Errors

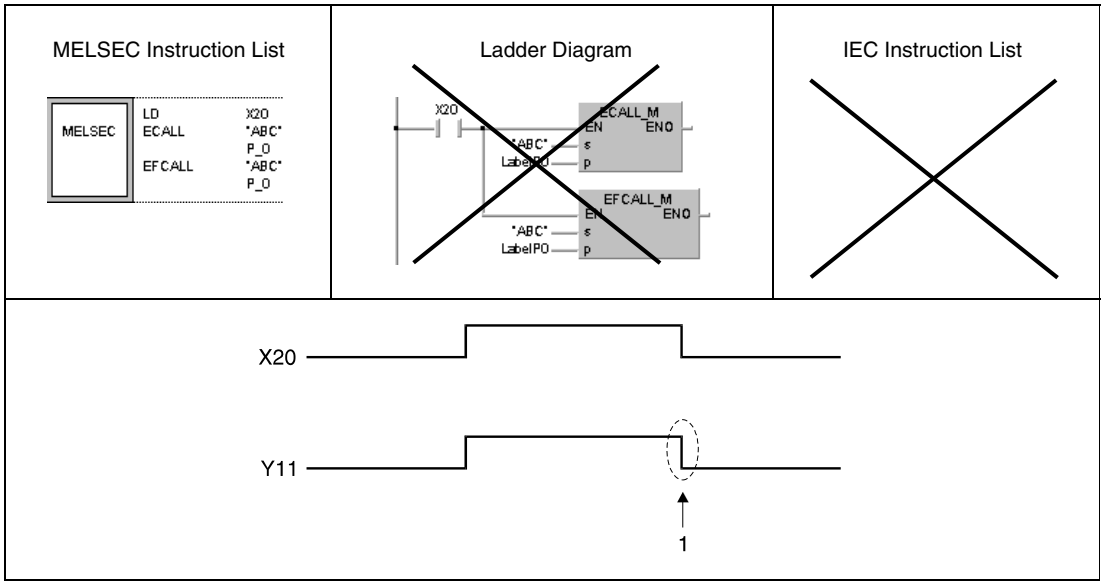
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified for the argument cannot be secured for the data size.
  (Error code 4101)
- After execution of an EFCALL instruction an END, FEND, GOEND, or STOP instruction is executed, without a prior RET instruction.
  (Error code 4211)
- An RET instruction is executed before an EFCALL instruction.
  (Error code 4212)
- More than 16 nesting levels are executed.
  (Error code 4213)
- There is no subroutine program stored at the specified pointer/label.
  (Error code 4210)
- The specified program file does not exist.
  (Error code 4210)
- The specified program file cannot be executed.
  (Error code 2411)

Program Example

EFCALL

While X20 is set, the following program executes the subroutine program at pointer address (label) P_0 in the program file "ABC". If X20 is reset, the EFCALL instruction resets the output Y11 as well (1).



7.6.8 XCALL

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
● [1)]	●	●	●	●	●

[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

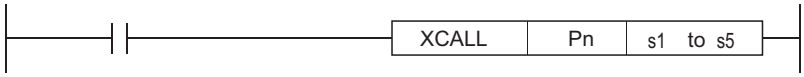
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□N□		Special Function Module U□NG□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				P
pn	—	—	—	—	—	—	—	—	●
s1 – s5	● [1)]	●	●	●	●	●	●	●	—

[1] Annunciators (F) cannot be used.

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
pn	Address number (pointer/label) of subroutine program	Pointer/label
s1 to s5	Device number that passes to subroutine	Bit BIN16-bit BIN 32-bit

Functions **Subroutine program call**

XCALL Subroutine program call

XCALL instruction executes the subroutine program and performs non-execution processing of the subroutine program.

- Execution of subroutine program:

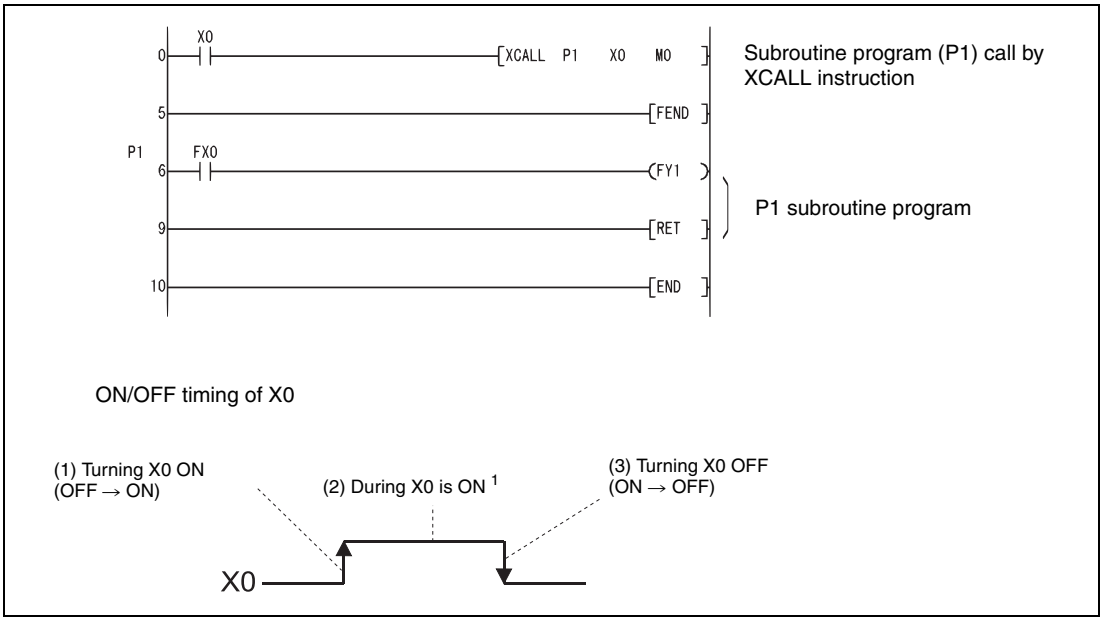
  Executes each coil instruction according to ON/OFF status of the condition contacts.

- Non-execution of subroutine program:

  Performs the same processing for each coil instruction as when the condition contacts are OFF status. The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

Instruction	Condition of contacts and coils
OUT instruction	All contacts and coils, designated by the OUT instruction are reset.
SET instruction	All contacts and coils, designated by these instructions remain their condition.
RST instruction	
SFT instruction	
Basic instructions	
Application instructions	
PLS instruction	All contacts and coils, designated by these instructions adopt a condition as if the execution conditions of the instructions were not set.
Instructions generating an output pulse	
Setting values of low- and high-speed timers	The setting values are reset to 0.
Setting values of retentive timers	The setting values remain set.
Setting values of counters	

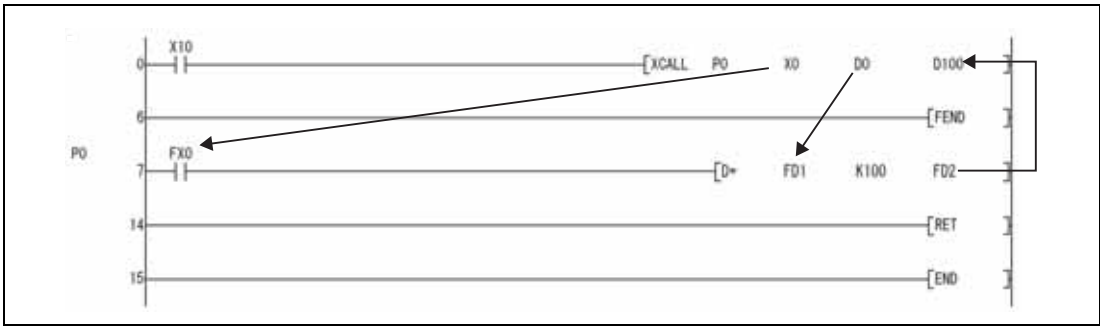
Operation of XCALL instruction varies according to the CPU module type. The following program example shows the operation of XCALL instruction for each CPU module.



1 Time during X0 is ON (2) does not include the time when turning X0 ON (1).

CPU module	Operation of XCALL instruction
• Process CPU (serial No. of first 5 digits : 07031 or lower) • High performance model QCPU (serial No. of first 5 digits: 06081 or lower)	1 When X0 is turned ON: Without process (Do not execute subroutine program of "P1".) 2 During X0 is ON: Execute subroutine program of "P1". 3 When X0 is turned OFF: Perform "Non-execution processing" of subroutine program of "P1".
• High performance model QCPU (serial No. of first 5 digits: 06082 or higher) • Process CPU (serial No. of first 5 digits : 07032 or higher)	1 Using SM734 (XCALL instruction executing condition designation) to select operation when X0 is turned ON. •When SM734 is OFF: Without process (Do not execute subroutine program of "P1".) •When SM734 is ON: Execute subroutine program of "P1". 2 During X0 is ON: Execute subroutine program of "P1". 3 When X0 is turned OFF: Perform "Non-execution processing" of subroutine program of "P1".
• Redundant CPU • Basic model QCPU • Universal model QCPU • LCPU	1 When X0 is turned ON: Execute subroutine program of "P1". 2 During X0 is ON: Execute subroutine program of "P1". 3 When X0 is turned OFF: Perform "Non-execution processing" of subroutine program of "P1".

When function devices (FX, FY, FD) are used by a subroutine program, specify a device with s1 through s5 corresponding to the function device. The contents to the devices specified by s1 to s5 to are as indicated below.



Prior to execution of the sub-routine program, bit data is transmitted to FX, and word data is transmitted to FD.

After the execution of the subroutine, the contents of FY and FD are transmitted to the corresponding device.

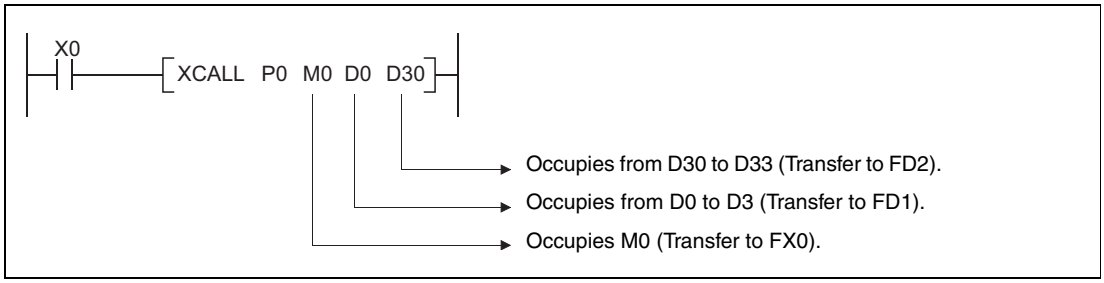
The processing units for the function devices are as follows:

- FX, FY: Bits
- FD : 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function Devices	Device	Data Size	Remark
FX FY	Bit device	1 point	—
	When bit designation is made for word device	1 bit	
FD	When digit designation of a bit device is used [1)]	4 words	The data size varies depending on the instruction to be used.
	Word device	4 words	

[1] An error will not occur even when the device number specified by s1 to s5 is not a multiple of 16 at the digit designation of the bit device.

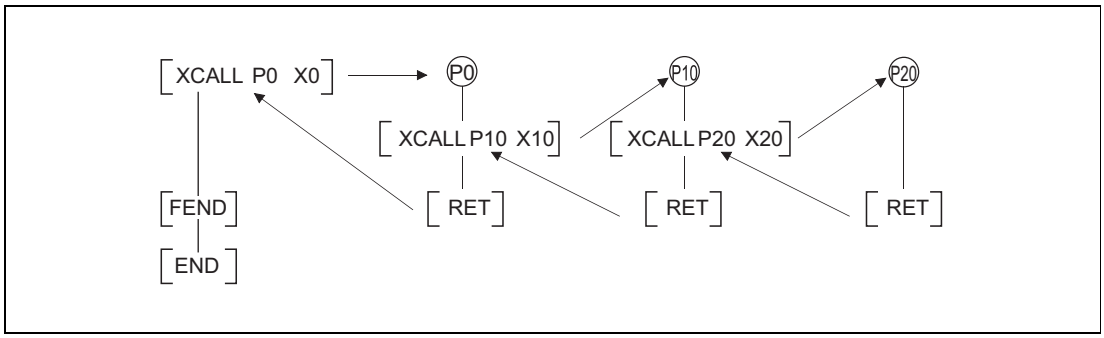


s1 to s5 can be used with the XCALL instruction.

The number of function devices used by sub-routine programs must be identical to the number of devices handed over by the XCALL instruction in s1 through s5. The function devices must be identical to the types of devices handed over by the XCALL instruction.

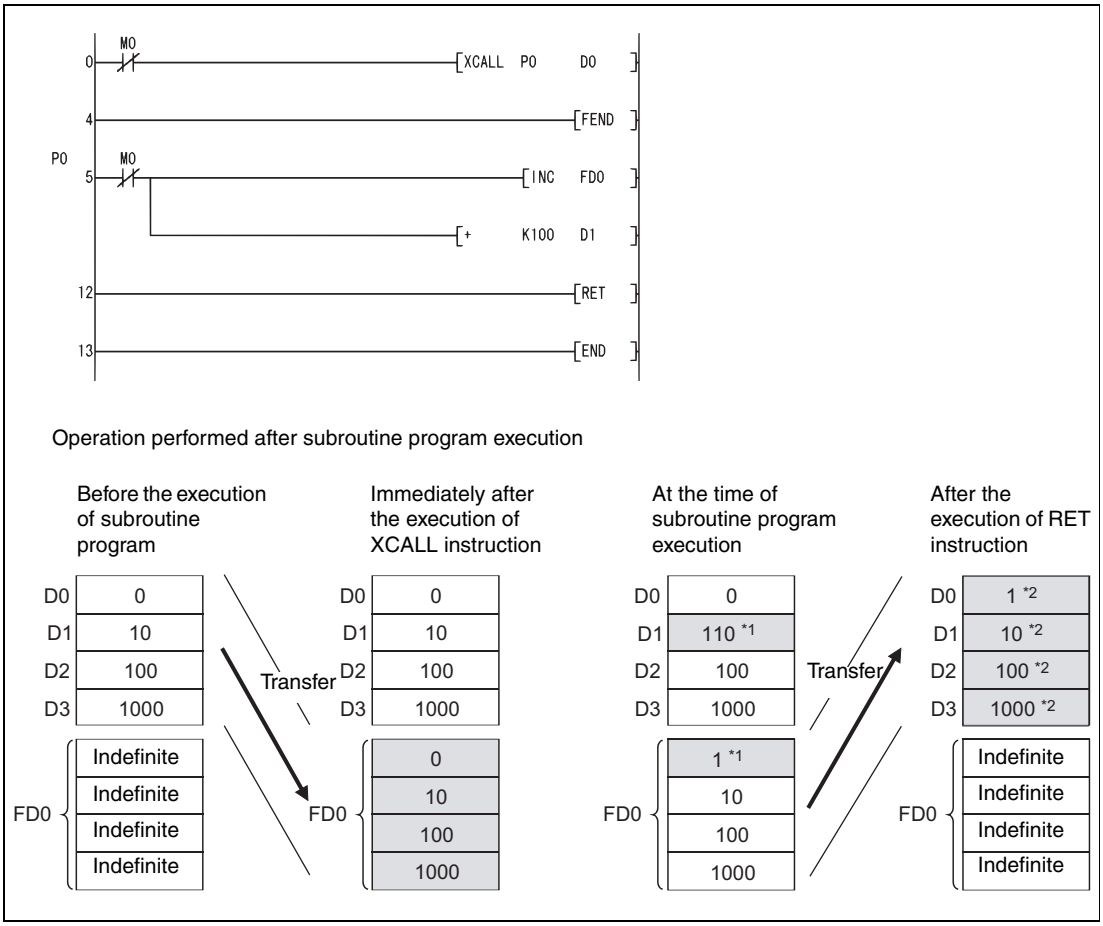
Device numbers specified in the argument of the XCALL instruction should not overlap. If they do overlap, it will not be possible to obtain accurate calculations.

In total up to 16 nesting levels can be programmed. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P) and XCALL instructions.



The device used for the argument of the XCALL instruction must not be used in a subroutine program. If used, it will not be possible to perform correct calculations. (Refer to the following program example.)

The processing to be executed when D1 is used in a subroutine program with D0 designated for FD0 in a subroutine program is shown below.



[1] Stores the execution result of the subroutine program.

[2] Replaced by the value of the function device.

Operation Errors

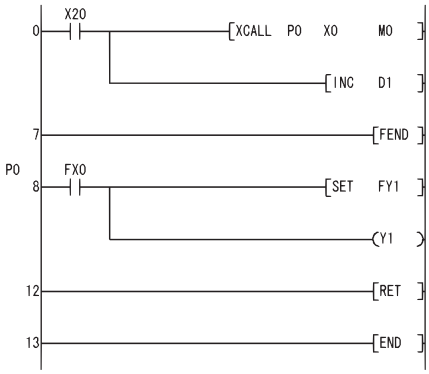
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified for the argument cannot be secured for the data size. (Error code 4101)
- After execution of an XCALL instruction an END, FEND, GOEND, or STOP instruction is executed, without a prior RET instruction. (Error code 4211)
- An RET instruction is executed before an XCALL instruction. (Error code 4212)
- More than 16 nesting levels are executed. (Error code 4213)
- There is no subroutine program stored at the specified pointer/label. (Error code 4210)

Program Example

XCALL

The following program executes a subroutine program with argument when X20 is turned ON.

MELSEC Instruction List	Ladder Diagram
IEC Instruction List	

7.6.9 COM (Refresh)

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
● [1]	● [2]	● [3]	[4]	[4]	[4]

[1] Basic model QCPU of serial No. 04121 or lower

[2] High Performance model QCPU of serial No. 04011 or lower

[3] Process CPU of serial No. 07031 or lower

[4] Refer to next section 7.6.10 for the COM instruction of these modules

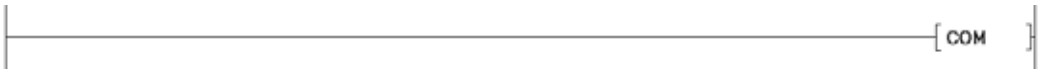
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

GX Works2



Variables

Set Data	Meaning	Data Type
—	—	—

Functions **Refresh instruction**

COM Refresh instruction for link and interface data

A COM instruction is used to speed up data communication with a remote I/O station. If the scan time of a master station is longer than that of a local station, a COM instruction enables correct processing of received input and output data.

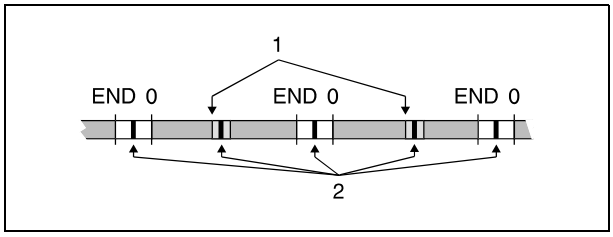
The executed function of the COM instruction depends on the operation condition of the special relay SM775:

- If SM775 is not set (0):
  Performs auto refresh and communication with a peripheral device. [1, 2)]
- If SM775 is set (1): Performs communication with peripheral device only. [1)]

[1] The following processing is performed in communication with peripheral device:
- Monitor processing of other stations
- Read processing by the serial communications module of the buffer memory of another intelligent function module

[2] The auto refresh includes the following processing:
- Refresh of MELSECNET/H
- CC-Link refresh
- Auto refresh of intelligent function modules.

On execution of a COM instruction the CPU temporarily interrupts the sequence program, performs general data processing (END processing), as well as auto refresh of intelligent function modules (including link refreshes). However, the low speed cyclic refresh of MELSECNET/H is not performed.



[1] COM instruction

[2] General data processing/ auto refresh (including link refresh) of intelligent function module

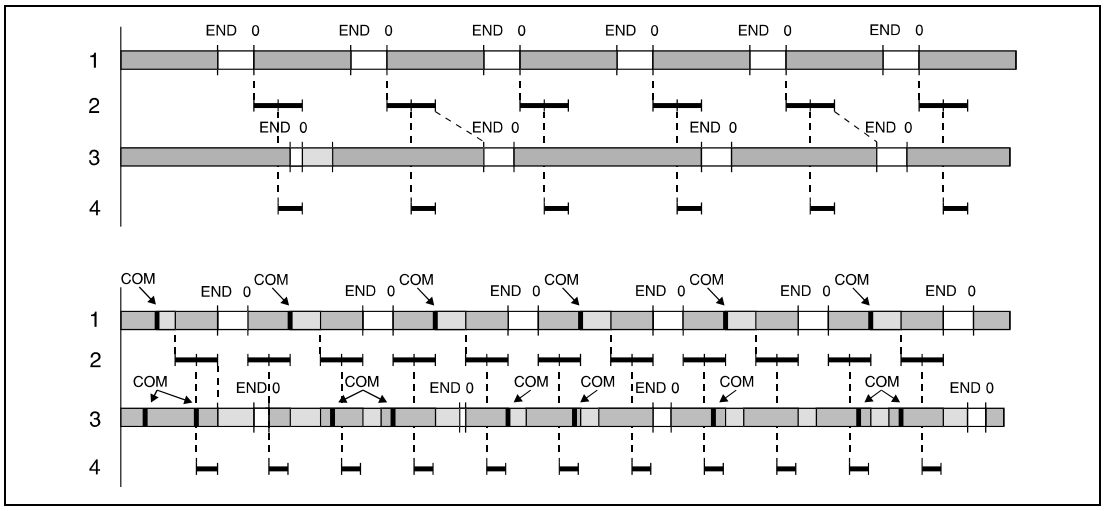
A COM instruction may be used any number of times in the sequence program. In this respect, note that the sequence program scan time is increased by the time taken for communication with peripheral device and the auto refresh (including the link refresh) of the intelligent function modules.

NOTE *The COM instruction cannot be used in the following programs:*

- *Low-speed execution type programs*
- *Interrupt programs*
- *Fixed scan execution type programs*

Data communication using the COM instruction

The upper diagram shows data communication events without a COM instruction.
The lower diagram shows data communication events using a COM instruction.



[1] Master station program

[2] Data communication

[3] Local station program

[4] Remote I/O station, I/O refresh

Data communication between links is speeded up in the sequence program of the master station via the COM instruction, because the number of communication events with the remote I/O station increases.

Data may not be received properly as shown above, if the scan time of the local station sequence program is longer than that of the master station. In this case, secure data communication is achieved with the COM instruction applied in the sequence program of the local station.

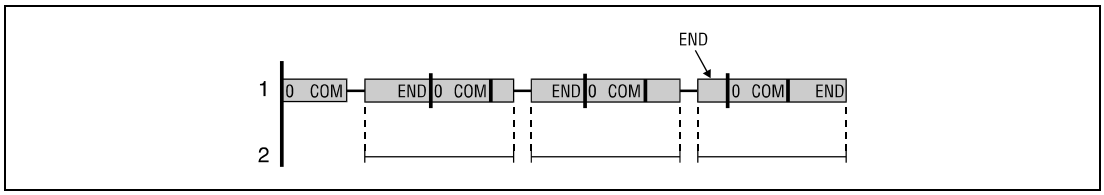
If a COM instruction is programmed in the sequence program of a local station, a link refresh is performed every time the local station receives the master station command between the following instructions:

– Step 0 and COM instruction

– COM instruction and COM instruction

– COM instruction and END instruction

If the link scan time of the link is longer than the sequence program scan time of the master station, data communication cannot be speeded up even if a COM instruction was programmed in the master station.



[1] Sequence program of the master station

[2] Link scan time in the slave station

7.6.10 COM (Selective Refresh)

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
● 1)	● 2)	● 3)	●	●	●

1 Basic model QCPU: The first five digits of the serial No. are "04122" or higher.

2 High Performance model QCPU: The first five digits of the serial No. are "04012" or higher.

3 Process CPU: The first five digits of the serial No. are "07032" or higher.

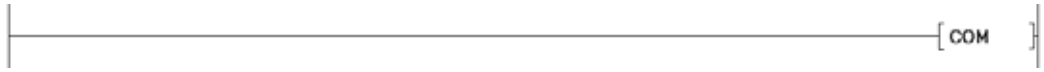
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



COM__GE1

Variables

Set Data	Meaning	Data Type
—	—	—

Functions

Select refresh instruction

COM Select refresh instruction

When the COM instruction is executed, the following refresh operations can be performed.

Refresh items	QCPU	LCPU
I/O refresh	●	●
CC-Link refresh	●	●
CC-Link IE controller network refresh	●	—
CC-Link IE field network refresh	●	—
MELSECNET/H refresh	●	—
Auto refresh of intelligent function modules	●	●
Auto refresh using QCPU standard area of multiple CPU system	●	—
Reading input/output data of all modules other than the multiple CPU system group	●	—
Auto refresh using the multiple CPU high speed transmission area of multiple CPU system	●	—
Communication with display unit	—	●
Service process (communication with peripheral device)	●	●

NOTE *The following processing is performed in communication with peripheral device.*

- *Monitor processing of other station*
- *Read of another intelligent function module buffer memory by the serial communication module*

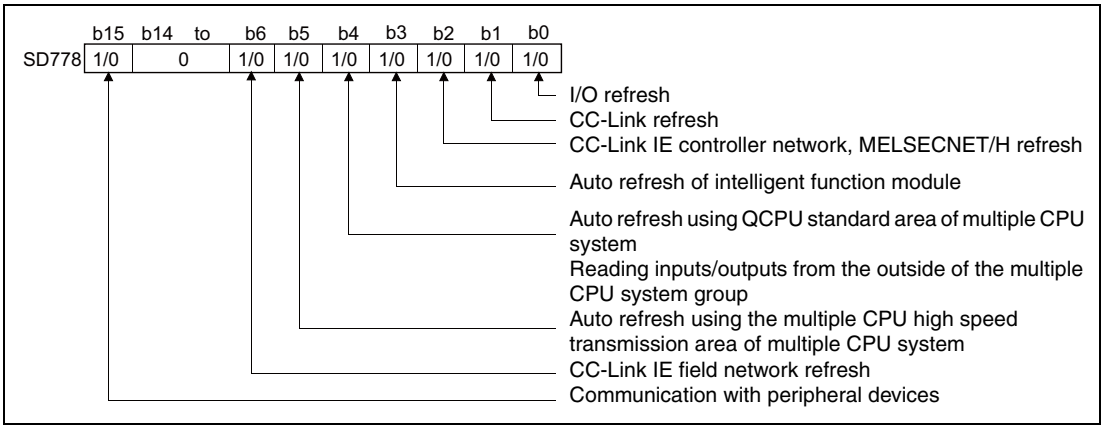
Select refresh items using SD778 and SM775

With SM775 ON, whether to execute a refresh or not can be designated by each bit of SD778 as shown below:

● For QCPU

Bit of SD778	Executed	Not executed
b0 to b6	1	0
b15	0	1

Example (see figure below):
To make only the send/receive processing with the remote I/O station faster, designate MELSECNET/H refresh only. Set only bits b2 and b15 of SD778 to 1 (SD778: 8004H).



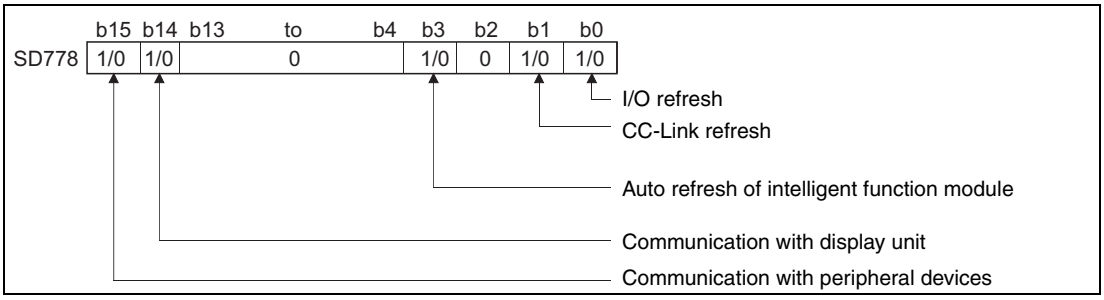
NOTE *Refresh between the multiple CPUs by the COM instruction is performed under the following condition.*

- *Receiving operation from other CPUs : When b4 of SD778 (auto refresh in the CPU shared memory) is 1.*
- *Sending operation from host CPU : When b15 of SD778 (communication with peripheral device is executed/not executed) is 0.*

● For LCPU

Bit of SD778	Executed	Not executed
b0 to b3, b14	1	0
b15	0	1

Example (see figure below):
To speed up processing of the display unit only, specify communication with the display unit only. Write "1" to bits b14 and b15 of SD778 (SD778: C000H).



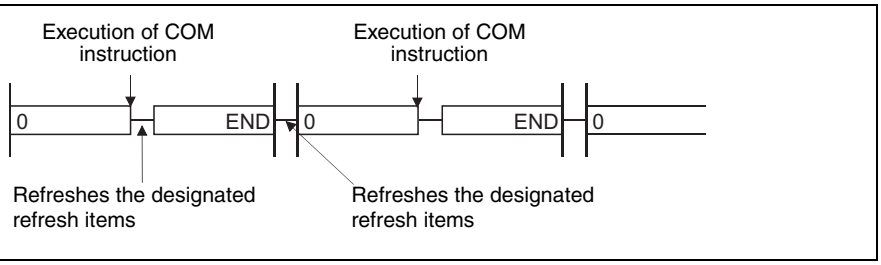
● Turning OFF SM775 refreshes all refresh items except I/O refresh.

● With SM775 turned to ON, select refresh items by SD778.

The following table shows the refresh items that can be designated by turning SM775 ON/OFF and with SD778.

Refresh items	QCPU		LCPU	
	When SM775 is OFF	When SM775 is ON	When SM775 is OFF	When SM775 is ON
I/O refresh	Not executed	Execution/ non-execution selectable	Not executed	Execution/ non-execution selectable
CC-Link refresh	Executed		Executed	
CC-Link IE controller network refresh			—	—
CC-Link IE field network refresh			—	—
MELSECNET/H refresh			—	—
Auto refresh of intelligent function modules			Executed	Execution/ non-execution selectable
Auto refresh using QCPU standard area of multiple CPU system			—	—
Reading input/output data of all modules other than the multiple CPU system group			—	—
Auto refresh using the multiple CPU high speed transmission area of multiple CPU system			—	—
Communication with display unit	—	—	Executed	Execution/ non-execution selectable
Service process (communication with peripheral device)	Executed	Execution/ non-execution selectable		

Upon the execution of the COM instruction, the CPU module suspends the processing of the sequence program, and refreshes the designated refresh item.



A COM instruction may be used any number of times in the sequence program. However, note that the sequence program scan time will be lengthened by the time taken for refresh time of the communication with peripheral devices and refresh item that are selected in SD778.

Only with the Universal model QCPU and LCPU, interruption is enabled during the execution of the COM instruction. However, note that the data can be separated if the refresh data is used by an interrupt program etc.

With the Built-in Ethernet port QCPU and LCPU, processing time may be increased if the service process was executed by the COM instruction while the built-in Ethernet ports are in Ethernet connection.

NOTES

The COM instruction cannot be used in the following programs:

- *Low-speed execution type programs*
- *Interrupt programs*
- *Fixed scan execution type programs*

For the redundant CPU, there are restrictions on use of the COM instruction. Refer to the following manual for details: QnPRHCPU User's Manual (Redundant System).

7.6.11 CCOM, CCOMP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

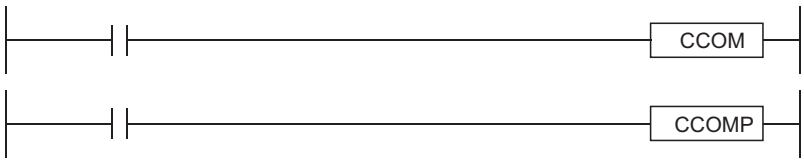
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
—	—	—

Functions

Select refresh instruction

CCOM Refresh instruction for link and interface data

Refer to section 7.6.10 for function details.

Operation Errors

When the CCOM(P) instruction is executed in the QnUD(H)CPU whose serial number (first five digits) is "10101" or lower, an error occurs.
(Error code 4100)

Program Example

CCOMP

Turning on M0 enables the program to execute the select refresh, while turning off M0 disables the program to execute the select refresh.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

7.6.12 IX, IXEND

CPU

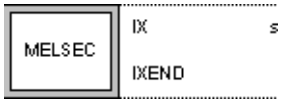
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●		

Devices

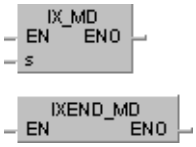
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—

GX IEC Developer

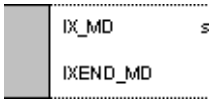
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

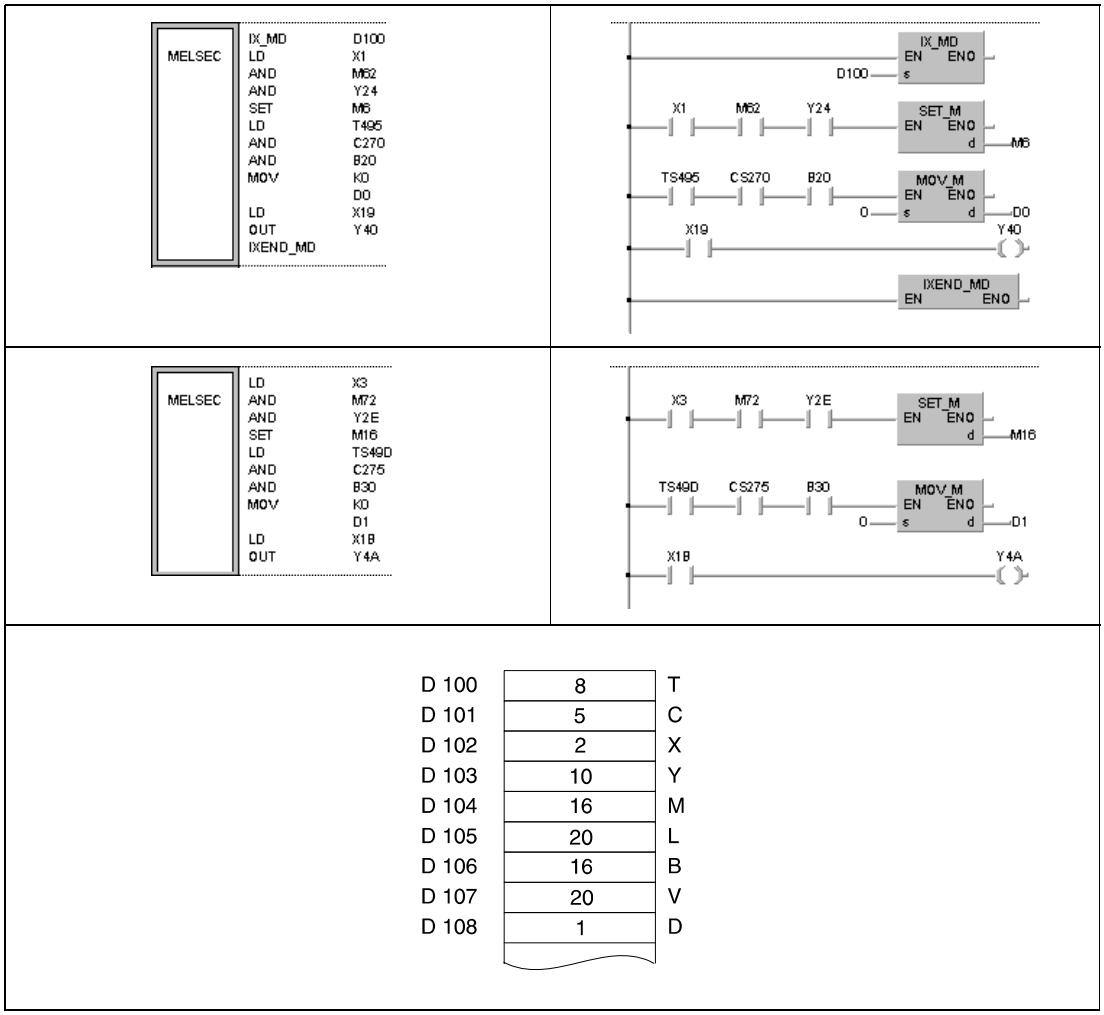
Set Data	Meaning	Data Type
s	First number of device storing data for index qualification	BIN 16-bit

Functions **Index qualification of entire program parts**

IX, IXEND Index qualification instruction

The instructions IX and IXEND are supported only in MELSEC mode in the GX IEC Developer. The IX and IXEND instructions perform index qualification on those devices in the program part located between the IX and IXEND instructions.

On index qualification, decimal values from an index table (s) are added to the device numbers. This new address in hexadecimal format becomes the valid address for further processing. Each device specified in s is assigned a specific type of device, on which the addition is applied. The following diagrams illustrate index qualification:



D 100	8	T
D 101	5	C
D 102	2	X
D 103	10	Y
D 104	16	M
D 105	20	L
D 106	16	B
D 107	20	V
D 108	1	D

The value in D100 (8) is added to the timer address TS495. The new address is TS49D.

The value in D101 (5) is added to the counter address CS270. The new address is CS275.

The value in D102 (2) is added to the addresses of the inputs X1 and X19. The new addresses are X3 and X1B.

The value in D103 (10) is added to the addresses of the outputs Y24 and Y40. The new addresses are Y2E and Y4A.

The value in D104 (16) is added to the addresses of the internal relays M6 and M62. The new addresses are M16 and M72.

The value in D106 (16) is added to the address of the link relay B20. The new address is B30.

The value in D108 (1) is added to the register address D0. The new address is D1.

PLS, PLF, and pulsed instructions that are executed once only on set input condition, cannot be addressed by index qualification via the IX/IXEND instruction

In cases where the new address, resulted from the addition exceeds the relevant address range, the instruction cannot be processed accurately.

If the IX and IXEND instructions are executed during a change between program sequences in the online mode (modifying in RUN mode) the instruction cannot be processed neither.

The values added to the addresses of word devices of which each bit can be accessed are stored as binary data. The initial addresses of the devices these values are specified for are stored in s.

In a program, between the IX and the IXEND instruction no index qualification can be performed.

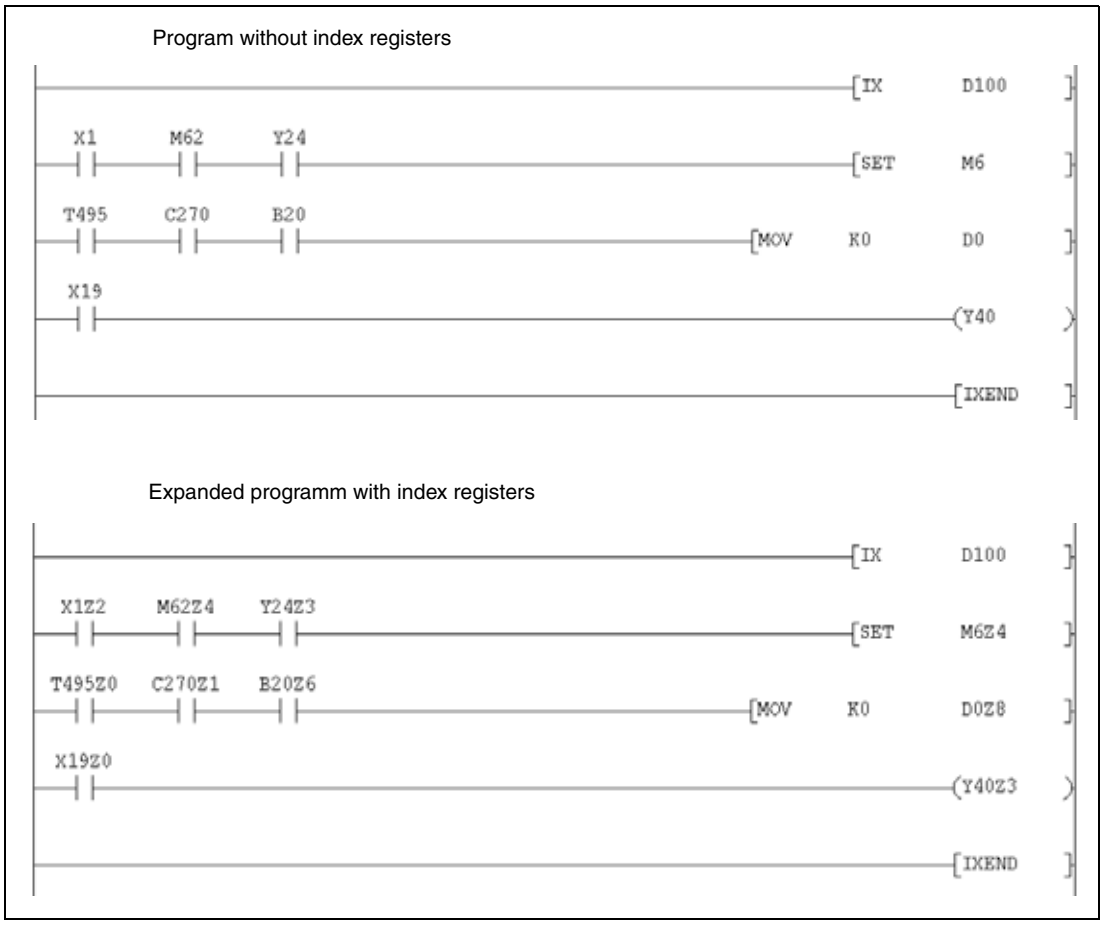
When a program is expanded, the indexed addresses of devices in a program part located between the IX and the IXEND instruction are transformed to addresses using index registers (Zn). The assignment of indexed addresses to the corresponding index registers is shown below:

s	Device	Index Register	s	Device	Index Register
s	Qualification value of timer (T)	Z0	s+8	Qualification value of data register (D)	Z8
s+1	Qualification value of counter (C)	Z1	s+9	Qualification value of link register (W)	Z9
s+2	Qualification value of input (X)	Z2	s+10	Qualification value of file register (R)	Z10
s+3	Qualification value of output (Y)	Z3	s+11	Qualification value of buffer register I/O (U)	Z11
s+4	Qualification value of internal relay (M)	Z4	s+12	Qualification value of buffer register (G)	Z12
s+5	Qualification value of latch relay (L)	Z5	s+13	Qualification value of network numbers of link devices with direct access (J)	Z13
s+6	Qualification value of link relay (B)	Z6	s+14	Qualification value of file register (ZR)	Z14
s+7	Qualification value of edge relay (V)	Z7	s+15	Qualification value of pointer (label)	Z15

The index registers Z10 to Z15 are not available for the Q00JCPU, Q00CPU, and Q01CPU.

Depending on the programming software used the user has to add the index registers in the sequence program between the IX and the IXEND instructions manually.

Example GX Works2



The index registers used between the IX and the IXEND instructions (Z0 to Z15) do not affect the index registers used by other instructions elsewhere in the program.

NOTES

When using the IX and IXEND instructions in both a normal sequence program and an interrupt sequence program, establish an interlock to avoid simultaneous execution. The interlock assumes the area between the IX and IXEND instructions in the normal sequence program as DI, disabling the interruption.

The IXDEV and IXSET instructions can be used to specify modification values. Refer to section 7.6.13 for details.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The IX and IXEND instructions are not programmed in conjunction.
  (Error code 4231)
- After execution of the IX instruction an END, FEND, GOEND or STOP instruction is executed before the IXEND instruction is executed.
  (Error code 4231)

Program Example

IX, IXEND

The following program processes the program loop between IX and IXEND for 10 times. With each loop the device numbers programmed within the loop are increased by 1. The table below shows the registers containing the values of the corresponding devices to be added. In addition the changes in the device numbers for the 1st, 2nd, and 10th loop are shown.

MELSEC Instruction List

```
MELSEC
        LD      SM400
        FMOV    K0
                D100
                K7
        FOR     K10
        IX      D100
        LD      B0
        OR      Y30
        ANI     X30
        OUT     Y30
        SET     M0
        +       D0
                D10
        LD      TS3
        AND     CS4
        MOV     K1
                D40
        IXEND
        LD      SM400
        BK+     D100
                K1
                D100
                K7
        NEXT
```

D	Device	Device Number Change / Loop			
		1.	2.	3.	10.
D100	Qualification value of timer (T)	T3	T4	T5	TC
D101	Qualification value of counter (C)	C4	C5	C6	CD
D102	Qualification value of input (X)	X10	X11	X12	X19
D103	Qualification value of output (Y)	Y30	Y31	Y32	Y39
D104	Qualification value of internal relay (M)	M0	M1	M2	M9
D106	Qualification value of link relay (B)	B0	B1	B2	B9
D108	Qualification values of data registers (D)	D0	D1	D2	D9
		D10	D11	D12	D19
		D40	D41	D42	D49

7.6.13 IXDEV, IXSET

CPU

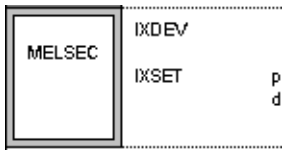
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●		

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				P
p	—	—	—	—	—	—	—	—	●
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List



Ladder Diagram

IEC Instruction List

GX Works2

Variables

Set Data	Meaning	Data Type
p	First number of device (pointer/label only) storing data for index qualification	Pointer/label
d	First number of device storing indexed addresses of devices	BIN 16-bit

Functions **Storing indexed device numbers in an index qualification table**

IXDEV/IXSET Instruction for writing to an index table

The instructions IXDEV and IXSET are supported in the GX Works2 or in MELSEC mode in the GX IEC Developer only.

The IXDEV and IXSET instructions read the addresses of the devices in the offset designation area and write these offset numbers to an index table in the device designated by d.

Refer to the instructions IX and IXEND (section 7.6.12) for the assignment of device types to their corresponding registers.

If a device type is not assigned in the offset designation the value 0 is stored in the index table.

The single bits of word devices are processed as dummy contact, i.e. only the address of a single bit can be read and written to the intex table. In order to address the dummy the corresponding bit is specified. Bit 0 (b0) in data register D0 is addressed D0.0. For bit designation in a 16-bit data word the hexadecimal values 0 through F are used.

Reading in the offset values applies as follows:

- Reading in the devices: T□, C□, X□, Y□, M□, L□, V□, B□

  The offset value indicated □ is read in and written to the corresponding registers.

- Reading the devices: D□.XX, W□.XX, R□.XX[1)], U□\G□.XX[1)], ZR□.XX[1)]

  The offset value indicated □ is read in and written to the corresponding registers.
  The value indicated XX serves as variable for the bit designation.

  [1] Not possible for Q00JCPU, Q00CPU, and Q01CPU

- Reading in the devices: J□/B□[1)], J□/W□[1)], J□/X□[1)], J□/Y□[1)]

  The offset value indicated □ is read in and written to the corresponding registers.
  If no offset value is to be written for the device following J□/, this value is to be set to 0.

  [1] Not possible for Q00JCPU, Q00CPU, and Q01CPU

- On programming the IXSET instruction the offset value of the device P□ is designated directly via address (pointer/label).

If in the offset designation area two identical device types are specified, the offset value of the latter device is valid.

The IXDEV and IXSET instructions have to be programmed in conjunction.

The offset value of the device ZR□.XX may range from 0 to 32767. The offset value is the remainder of the quotient of the device number divided by 32767, and is written to the corresponding register.

For the dummy contacts in the offset designation area only LD and AND instructions are valid. All other instructions are ignored.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

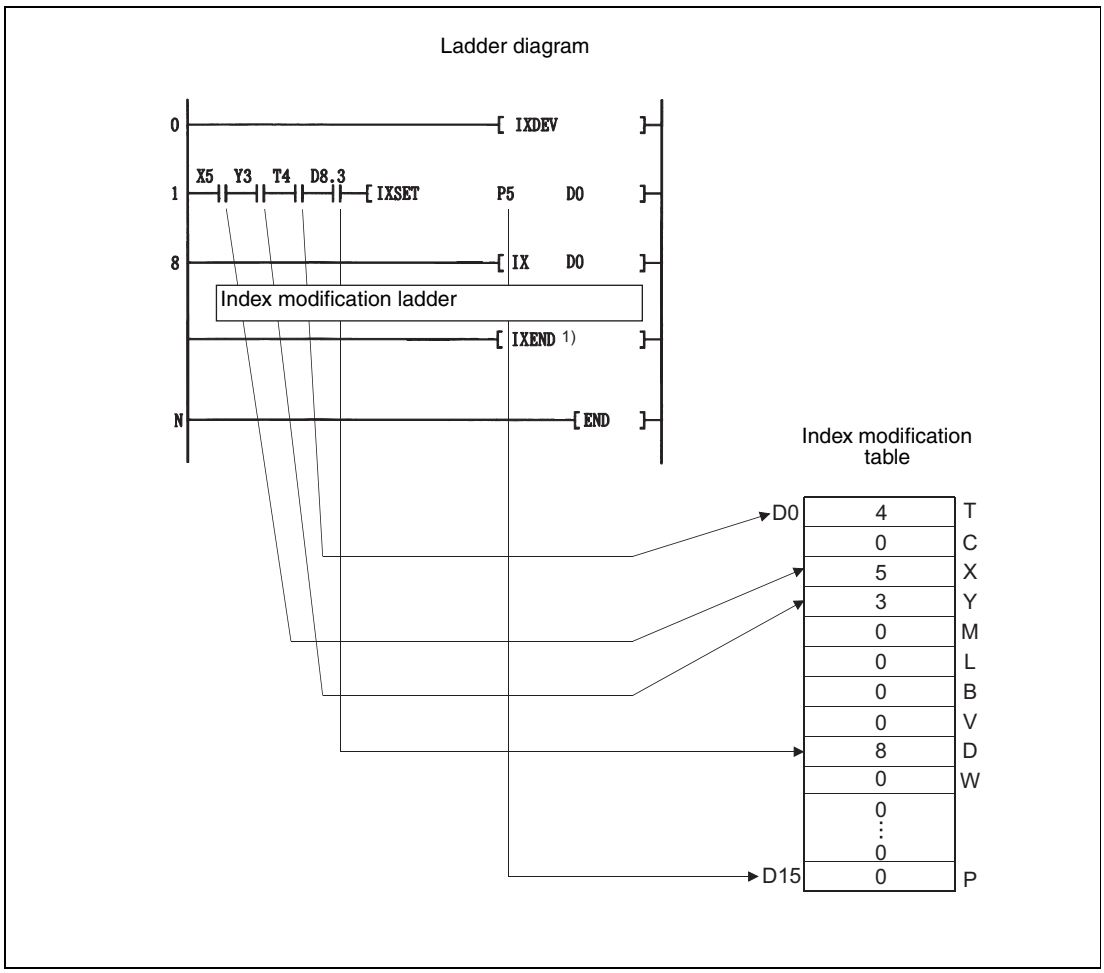
- The IXDEV and IXSET instructions are not programmed in conjunction. (Error code 4231)

Program Example

IXDEV, IXSET

The following program changes the modification values for input (X), output (Y), data register (D) and pointer (P).

When using a basic model QCPU, the devices R, U/G, J, ZR and P cannot be used.



[1] Refer to the instructions IX and IXEND (section 7.6.12) for the assignment of device types to their corresponding registers.

7.7 Data table operation instructions

The operation instructions for data tables write and read data to and from a data table. Current data are written to the table and read out in a different order for further processing. In addition, these instructions enable deleting and inserting specific data blocks.

The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Write data to a data table	FIFW	FIFW_M
	FIFWP	FIFWP_M
Read data entered first from data table	FIFR	FIFR_M
	FIFRP	FIFRP_M
Read data entered last from data table	FPOP	FPOP_M
	FPOPP	FPOPP_M
Delete specified data blocks from data table	FDEL	FDEL_M
	FDELP	FDELP_M
Insert specified data blocks in data table	FINS	FINS_M
	FINSP	FINSP_M

7.7.1 FIFW, FIFWP

CPU

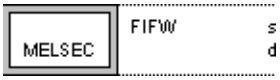
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—

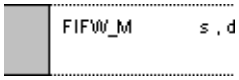
GX IEC Developer

MELSEC Instruction List

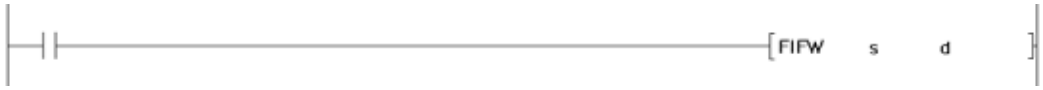


Ladder Diagram

IEC Instruction List



GX Works2



Variables

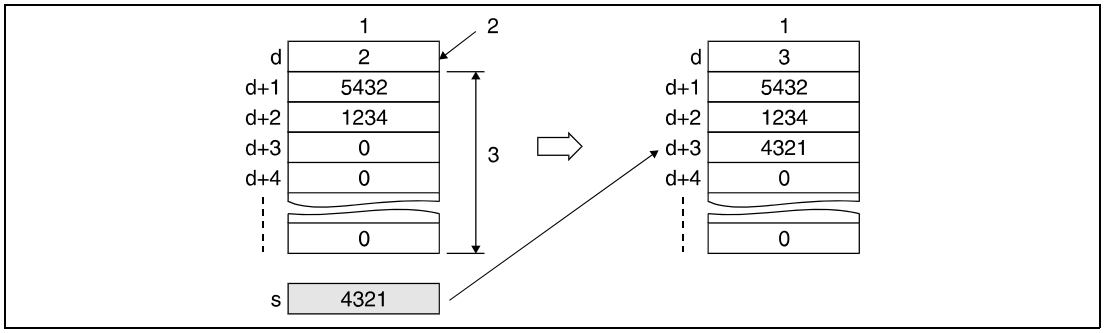
Set Data	Meaning	Data Type
s	Data to be written to the data table or devices storing such data	BIN 16-bit
d	First number of data table	

Functions

Writing data to a data table

FIFW Instruction for data entry

The FIFW instruction writes data in a sequence specified by s to a data table. This table is specified by the address range in d and conducts data in the sequence of their entry. In the first address of the data range in d the total number of data records contained in the table is stored. Therefore, the value at this address is the position pointer for data to be recorded in the table. On each execution of the FIFW instruction this value is increased by 1. Thus, following data are recorded from the address d+1.



[1] Data table

[2] Position pointer

[3] Data table range

Prior to the first FIFW instruction the contents of the device specified in d have to be cleared.

The number of data records to be recorded and the address range of the data table have to be controlled on programming by the user.

Operation Errors

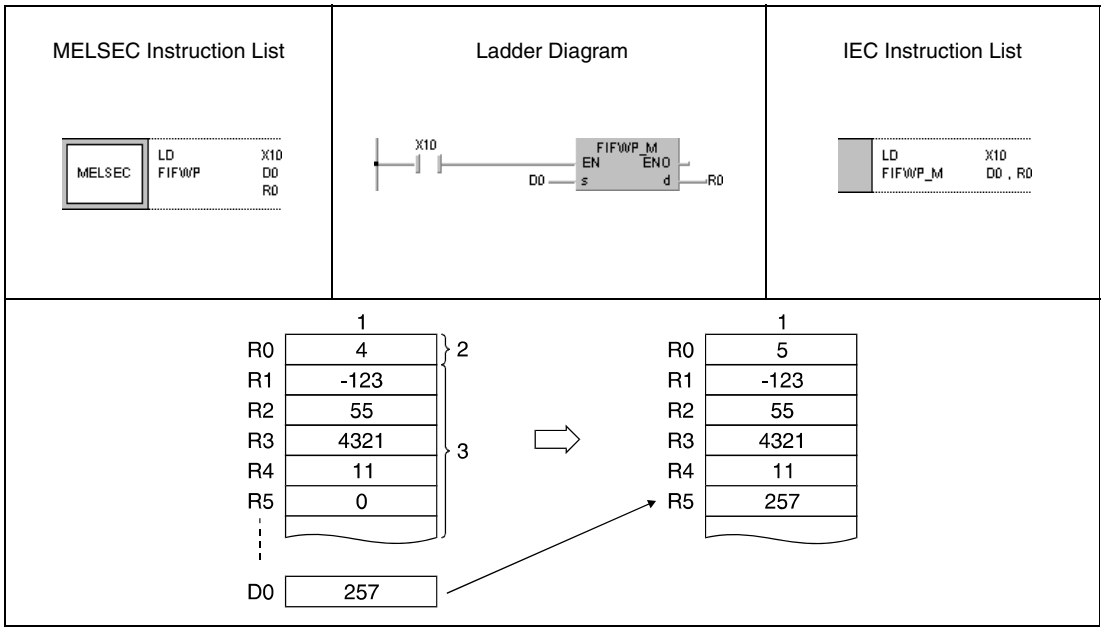
In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data table range of the FIFO table exceeds the relevant storage device range when executing the FIFW instruction.
  (Error code 4101)

Program Example 1

FIFWP

The following program specifies the storage range of the data table via the data registers R0 through R5. The initital address of the storage range (R0) contains the position pointer, indicating the number of stored data records. With leading edge from X10, data in D0 are stored at the next available storage position of the data table (R5).



[1] Data table

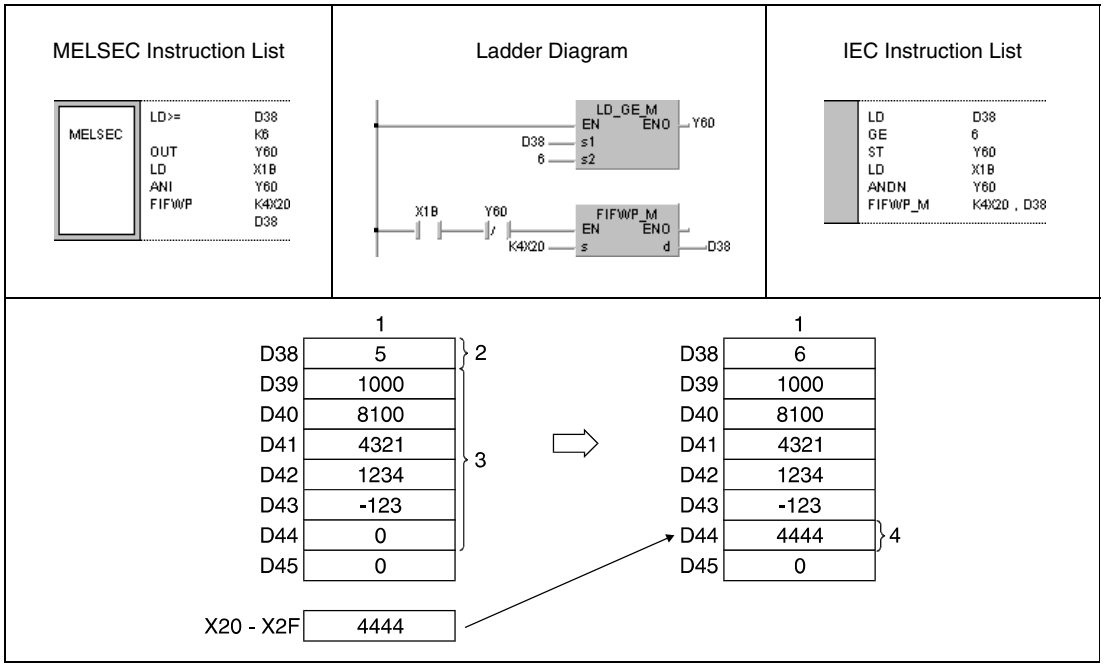
[2] Position pointer

[3] Data table range

Program Example 2

FIFWP

The following program specifies the storage range of the data table via the data registers D38 through D44. The initital address of the storage range (D38) contains the position pointer, indicating the number of stored data records. With leading edge from X1B, data at the inputs X20 through X2F are stored at the next available storage position of the data table (D44). The data table specified here stores at maximum 6 data records. Therefore, Y60 is programmed as a limiter of the FIFW instruction. The output is set, if the contents of D38 are greater than or equal to 6.



[1] Data table

[2] Position pointer

[3] Data table range

[4] Highest available storage address

7.7.2 FIFR, FIFRP

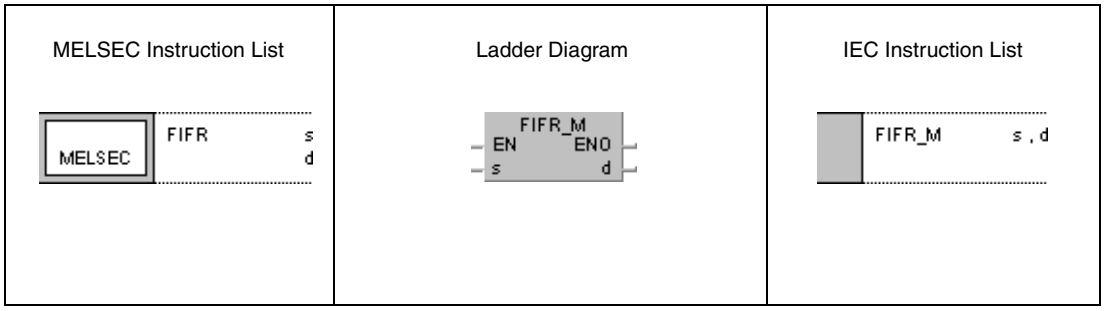
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

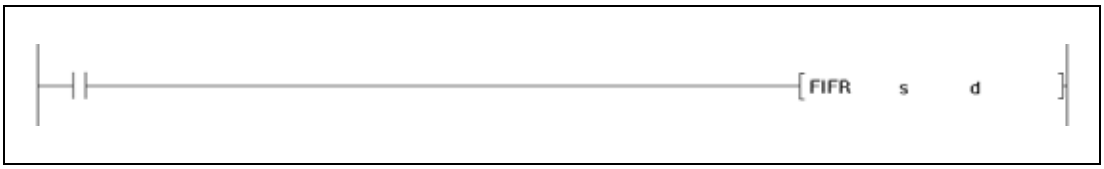
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	—	
d	—	●	●	—	—	—	—	—	—

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
s	First number of device storing read out data	BIN 16-bit
d	First number of data table	

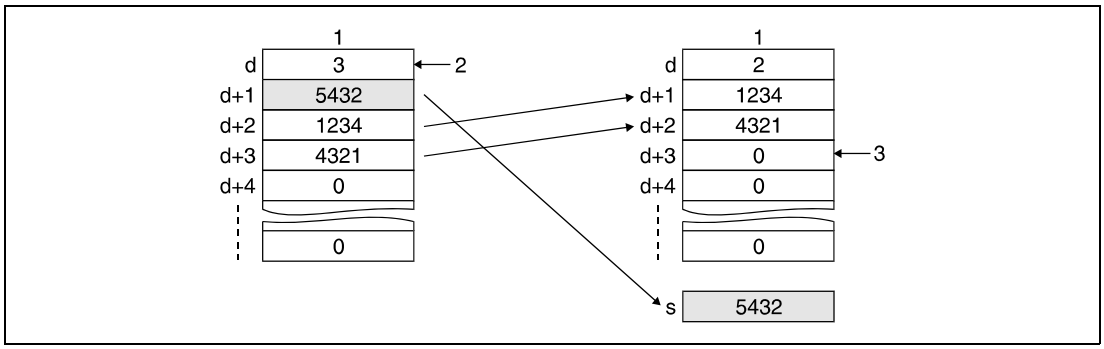
Functions

Reading data entered first from a data table

FIFR Instruction for reading data entered first

The FIFR instruction reads data from a data table and stores them in a specified storage range. Reading the data begins with the first address d+1 after the position pointer. The data is transferred to the storage range specified by s.

The data in the data table are moved successively to the beginning of the table in order of their entry. All preceding data are cleared. After reading out, the value of the position pointer (first address in d) is decreased by 1.



[1] Data table

[2] Position pointer

[3] This register is reset to 0

NOTE *Make sure this instruction is not executed, while d (position pointer) contains the value 0.*

Operation Errors

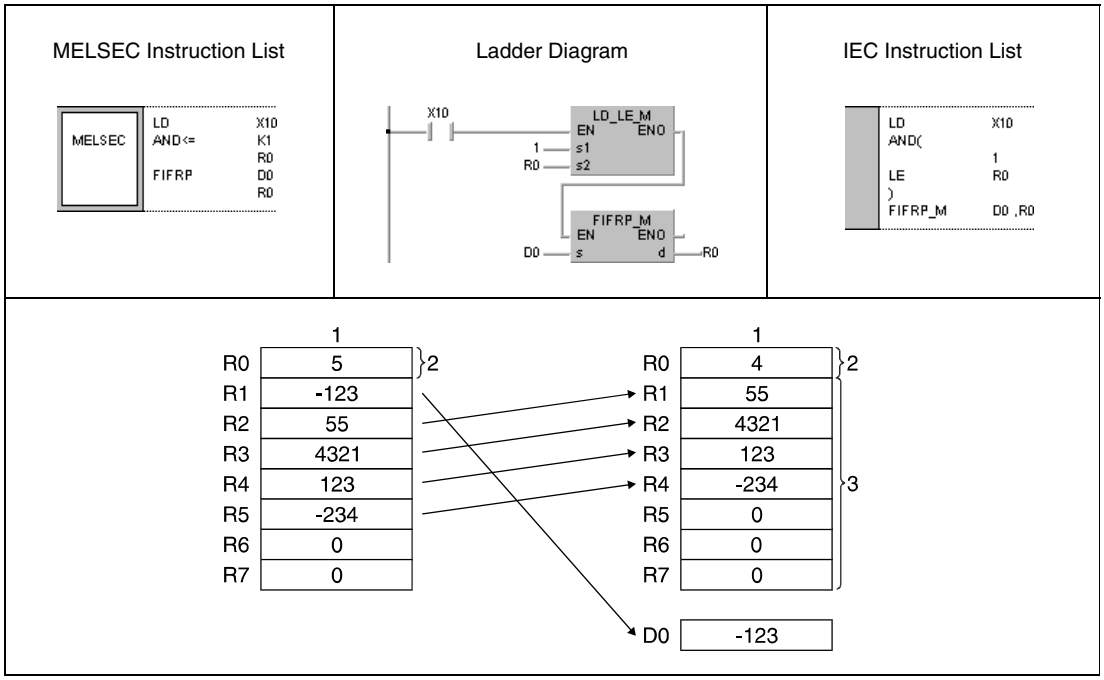
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- An FIFR instruction is executed while the position pointer contains the value 0.
  (Error code 4100)
- The device table range exceeds the corresponding device range when executing the FIFR instruction.
  (Error code 4101)

Program Example 1

FIFRP

With leading edge from X10, the following program reads the data value in R1 (first entered value) of the data table from R0 through R7 and stores the value in the register D0. At the beginning the value of the position pointer is 5 and after the execution 4. The preceding comparison operation avoids the execution of the FIFR instruction, if the position pointer (R0) contains the value 0.



[1] Data table

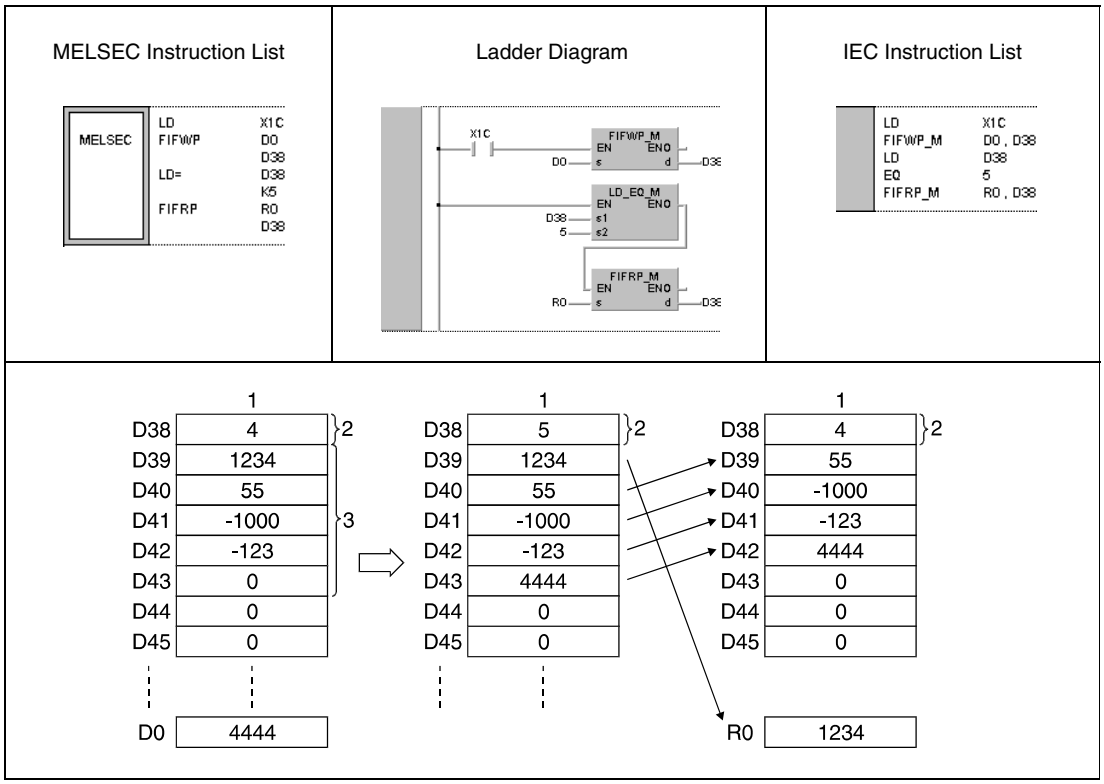
[2] Position pointer

[3] Data table range

Program Example 2

FIFRP

With leading edge from X1C, the following program writes a value from D0 to the data table from D38 through D43. If the value of the position pointer is 5, the first value of the FIFO table is read and passed on to R0. This process is repeated with every leading edge from X1C.



[1] Data table

[2] Position pointer

[3] Data table range

7.7.3 FPOP, FPOPP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

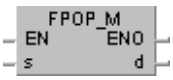
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	—	—
d	—	●	●	—	—	—	—	—	—

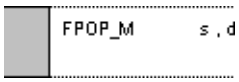
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

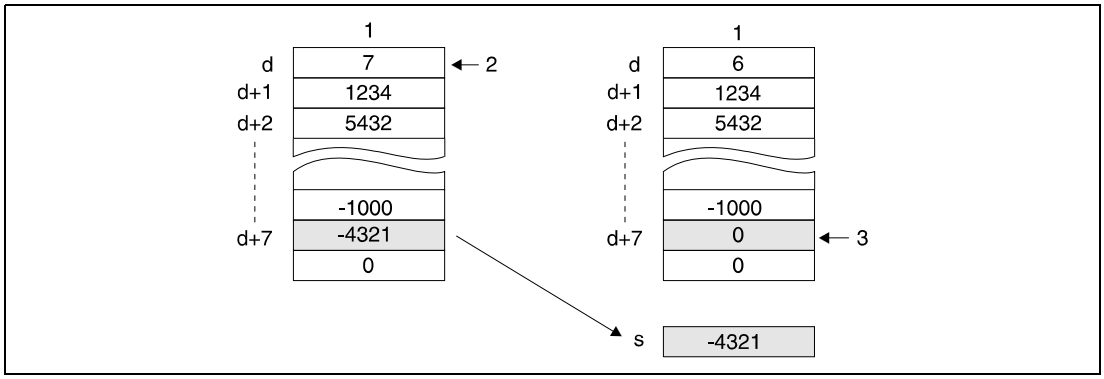
Set Data	Meaning	Data Type
s	First number of device storing read data	BIN 16-bit
d	First number of data table	

Functions **Reading data entered last from a data table**

FPOP Instruction for reading data entered last

The FPOP instruction reads data from a data table and stores them in a specified storage range. Reading the data begins with the last address d+n in the data table. The data is transferred to the storage range specified by s.

The read address in the data table is reset to 0. After reading out, the value of the position pointer (first address in d) is decreased by 1.



[1] Data table

[2] Position pointer

[3] This register is reset to 0

NOTE *Make sure this instruction is not executed, while d (position pointer) contains the value 0.*

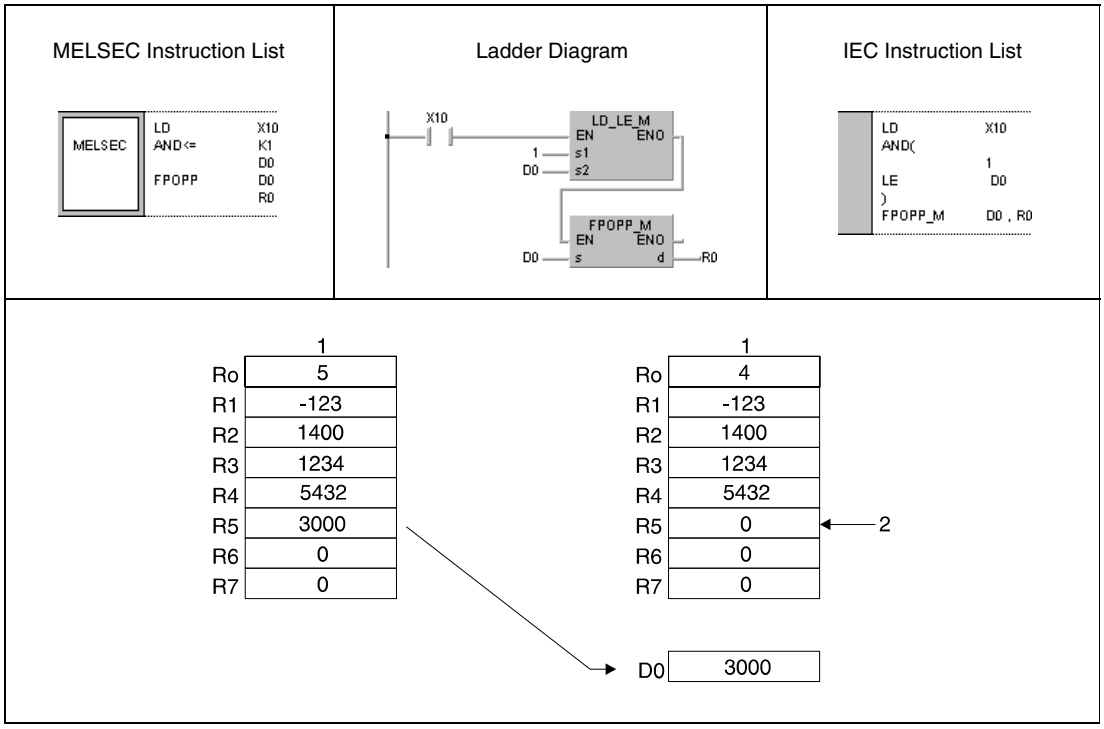
Operation Errors In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- An FPOP instruction is executed while the position pointer contains the value 0. (Error code 4100)
- The data table range exceeds the corresponding device range when executing the FPOP instruction. (Error code 4101)

Program Example 1

FPOPP

With leading edge from X10, the following program reads the data value in R5 (value entered last) of the data table from R0 through R7 and stores the value in the register D0. At the beginning the value of the position pointer is 5 and after the execution 4. The preceding comparison operation avoids the execution of the FPOPP instruction, if the position pointer (R0) contains the value 0.



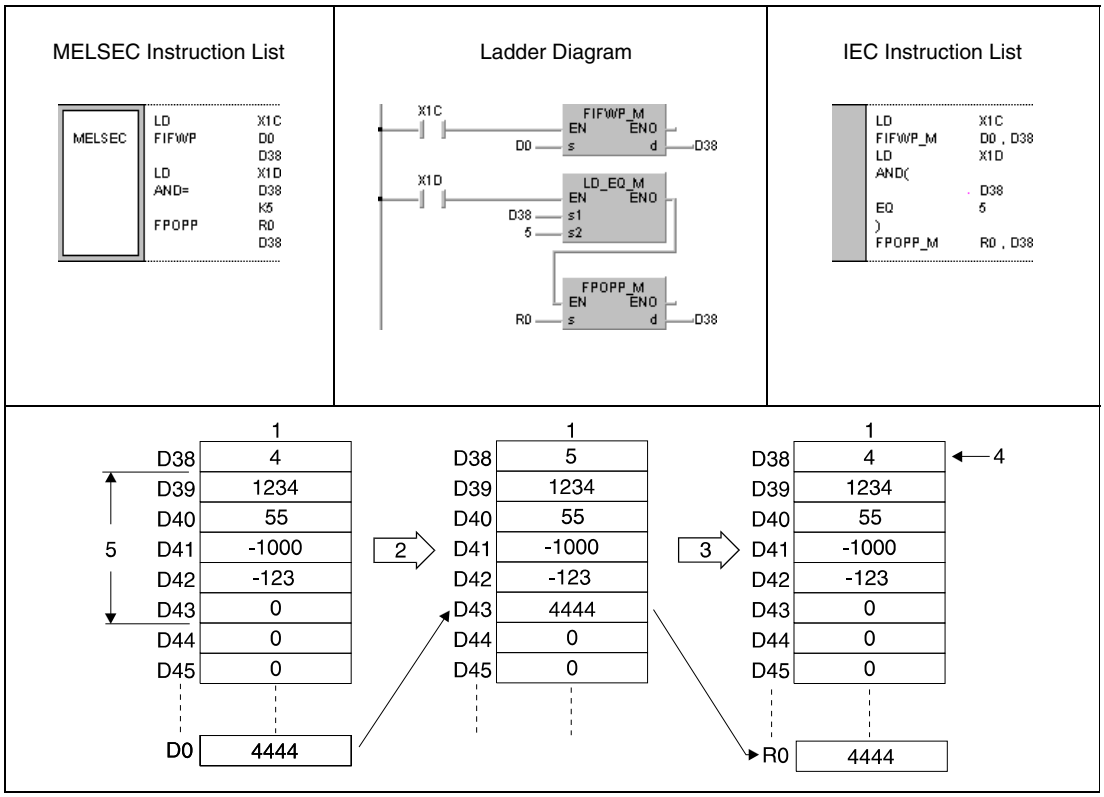
[1] Data table

[2] This register is reset to 0

Program Example 2

FPOPP

With leading edge from X1C, the following program writes a value from D0 to the data table from D38 through D43. If the value of the position pointer is 5, with leading edge from X1D the value in register D43 is read and passed on to R0.



[1] Data table

[2] Leading edge from X1C

[3] Leading edge from X1D

[4] Position pointer

[5] Current address range of data table

7.7.4 FDEL, FDELP, FINS, FINSP

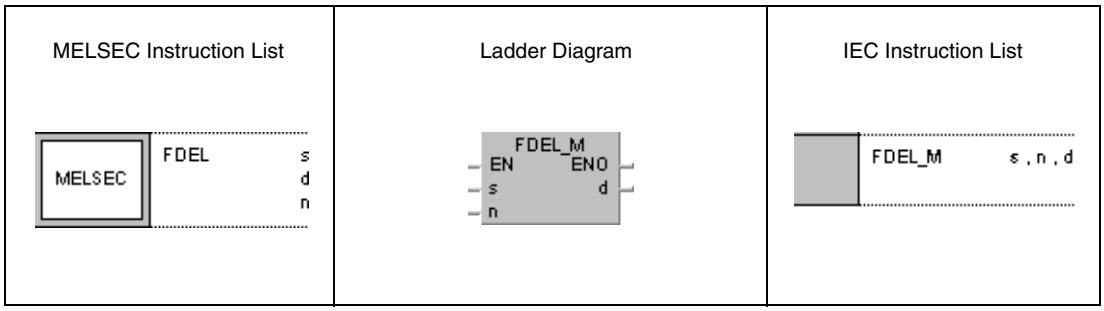
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

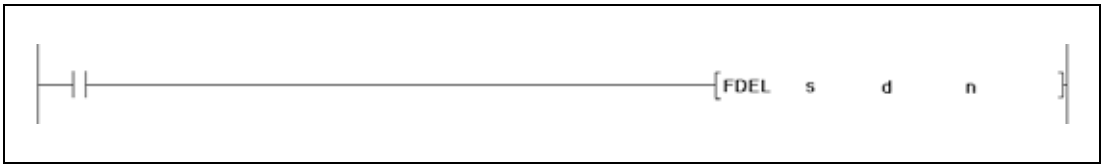
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	—	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer



GX Works2



Variables

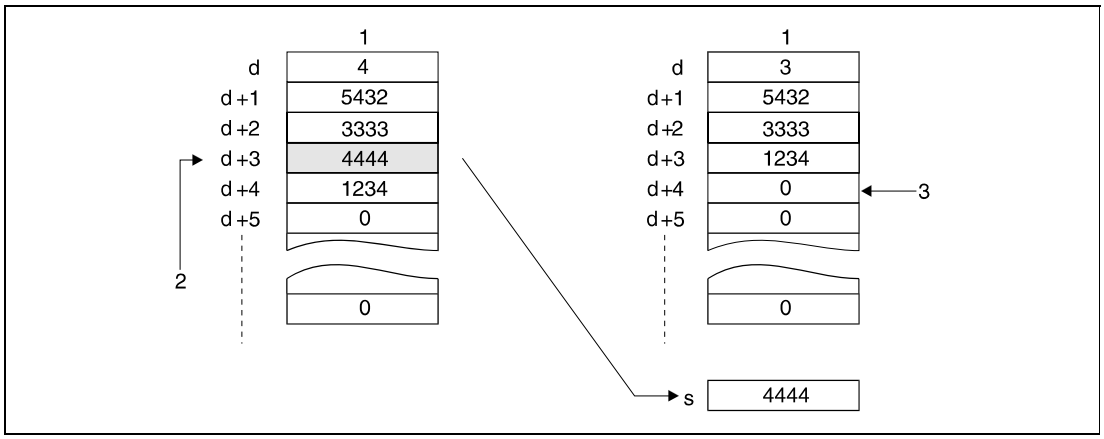
Set Data	Meaning	Data Type
s	Data to be inserted into the data table at a specified address or device storing such data. First number of device storing data to be deleted from a data table at a specified address.	BIN 16-bit
d	First number of data table	
n	Number of address where data is to be inserted or deleted	

Functions **Deleting and inserting specified data blocks in a data table**

FDEL Deleting specified data blocks

The FDEL instruction deletes the nth data block after the postion pointer from a data table specified by d and stores this value in a device specified in s.

The data in the data table are shifted together after deletion of one data block. After reading, the value of the position pointer (first address in d) is decreased by 1.



[1] Data table

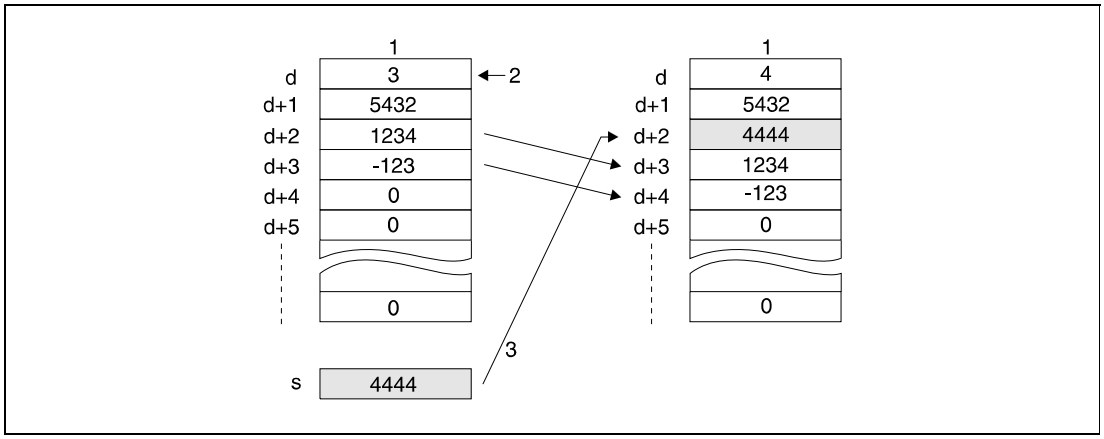
[2] For n=3 the data block d+3 is deleted.

[3] This register is reset to 0

FINS/FINSP Inserting specified data blocks

The FINS instruction inserts a 16-bit data block specified by s at the nth position after the position pointer into the data table specified by d.

The data blocks following the inserting position are shifted on by one address. After inserting, the value of the position pointer (first address in d) is increased by 1.



[1] Data table

[2] Position pointer

[3] For n=2 the data block is inserted at d+2

Operation Errors

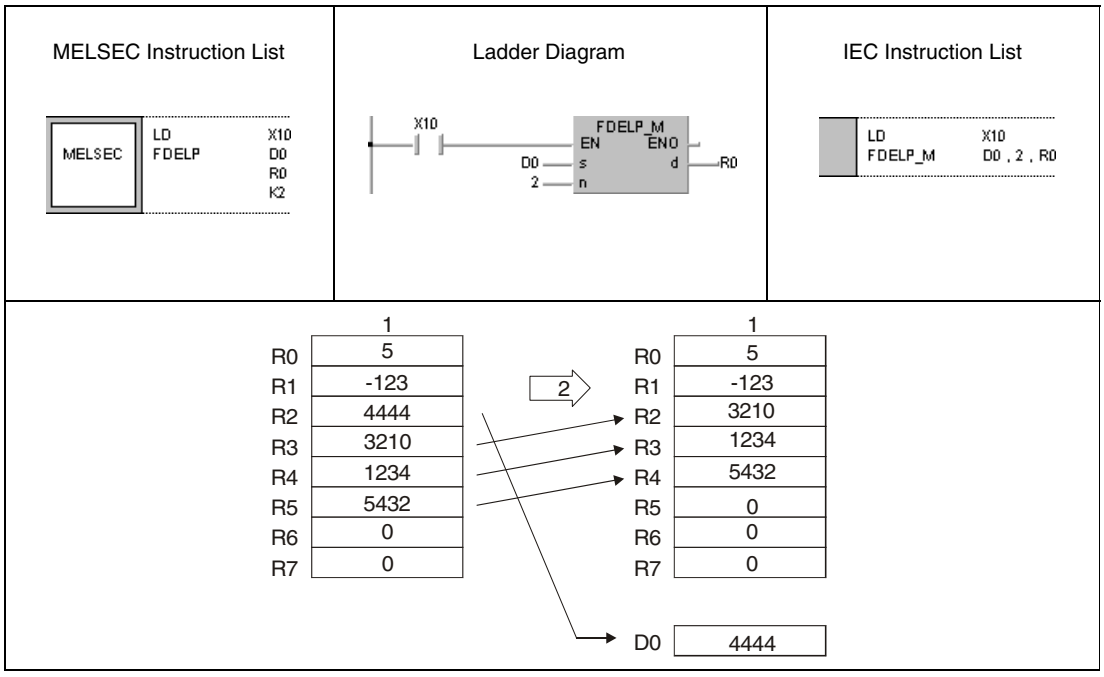
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The nth position from d is larger than the data storage number at the execution of the FDEL instruction.
  (Error code 4101)
- The inserting position in d specified by n via the FINS instruction exceeds the address range of existing data blocks plus 1.
  (Error code 4101)
- The value of n exceeds the device range of the table d.
  (Error code 4101)
- The FDEL or FINS instruction was executed when n = 0.
  (Error code 4100)
- The FDEL was executed when the value of d was 0.
  (Error code 4100)
- The data table range exceeds the corresponding device range when the FDEL or FINS instruction is executed.
  (Error code 4101)

Program Example 1

FDELP

When X10 goes ON, the data from the 2nd position (R2) of the data table ranging from R0 to R7 will be deleted and the data stored in D0.



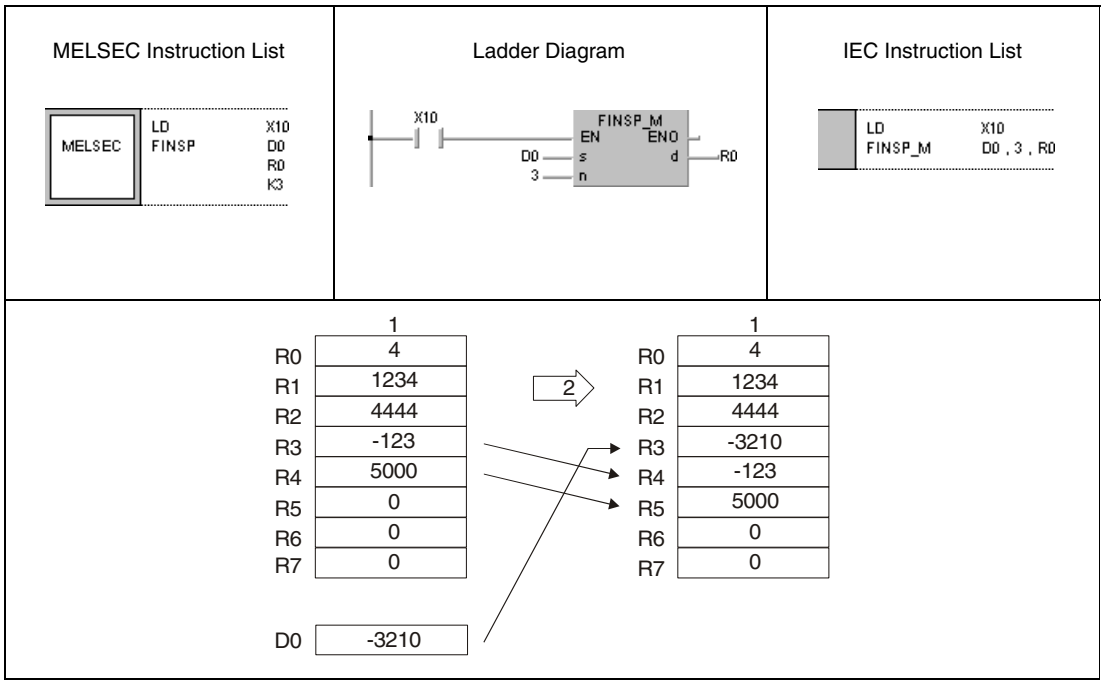
[1] Data table

[2] Leading edge of X10

Program Example 2

FINSP

The following program inserts the data at D0 at the 3rd position of the data table ranging from R0 to R7 when X10 goes ON.



[1] Data table

[2] Leading edge of X10

7.8 Buffer memory access instructions

The following instructions access the buffer memory of special function modules. These instructions enable the CPU to exchange data with the according modules.

The following table gives an overview of the instructions:

Function	MELSEC instruction in MELSEC Editor	MELSEC instruction in IEC Editor
Reading data from a special function module	FROM	FROM_M
	FROMP	FROMP_M
	DFRO	DFRO_M
	DFROP	DFROP_M
Writing data to a special function module	TO	TO_M
	TOP	TOP_M
	DTO	DTO_M
	DTOP	DTOP_M

7.8.1 FROM, FROMP, DFRO, DFROP

CPU

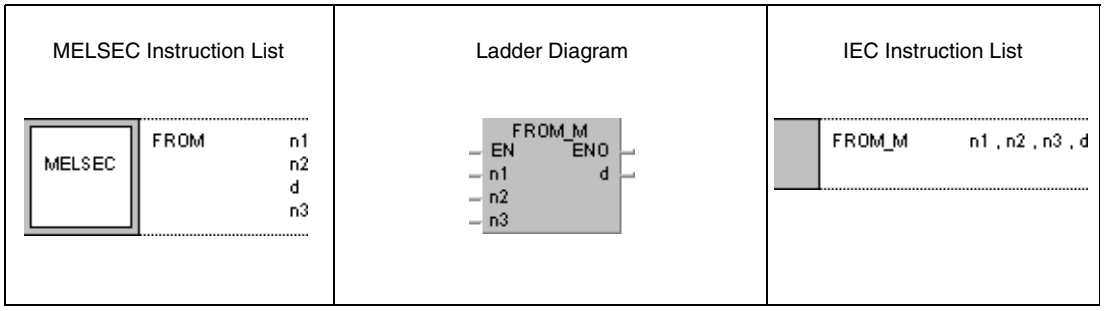
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	● [1)]	●

[1] Other than Q00UJCPU

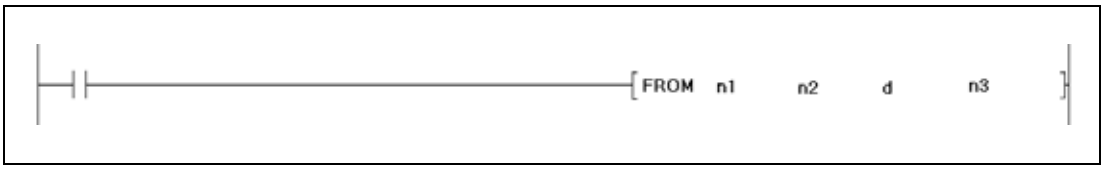
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				U
n1	●	●	●	●	●	●	●	●	●
n2	●	●	●	●	●	●	●	●	—
d	●	●	●	—	—	—	—	—	—
n3	●	●	●	●	●	●	●	●	—

GX IEC Developer



GX Works2



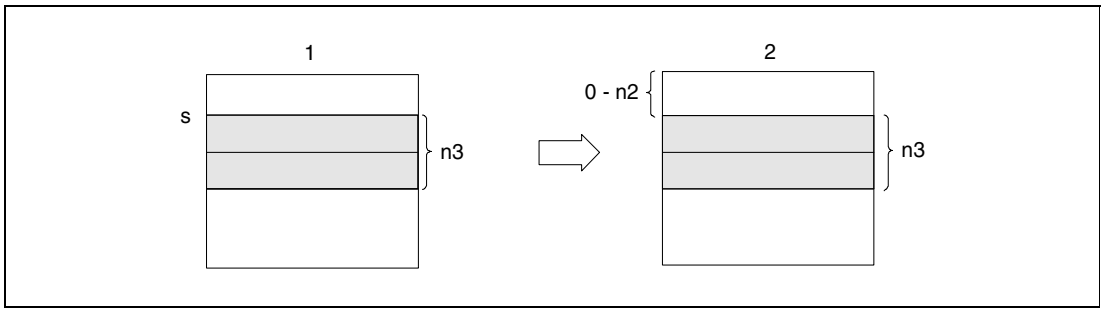
Variables

Set Data	Meaning	Data Type
n1	Head address of special function module on base unit	BIN 16-bit
n2	First number of memory address area for data to be read	BIN 16-bit
d	First number of memory address area of the CPU to be written to	BIN 16/32-bit
n3	Number of data words to be read	BIN 16-bit

Functions **Reading 1-word and 2-word data from a special function module**

FROM Reading 1-word data (16-bit)

The FROM instruction reads 1-word data from the buffer memory of a special function module and stores it in a specified memory address area of the CPU. The first address of data to be read is specified by n2, the number of data words is specified by n3, and the head address of the special function module, resulting from the position of the module on the base unit is specified by n1. The memory address area of the CPU storing the data is specified by d.



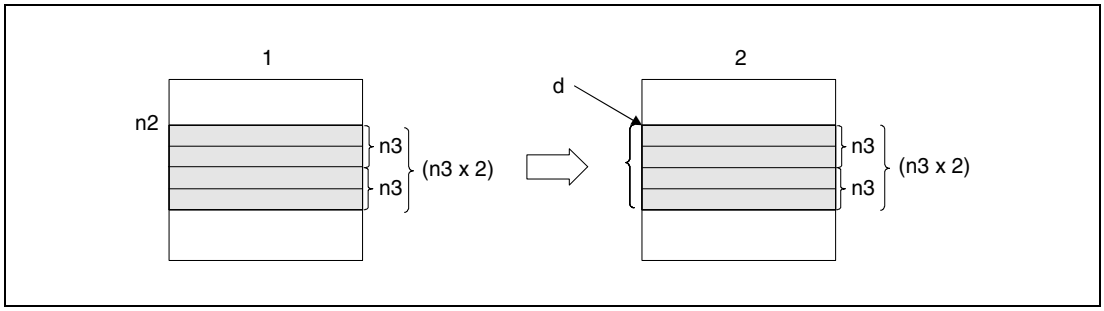
[1] Buffer memory of special function module

[2] Memory of the CPU

NOTE *The FROM instruction can also be used to read data from shared memory of another station in a multi CPU system. Refer to section 9.2.1 for more details.*

DFRO Reading 2-word data (32-bit)

The DFRO instruction reads 2-word data from the buffer memory of a special function module. The first address of data to be read is specified by n2, the number of data words (2-multiple) is specified by n3, and the head address of the special function module is specified by n1. The memory address area of the CPU storing the data is specified by d.



[1] Buffer memory of special function module

[2] Memory of the CPU

NOTE *Data read from special function modules is also possible with the use of a special function module device. In this case the devices are specified as U□\G□ (U (Headadress of the special function module)/G (Buffer memory adress)).*

For the special function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- No signals have been exchanged with the special function module at the execution of the instruction.
  (Error code 1412)
- An error has occured in the special function module at the the execution of the instruction.
  (Error code 1402)
- The I/O number specified by n1 is not a special function module.
  (Error code 2110)
- The number of data words specified in n3 (2 x n3 for DFRO) exceeds the storage range of the device specified by d.
  (Error code 4101)
- The address specified by n2 is outside the buffer memory range.
  (Error code 4101)

Program Example 1

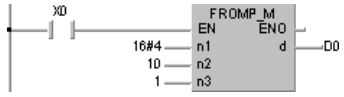
FROMP

With leading edge from X0, the following program reads the digital values of channel CH1 from address 10 of the buffer memory of an Q68ADV module. The memory address area of the module is 040 through 05F. The read data is stored in D0.

MELSEC Instruction List

```
MELSEC  LD    X0
        FROMP H4
              K10
              D0
              K1
```

Ladder Diagram



IEC Instruction List

```
LD   X0
FROMP_M  16#4 , 10 , 1 , D0
```

Program Example 2

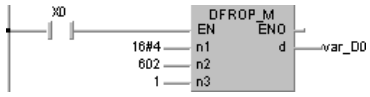
DFROP

With leading edge from X0, the following program reads the x-axis data at the addresses 602 and 603 in the buffer memory of an QD75P4 module. The memory address area of the module is 040 through 05F. The read data is stored in D0 and D1.

MELSEC Instruction List

```
MELSEC  LD    X0
        DFROP H4
              K602
              D0
              K1
```

Ladder Diagram



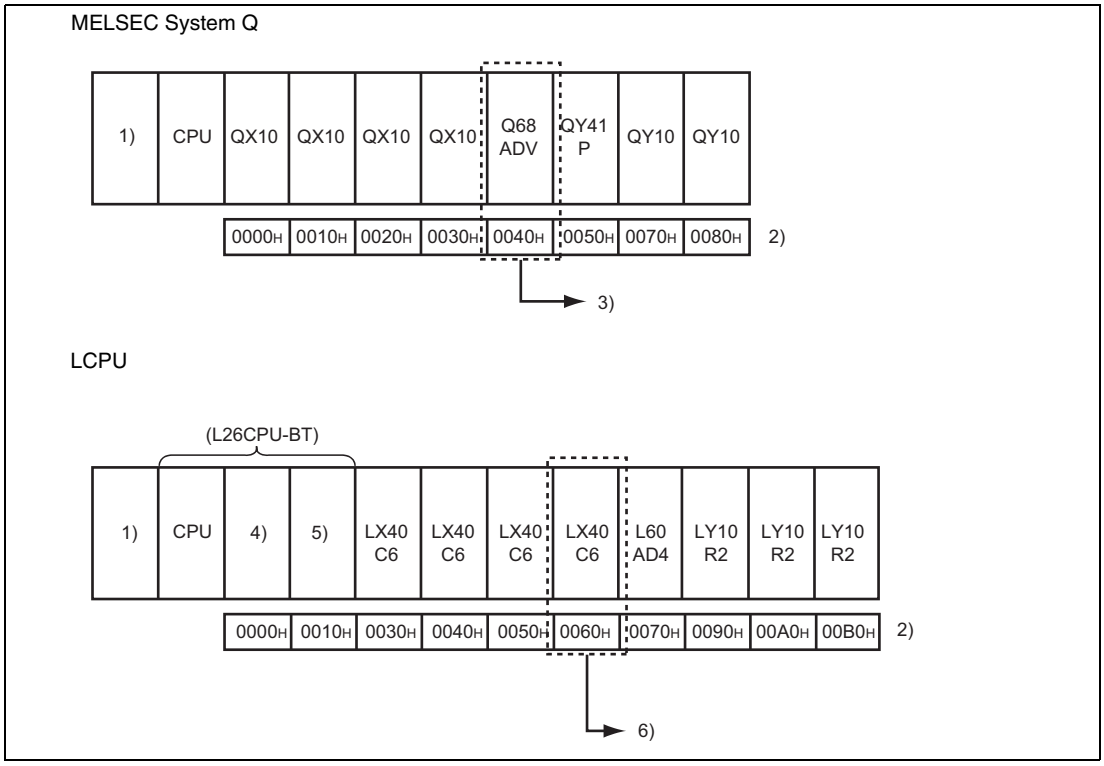
IEC Instruction List

```
LD   X0
DFROP_M  16#4 , 602 , 1 , var_D0
```

NOTE

The program example 2 will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

NOTES *The value of n1 is specified by the upper 3 digits of hexadecimal 4-digit representation of the head I/O number of the slot in which an intelligent function module is mounted.*



[1] Power supply module

[2] Head I/O number configured in the I/O assignment setting

[3] Head address of special function module: 0040H --> n1 = K4 or H4

[4] Built-in I/0

[5] Built-in CC-Link

[6] Head address of special function module: 0060H --> n1 = K6 or H6

QCPU and LCPU establish the automatic interlock of the FROM/DFRO instructions.

7.8.2 TO, TOP, DTO, DTOP

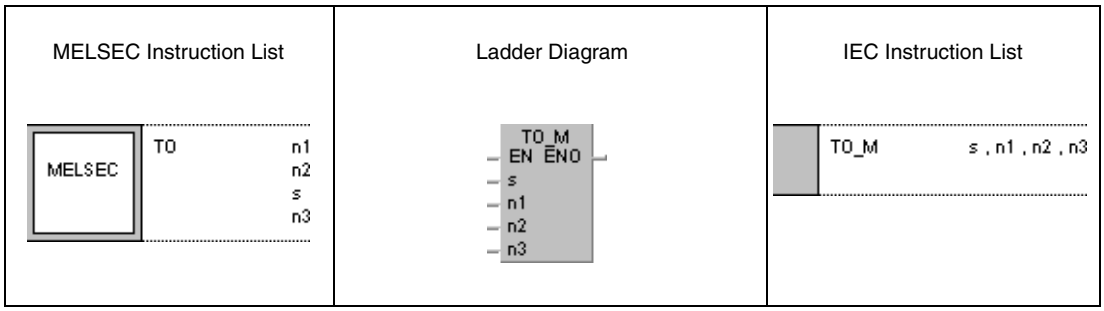
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

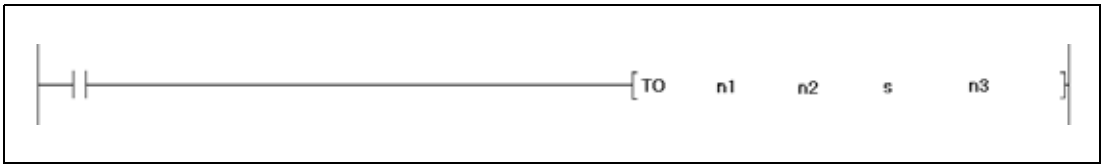
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				U
n1	●	●	●	●	●	●	●	●	●
n2	●	●	●	●	●	●	●	●	—
s	●	●	●	—	—	—	—	●	—
n3	●	●	●	●	●	●	●	●	—

GX IEC Developer



GX Works2



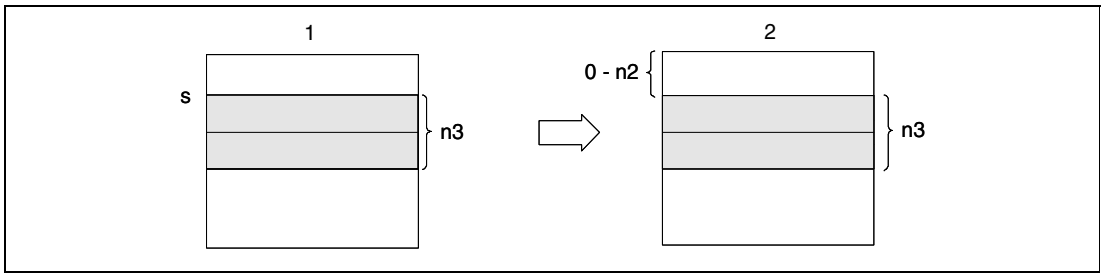
Variables

Set Data	Meaning	Data Type
n1	Head address of special function module on base unit. Specified with the upper three digits when the head I/O number is expressed in 4 hexadecimal digits.	BIN 16-bit
n2	First number of memory address area to be written to	
s	Data to be written or first number of memory address area of the CPU storing data to be written	BIN 16-/32-bit
n3	Number of data words to be written	BIN 16-bit

Functions **Writing 1-word and 2-word data to the buffer memory of a special function module**

TO Writing 1-word data (16-bit)

The TO instruction writes 1-word data from the memory of the CPU to the buffer memory of a special function module. The first address of the memory area data is to be written to is specified by n2, the number of data words is specified by n3, and the address of the special function module, resulting from the position of the module on the base unit is specified by n1. The first address of the memory address area the data is to be read from is specified by s.

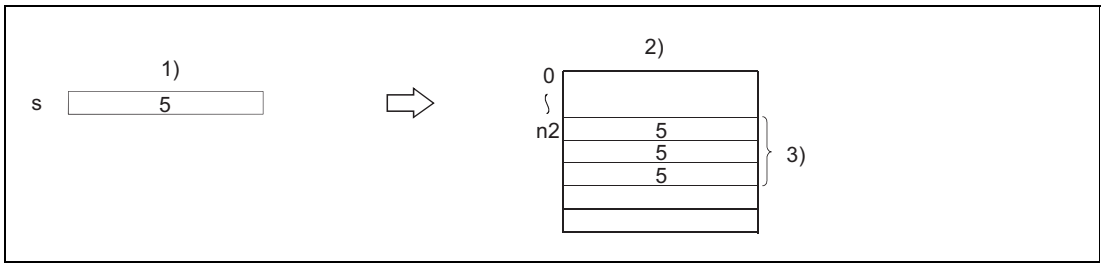


[1] Memory of the CPU

[2] Buffer memory of special function module

When a constant is designated to s, the instruction writes the same data (value designated to s) to the area of n3 points starting from the specified buffer memory address. s can be designated in the following range: -32768 to 32767 or 0H to FFFFH.

Following figure shows an example when the constant 5 is designated to s.



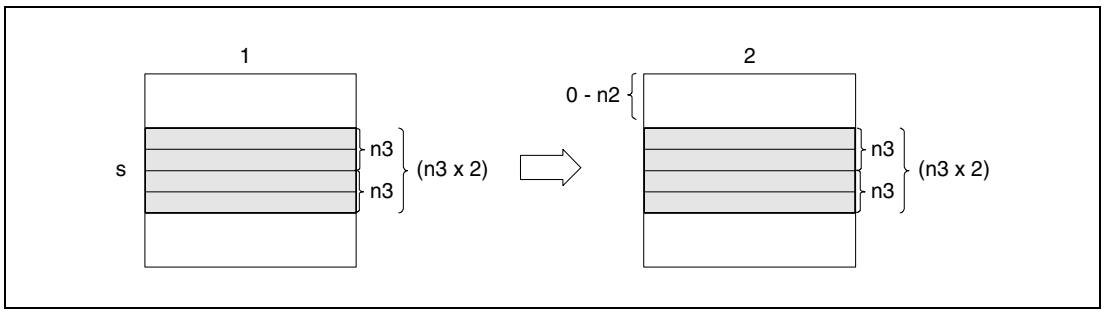
[1] CPU module

[2] Buffer memory of special function module

[3] n3 words (same data is written)

DTO Writing 2-word data (32-bit)

The DTO instruction writes 2-word data from the memory of the CPU to the buffer memory of a special function module. The first address of the memory area data is to be written to is specified by n2, the number of data words (2-multiple) is specified by n3, and the address of the special function module is specified by n1. The first address of the memory address area the data is to be read from is specified by s.

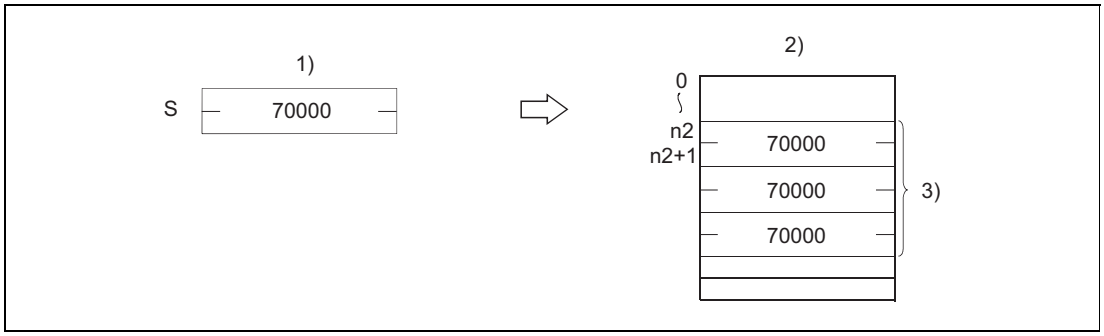


[1] Memory of the CPU

[2] Buffer memory of special function module

When a constant is designated to s, the instruction writes the same data (value designated to s) to the area of n3 x 2 points starting from the specified buffer memory address. s can be designated in the following range: -2147483648 to 2147483647 or 0H to FFFFFFFFH.

Following figure shows an example when the constant 70000 is designated to s.



[1] CPU module

[2] Buffer memory of special function module

[3] n3 x 2 words (same data is written)

NOTE

Data read from intelligent function modules is also possible with the use of a special function module device. In this case the devices are specified as U□\G□ (U (Headadress of the special function module)/G (Buffer memory adress)).

For the special function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

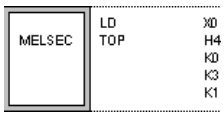
- No signals have been exchanged with the special function module at the execution of the instruction.
  (Error code 1412)
- An error has occured in the special function module at the execution of the instruction.
  (Error code 1402)
- The I/O number specified by n1 is not a special function module.
  (Error code 2110)
- The number of data words specified by n3 (2 x n3 for DTO) exceeds the storage range of the device specified by d.
  (Error code 4101)
- The address specified by n2 is outside the buffer memory range.
  (Error code 4101)

Program Example 1

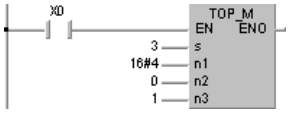
TOP

With leading edge from X0, the following program sets the channels CH1 and CH2 on an Q68AD module to execute A/D conversion. The special function module is at address 040 through 05F. The value 3 is written to the buffer memory at address 0.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

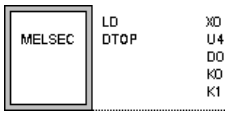
```
LD    X0
TOP_M 3 , 16#4 , 0 , 1
```

Program Example 2

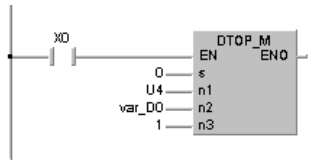
DTOP

With leading edge from X0, the following program resets the x-data values at the buffer memory addresses 41 and 42 of a QD75P4 module to 0. The special function module is at address 040 through 05F.

MELSEC Instruction List



Ladder Diagram



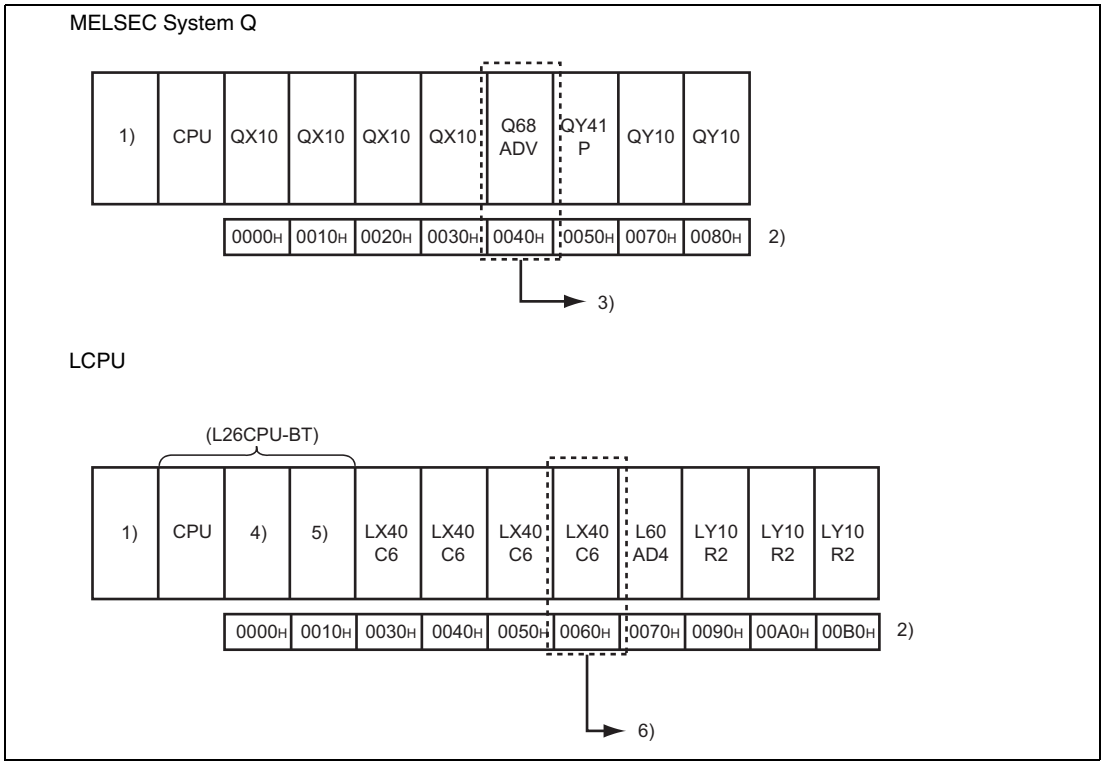
IEC Instruction List

```
LD    X0
DTOP_M 0 , U4 , var_D0 , 1
```

NOTE

The program example 2 will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

NOTE *The value of n1 is specified by the upper 3 digits of hexadecimal 4-digit representation of the head I/O number of the slot in which an intelligent function module is mounted.*



[1] Power supply module
[2] Head I/O number configured in the I/O assignment setting
[3] Head address of special function module: 0040H --> n1 = K4 or H4
[4] Built-in I/0
[5] Built-in CC-Link
[6] Head address of special function module: 0060H --> n1 = K6 or H6

QCPU and LCPU establish the automatic interlock of the TO/DTO instructions.

7.9 Display instructions

The CPU modules of the MELSEC System Q and the L series supply several instructions that output ASCII characters at the outputs of an output module or on a LED display on the front panel of suitable CPU modules.

Function	MELSEC instruction in MELSEC Editor	MELSEC instruction in IEC Editor
ASCII character output	PR	PR_M
	PRC	PRC_M
Clear display	LEDR	LEDR_M

The LED display complies to the following priority:

1. Display of self diagnostics error
2. Display of CHK instruction
3. Display of number of annunciator F
4. BATTERY ERROR

If one of the first three displays is indicated, the execution of a display instruction does not change the current reading.

7.9.1 PR

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●			

Devices

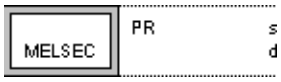
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	● [2]	● [2]	—	—	—	●	●	—
d	● [1]	—	—	—	—	—	●	—	—

[1] Y only

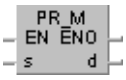
[2] Local devices and the file registers set for individual programs cannot be used.

GX IEC Developer

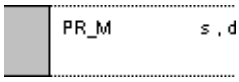
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
s	First number of device storing ASCII code	Character string
d	Head address of output module for ASCII code output	Bit

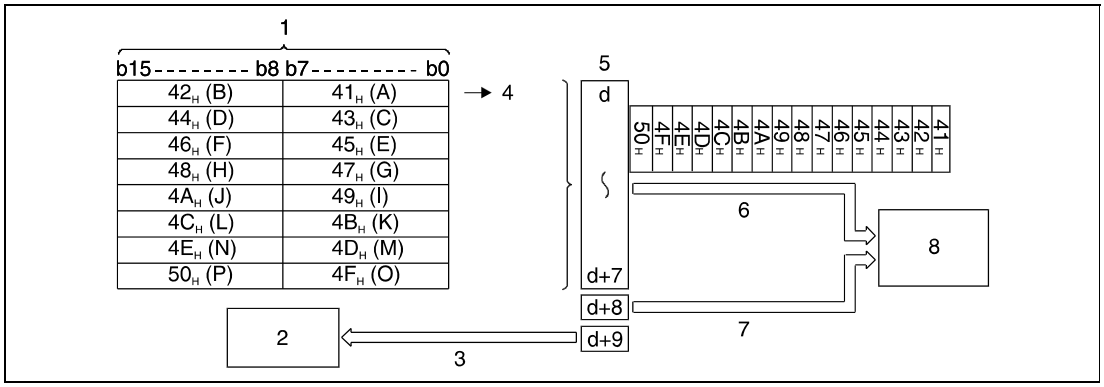
Functions **Output to a peripheral device**

PR **Output of an ASCII character string**

The PR instruction supplies two functions. Its function depends on the status of special relay SM701.

● SM701 set (1) (function 1)

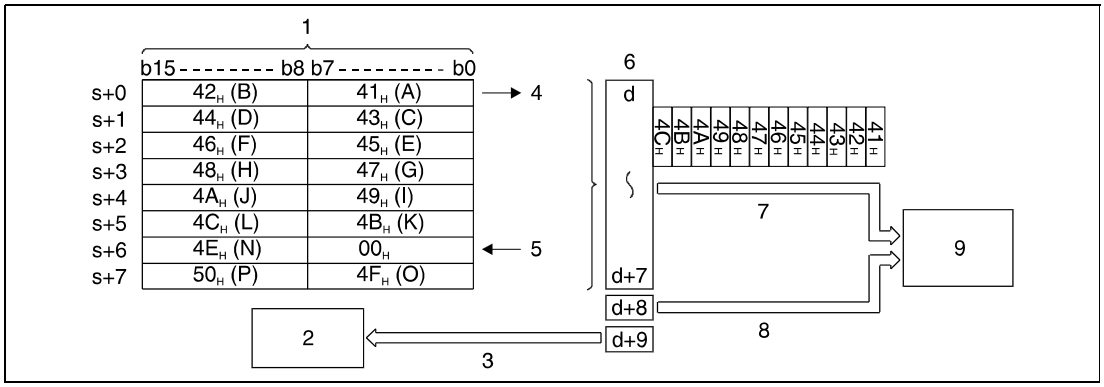
Output of an ASCII character string of 16 characters to an output module. The character string, divided into twice 8 characters, is read from the address area s and output to the outputs specified by d.



[1] Device storing ASCII code
[2] Sequence program
[3] Flag indicating that PR instruction is in progress (used as interlock)
[4] Start of output
[5] Outputs Y
[6] Output of ASCII code
[7] Output of strobe signal
[8] Printer or display device

● SM701 not set (0) (function 2)

Output of ASCII character string data up to the character code "00H" in hexadecimal format from the address area s to the outputs specified by d.



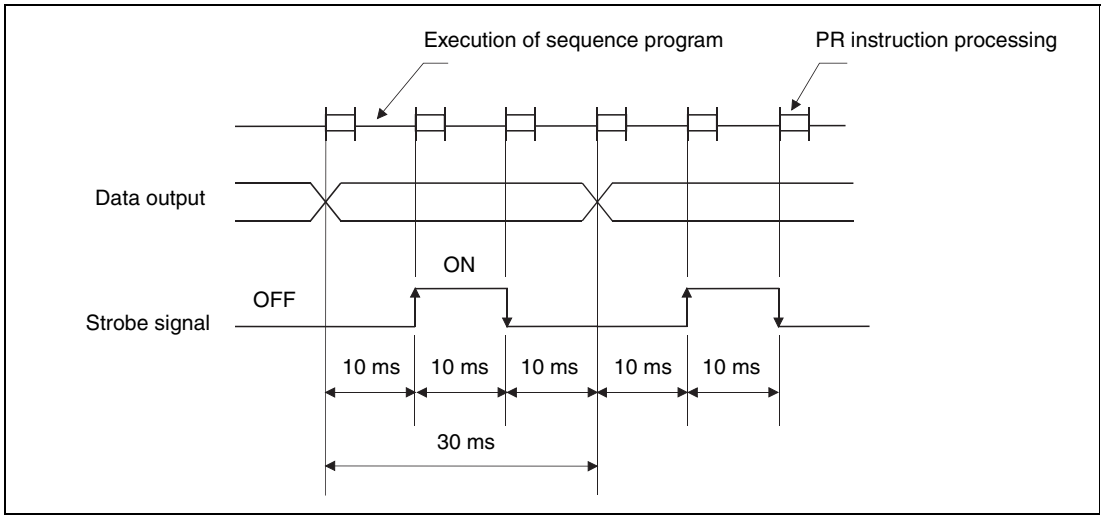
[1] Device storing ASCII code
[2] Sequence program
[3] Flag indicating that PR instruction is in progress (used as interlock)
[4] Start of output
[5] End of character string (end of transmission)
[6] Outputs Y
[7] Output of ASCII code
[8] Output of strobe signal
[9] Printer or display device

If the content of the devices storing ASCII code is overwritten during the output, the current data is output.

Following the execution of the PR instruction, the PR instruction execution flag (d+9 device) remains ON until the completion of the transmission of the designated number of characters.

For the execution of a PRC instruction an output module with 10 successive binary outputs is needed. The address area begins at the output number specified by d. The 10 output addresses of the output module are processed independently from an I/O refresh after the END instruction in the program sequence.

Output signals from the output module are transmitted at the rate of 30 ms per character. Thus, processing n characters takes n x 30 ms. The output transmission is controlled via 10 ms interrupts, so the sequence program is processed continuously.



In addition to the ASCII code a strobe signal (ON = 10 ms, OFF = 20 ms) is output at address Y= d+8.

The PR and PRC instructions can be executed multiple times. Yet, an interlock should be established via the PR instruction execution flag (output device Y= d+9) so the PR and PRC instructions are not executed simultaneously.

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- There is no 00H code within the range of the device specified by s when SM701 is OFF. (Error code 4101)

Program Example

PR

With leading edge from X0, the following program converts the character string "ABCDEFGH-IJKLMNOP" into ASCII code and stores it in data registers D0 through D7. After setting X3 ON, the ASCII code in D0 through D7 is output to the outputs Y14 through Y1D.

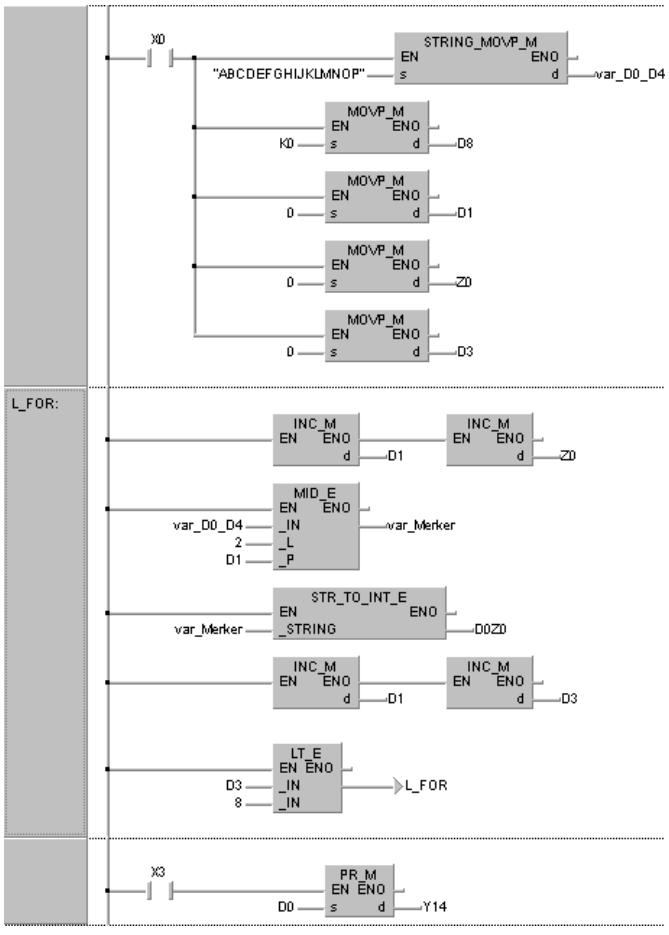
MELSEC Instruction List

```
MELSEC   LD      X0
         $MOVP   "ABCDEFGH"
                 D0
         $MOVP   "IJKLMNOP"
                 D4
         LD      X3
         PR      D0
                 Y14
```

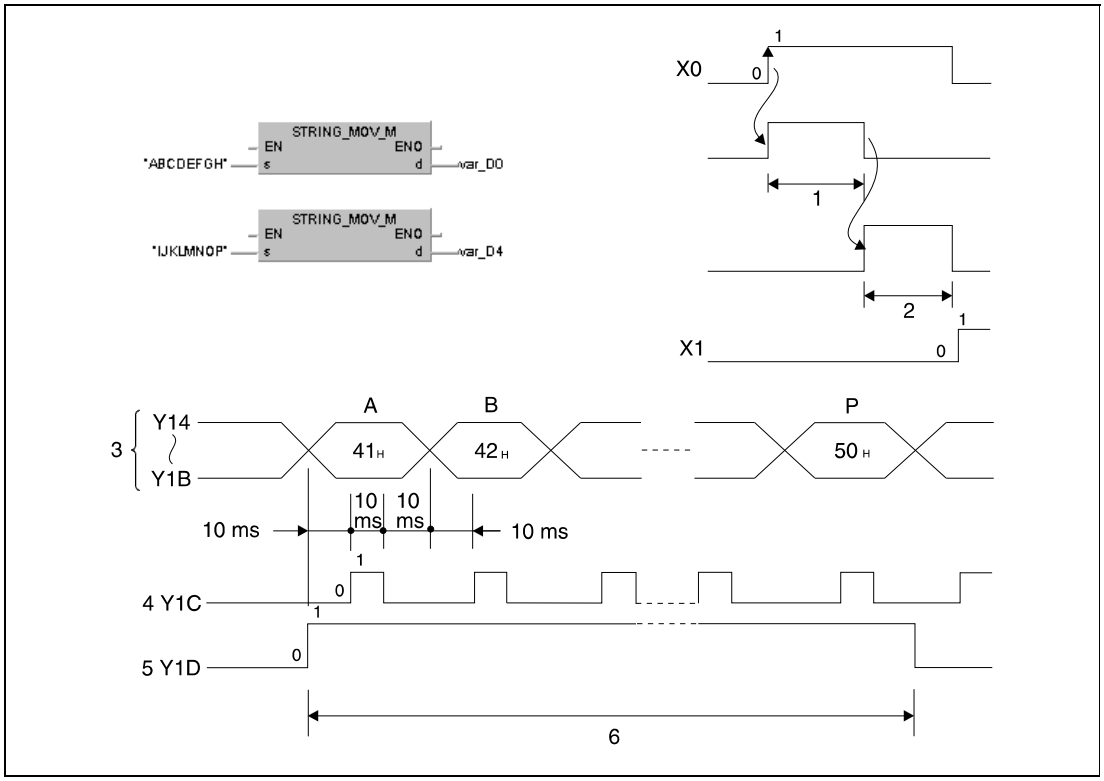
IEC Instruction List

```
         LD              X0
         STRING_MOVP_M   "ABCDEFGH", D0
         STRING_MOVP_M   "IJKLMNOP", D4
         LD              X3
         PR_M            D0, Y14
```

Ladder Diagram



The following timing charts illustrate the processing of the program:



[1] Storage of character string "ABCDEFGH" in D0 through D3

[2] Storage of character string "IJKLMNOP" in D4 through D7

[3] ASCII code

[4] Strobe signal

[5] PR instruction execution flag

[6] Processing the PR instruction (period = 480 ms)

NOTES

If SM701 is not set, the value "00H" has to be written to register D8. Without this character code an operation error would occur in the program example above.

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.9.2 PRC

CPU

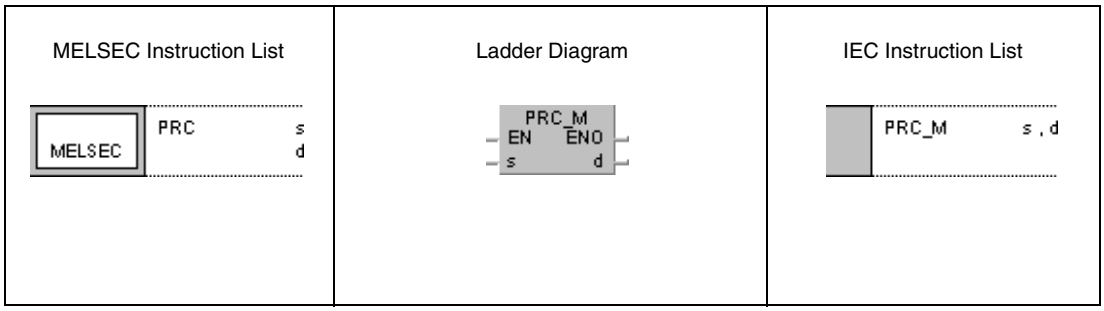
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●			

Devices

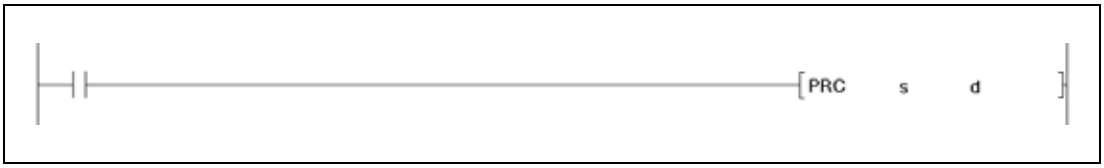
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				P, I, J, U
s	●	●	●	●	●	●	—	—	●
d	● [1)]	—	—	—	—	—	—	—	—

[1] Y only

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
s	First number of device storing comment to be output	BIN 16-bit
d	Head address of output module for comment output	Bit

Functions

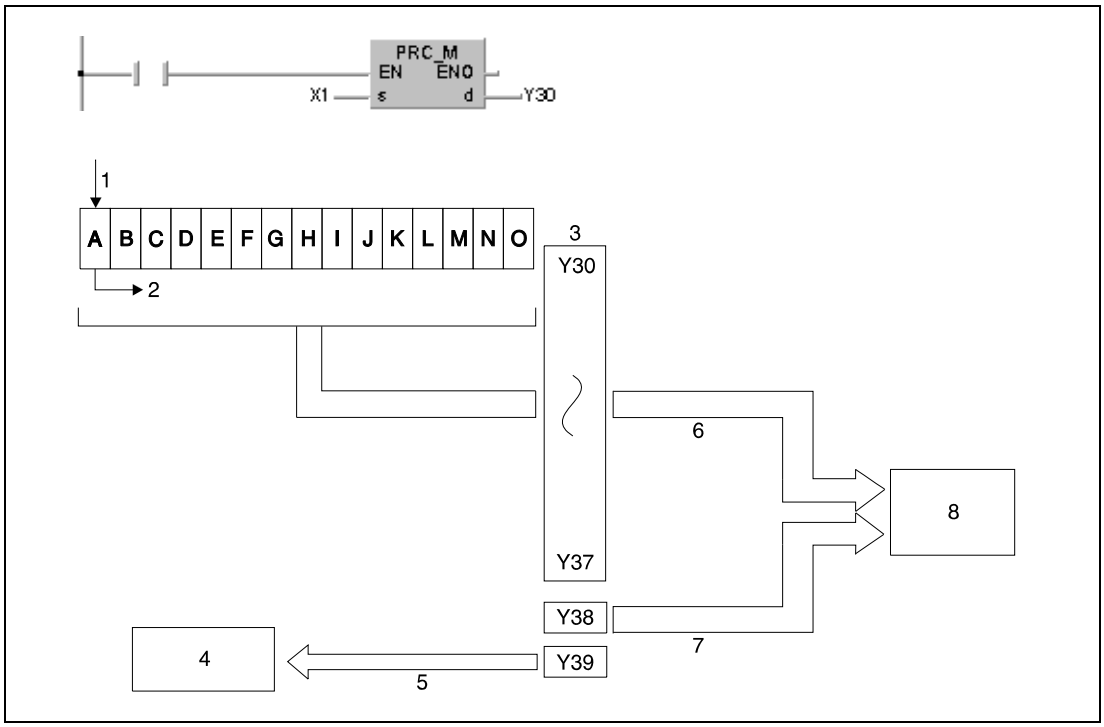
Output to a peripheral device

PRC Output of a comment

The PRC instruction outputs a comment (in ASCII code) of a device designated by s to an output module designated by d.

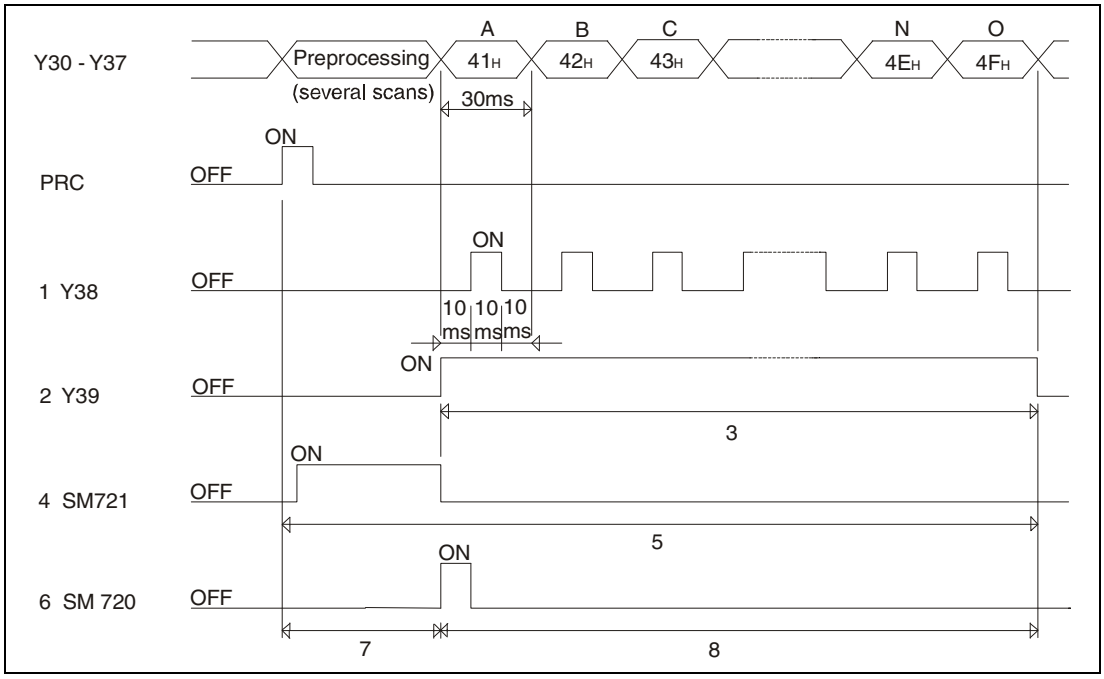
The output of either 16 or 32 characters can be chosen. The choice is specified via special relay SM701.

- If SM701 is set (1), 16 characters are output.
- If SM701 is not set (0), 32 characters are output.



[1] Comment (ASCII code) from X1 onwards
[2] Start of output
[3] Outputs Y
[4] Sequence program
[5] PR instruction execution flag (used as interlock)
[6] Output of ASCII code
[7] Output of strobe signal
[8] Printer or display device

The processing of the PRC instruction is shown in the following timing chart:



[1] Strobe signal

[2] PRC instruction execution flag

[3] Processing time (16 x 30 ms = 480 ms) for the PRC instruction

[4] File access in process flag

[5] The PRC instruction cannot be executed again

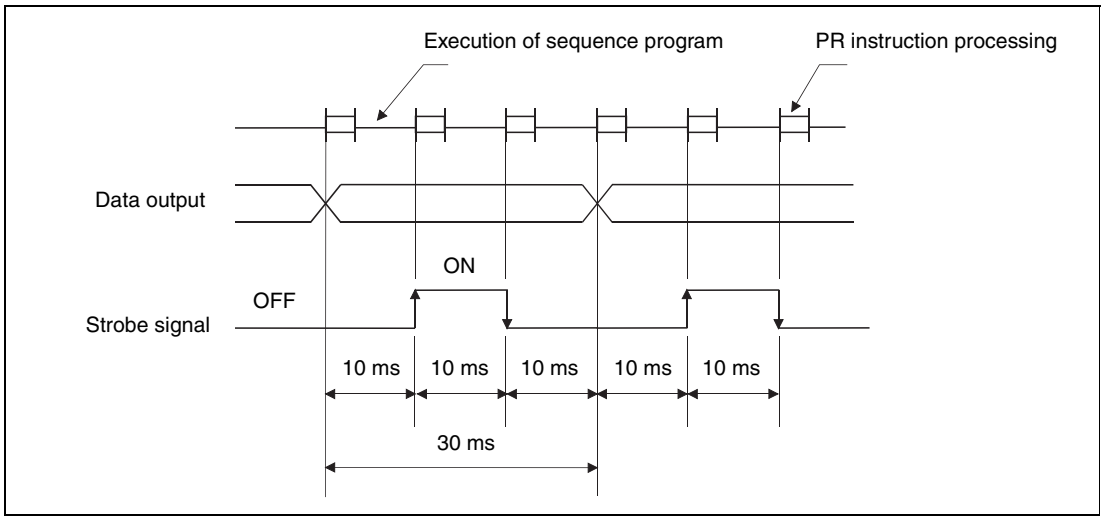
[6] File access completion flag

[7] No other instruction can be executed

[8] Instructions other than PRC, SP.FREAD, SP.FWRITE, PLOAD, PUNLOAD and PSWAPP can be executed

There are 10 binary outputs of a digital output module assigned. The address area begins at the output address Y specified by d.

Output signals from the output module are transmitted at the rate of 30 ms per character. Thus, processing n characters takes n x 30 ms. The output transmission is controlled via 10 ms interrupts, so the sequence program is processed continuously.



In addition to the ASCII code a strobe signal (ON = 10 ms, OFF = 20 ms) is output at address Y= d+8.

During the output of 16 characters of ASCII code, the PRC instruction execution flag d+9 is set ON. Thus, the output Y at address d+9 is set as long as the PRC instruction is executed. The PR and PRC instructions can be executed multiple times. Yet, an interlock should be established via the PRC instruction execution flag (output device Y= d+9) so the PR and PRC instructions are not executed simultaneously.

If the address area s does not contain data, the instruction is not executed.

The PRC instruction can only access comments already stored in the PLC. For conversion from alphanumeric data into ASCII code an ASC instruction has to be applied.

After the execution of the PRC instruction is finished, SM720 turns ON for one scan. SM721 turns ON during the execution of the PRC instruction. The PRC instruction cannot be executed when SM721 is already ON. If an attempt is made, the processing will not be performed.

NOTES

The PRC instruction can only access comments stored in a memory card. The PRC instruction can not access comments stored in the internal memory.

The comment file accessed by the PRC instruction is set at the "PC File Setting" in the Parameter mode. The output of a comment file with the PRC instruction is not possible if no comment file has been set.

Do not execute the PRC instruction during an interrupt program. Otherwise, malfunction may result.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

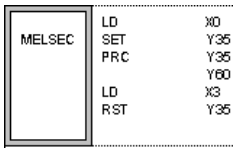
- The PRC instruction is executed while a comment is written during RUN. (Error code 4100)

Program Example

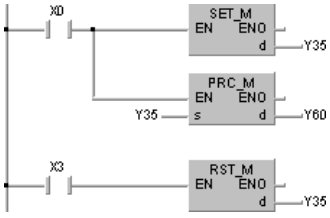
PRC

If X0 is set ON, the following program sets output Y35 ON and outputs the comment at Y35 in ASCII code simultaneously at the outputs Y60 through Y69. After setting X3 ON, Y35 is reset OFF.

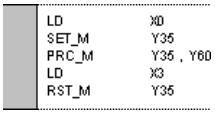
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



7.9.3 LEDR

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

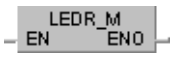
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List

GX Works2

Variables

Set Data	Meaning	Data Type
—	—	—

Functions

Resetting annunciators and error displays

LEDR Reset instruction

The LEDR instruction resets annunciators that were set automatically when an operation error occured.

Operation of the LEDR instruction with an annunciator set during self-diagnosis:

If during self-diagnosis an error occurs that does not affect the accurate operation of the CPU, the execution of a LEDR instruction clears the "ERROR" LED or the error display on the CPU.

In addition, SM0, SM1 and SD0 at the user program have to be reset, because they are not reset automatically by the LEDR instruction. Further steps required to reset the annunciator are not executed neither.

Operation of the LEDR instruction on occurence of a battery error:

If the LEDR instruction is executed after a battery replacement, the "BAT. ARM/BAT." LED on the front panel of the CPU and the error display on the CPU are cleared. At the same time, SM51 is reset automatically.

Operation of the LEDR instruction with an annunciator F set on a CPU without LED display:

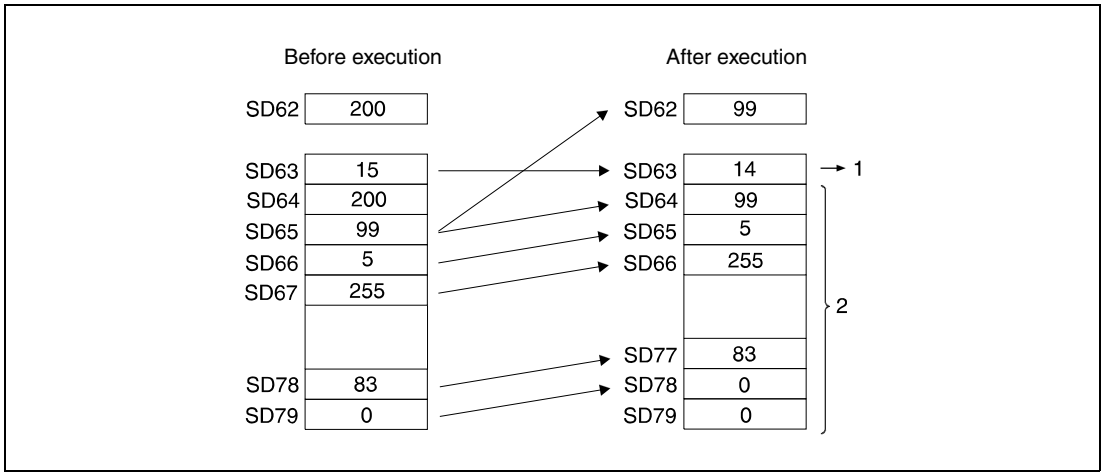
After execution of the LEDR instruction the following operations are executed:

– The "USER" LED on the front panel of the CPU flickers and then turns off.

– The annunciator F stored in SD62 and SD64 are reset and the annunciators stored in SD65 through SD79 are shifted for further processing.

– The new number of annunciator F shifted to SD64 is written to SD62.

– The accumulator of the annunciator in SD63 is decremented by 1. If SD63 is already at 0, this value remains unchanged.



[1] Number of stored annunciators

[2] F number storage area

Operation of the LEDR instruction with an annunciator F set on a CPU with LED display:

After execution of the LEDR instruction, the following operations are executed:

– The annunciator displayed on the LED display of the CPU is cleared.

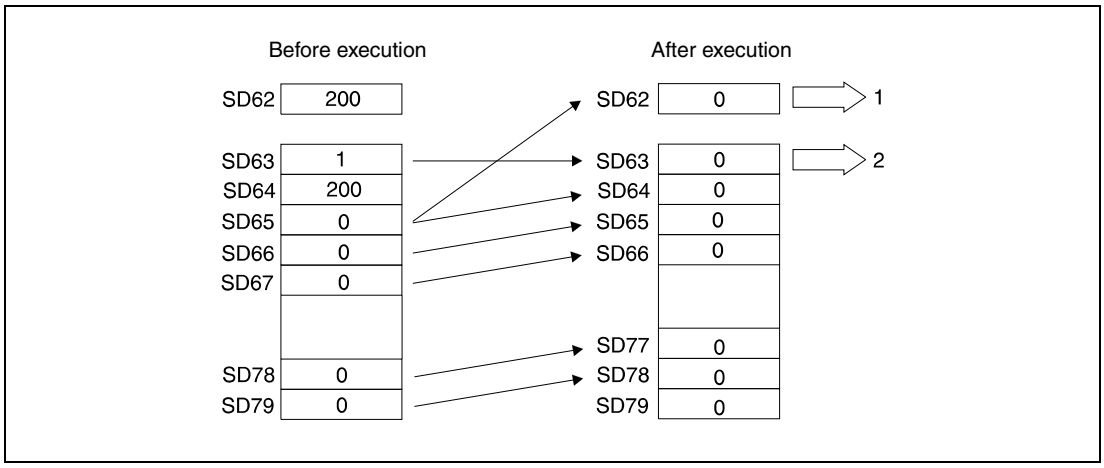
– The "USER" LED on the front panel of the CPU flickers and then turns off.

– The annunciators F stored in SD62 and SD64 are cleared and the annunciators stored in SD65 through SD79 are shifted for further processing.

– The new number of annunciator F shifted to SD64 is written to SD62.

– The accumulator of the annunciator in SD63 is decremented by 1. If SD63 is already at 0, this value remains unchanged.

– The current number of annunciator stored in SD62 is displayed. If SD63 is already at 0, there is nothing displayed.



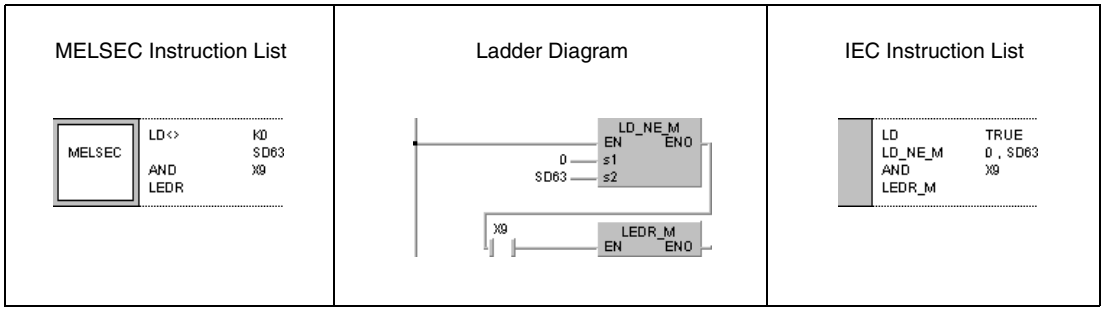
[1] Since SD63 is at value 0, no annunciator is displayed on the LED display.

[2] Number of stored annunciators

Program Example

LEDR

If X9 is set and the value in register SD63 is not equal to 0, the following program executes a LEDR instruction.



NOTE *The defaults for the error item numbers set in special register SD207 to SD209 and the order of priority is shown in the table below:*

Order of priority	Factor number (Hexadecimal)	Description	Remark	QCPU	LCPU
1	1	AC DOWN	Power supply cut	●	●
		SINGLE PS.DOWN	Redundant base unit power supply voltage drop	●	○
		SINGLE PS.ERROR	Redundant power supply module fault	●	○
2	2	UNIT VERFY ERR.	I/O module verify error	●	○
		FUSE BREAK OFF	Blown fuse	●	○
		SP. UNIT ERROR	Special function module verify error	●	○
		SP. UNIT DOWN	Special function module verification error Special function module error	○	●
3	3	OPERATION ERROR	Operation error	●	●
		LINK PARA ERROR	Link parameter error	●	○
		SFCP OPE. ERROR	SFC instruction operation error	●	○
		SFCP EXE. ERROR	SFC program execution error	●	○
		REMOTE PASS.FAIL	Remote password error	○	●
		SNTP OPE.ERROR	SNTP error	○	●
4	4	ICM.OPE ERROR	Memory card operation error	●	●
		FILE OPE ERROR	File assess error	●	●
		EXTEND INST. ERROR	Extend instruction error	●	○
		OPE. MODE DIFF.	Operation status, switch mismatch	●	○
		CAN'T EXE.MODE	Current mode-time function execution disabled	●	○
		TRK.TRANS.ERR.	Tracking data transmission error	●	○
		TRK.SIZE ERROR	Tracking capacity excess error		
		TRK.DISCONNECT	Tracking cable not connected, failure		
		FLASH ROM ERROR	Flash ROM access count exceeded error	○	●
5	5	PRG.TIME OVER	Constant scan setting time over error	●	●
			Low speed execution monitoring time over error	●	○
6	6	CHK instruction	Error detected with CHK instruction	●	●
7	7	Annunciator	—	●	●
8	8	LED instruction	—	●	●
9	9	BATTERY ERR.	Battery error	●	●
10	A	Clock data	—	●	●
11	B	CAN'T SWITCH	System switching error	●	○
		STANDBY SYS.DOWN	Standby system not started/stop error	●	○
		EM.COPY EXE.	Memory copy function executed	●	○
12	C	DISPLAY ERROR	Display unit error	○	●

● Error or event occurs
○ Error or event does not occur

If the highest priority is given to the annunciator, it can be reset with priority by the LEDR instruction. (Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU)

7.10 Failure diagnosis and debugging

These instructions are for failure diagnosis and debugging support failure checks.

The following table gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Failure check	CHKST	CHKST_M
	CHK	CHK_M
	CHKCIR	CHKCIR_M
	CHKEND	CHKEND_MD

NOTE *Please check, whether these functions are available and supported by your version of the GX IEC Developer.*

7.10.1 CHKST, CHK

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●		

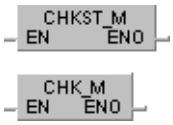
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

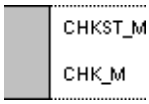
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
—	—	—

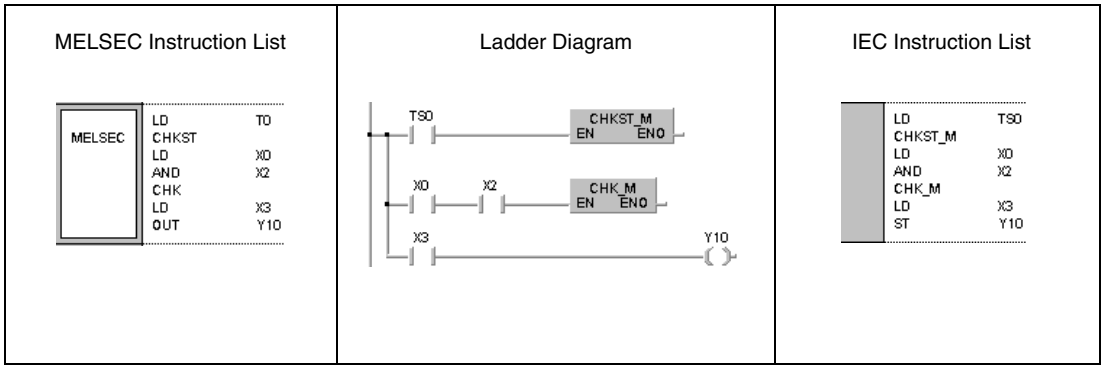
Functions

Failure check for bidirectional operations

CHKST Start instruction for the CHK instruction

The CHKST instruction starts the execution of the CHK instruction. If the execution condition for the CHKST instruction is not set (0), the program step following the CHK instruction will be executed.

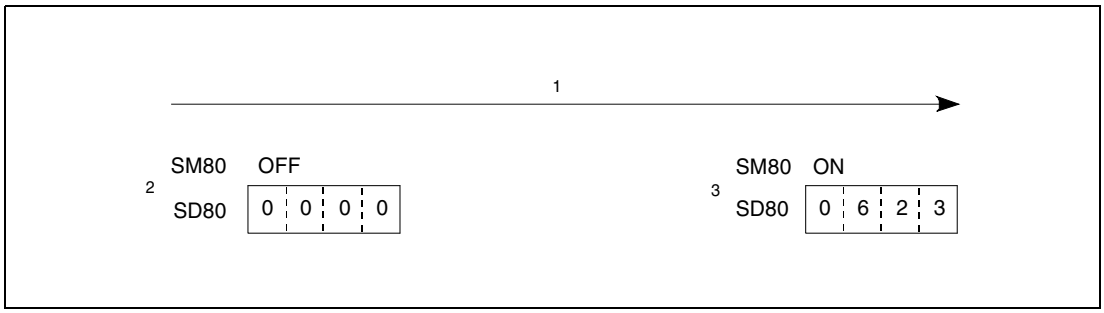
With the execution condition for the CHKST instruction set (1), the CHK instruction is executed. In the ladder diagram below these instructions are programmed.



CHK Failure check instruction

The CHK instruction with some CPU types (and depending on the control mode) supports failure check operations for contact circuits with limit switches that monitor bidirectional movement. Once an error occurs within such a circuit, the special relay SM80 is set and the corresponding error code is stored in special register SD80.

The error code is stored as BCD 4-digit data value in special register SD80. The upper 3-digits store the contact number of the corresponding contact (here contact 62) and the lower digit stores the number of the failure check circuit (coil number 1 to 6; here coil number 3).



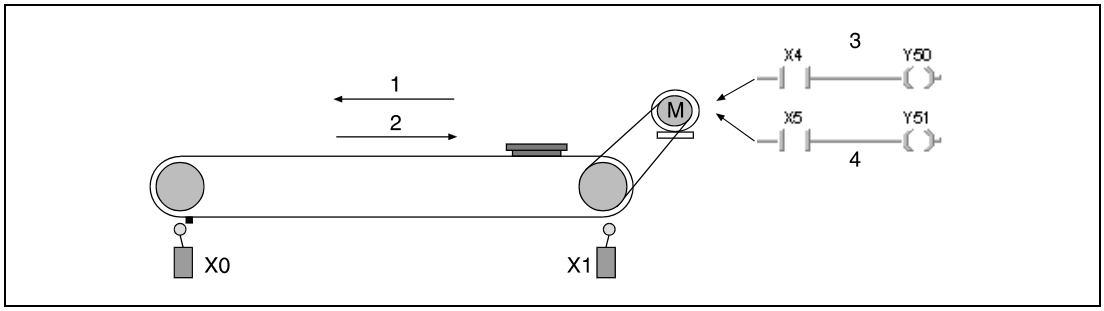
[1] Contact 62; coil number 3 (during failure check)

[2] Before failure check

[3] After failure check

The input contacts programmed prior to the CHK instruction do not serve as execution condition for the CHK instruction but as specification of the check conditions.

In the following, the failure check programming via the CHK instruction is illustrated with a concrete example. The following illustration shows a conveyor belt that moves from the left to the right travel limit. The corresponding travel limits are detected via limit switches (X0 and X1). The start contact for advance movement is X4 and for retract movement is X5.



[1] Advance movement
[2] Retract movement
[3] Advance command
[4] Retract command

The diagrams below show a sample program for the operation and failure check of the conveyor belt shown above.

During error free operation the program jumps to the program step following the CHK instruction. With leading edge from X4, the conveyor belt is advanced, and Y0 is set for failure check. With leading edge from X5, the conveyor belt is retracted, and Y0 is reset.

The timer T0 watches the duty cycle time. If the duty cycle time is exceeded the CHKST instruction is set via the contact TS0. In the next program step the CHK instruction is executed, and the error code is stored in the special register SD80.

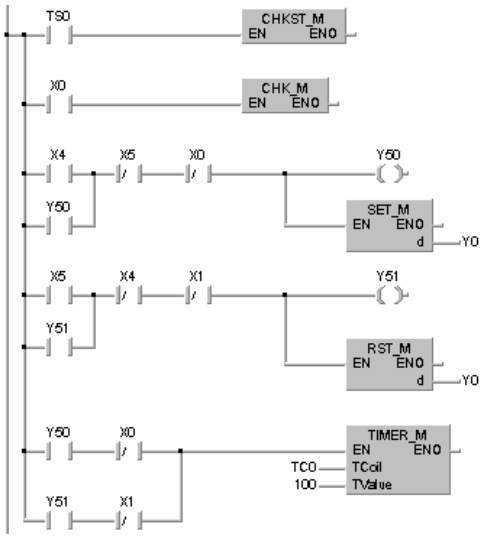
MELSEC Instruction List

```
MELSEC  LD      T0
        CHKST
        LD      X0
        CHK
        LD      X4
        OR      Y50
        ANI     X5
        ANI     X0
        OUT     Y50
        SET     Y0
        LD      X5
        OR      Y51
        ANI     X4
        ANI     X1
        OUT     Y51
        RST     Y0
        LD      Y50
        ANI     X0
        LD      Y51
        ANI     X1
        ORB
        OUT     T0
                K100
```

IEC Instruction List

```
LD          TS0
CHKST_M
LD          X0
CHK_M
LD          X4
OR          Y50
ANDN        X5
ANDN        X0
ST          Y50
S           Y0
LD          X5
OR          Y51
ANDN        X4
ANDN        X1
ST          Y51
R           Y0
LD          Y50
ANDN        X0
OR(
            Y51
ANDN        X1
)
TIMER_M     TC0 , 100
```

Ladder Diagram



The operations of the CHK instruction can be illustrated through the following ladder diagrams, of which the functions are similar to the execution of the CHK instruction.

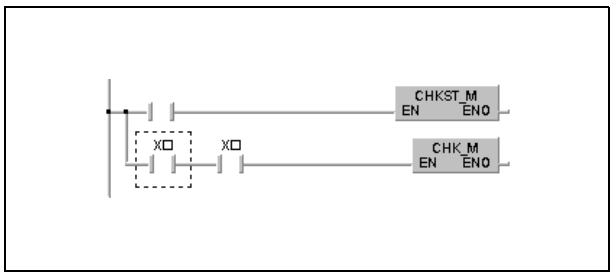
The contact numbers of the limit switches for advance movement X☐ and retract movement X☐+1 have to be designated successively. The number of the advance limit switch X☐ must be less than the number of the retract limit switch X☐+1. The contact number of the advance limit switch is assigned to an output Y☐ with the same address. According to the program example, this output is set during advance movement and reset during retract movement.

For better comprehensibility of the program example above, the contacts X0 (X☐), X1 (X☐+1) and Y0 (Y☐) are applied directly for specification of the coil number. Depending on the program they can be replaced by any other number.

NOTE *The outputs Y☐ are treated as internal relays and cannot be output to external devices.*

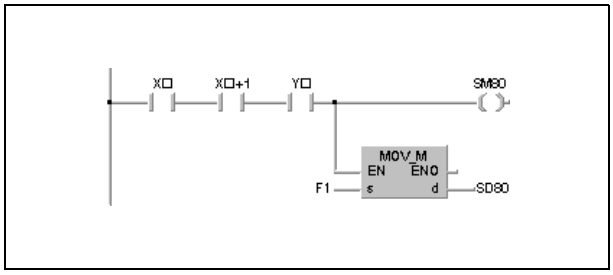
The following diagrams concerning the CHK instructions and the 6 generated failure check circuits (error conditions) are arranged in pairs.

In the following, the CHK instructions are illustrated. The contact indicated X☐ serves as variable for maximum 150 contacts (150 conveyor belts or similar applications).



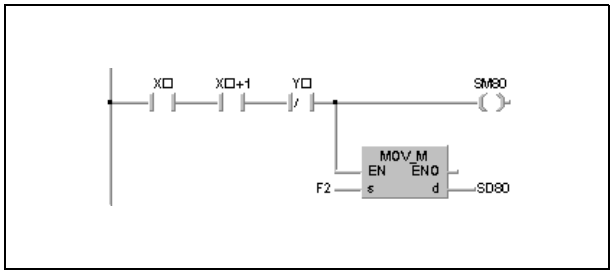
Failure check circuit 1 (coil number 1):

Both limit switches respond to the advance movement of the conveyor belt.



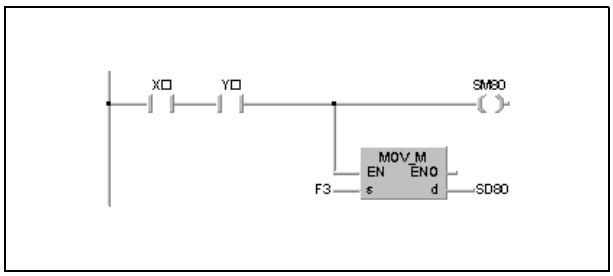
Failure check circuit 2 (coil number 2):

Both limit switches respond to the retract movement of the conveyor belt.



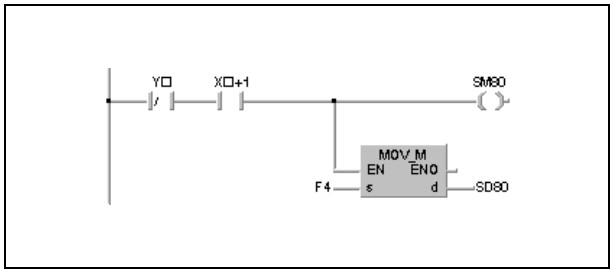
Failure check circuit 3 (coil number 3):

Advance command for set advance limit switch.



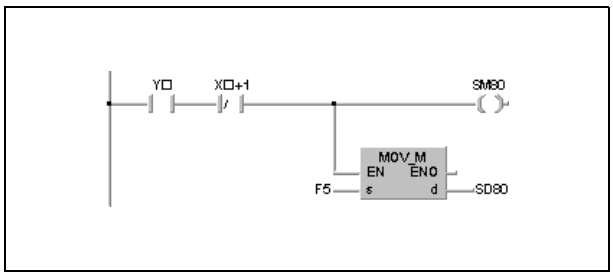
Failure check circuit 4 (coil number 4):

Retract command for set retract limit switch.



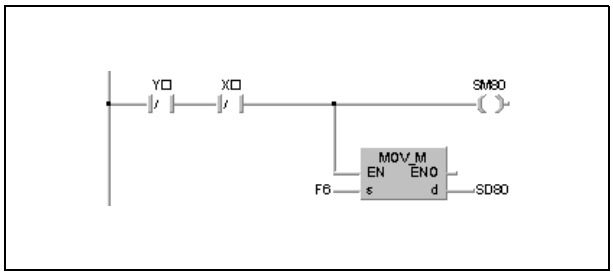
Failure check circuit 5 (coil number 5):

Advance command for reset retract limit switch.

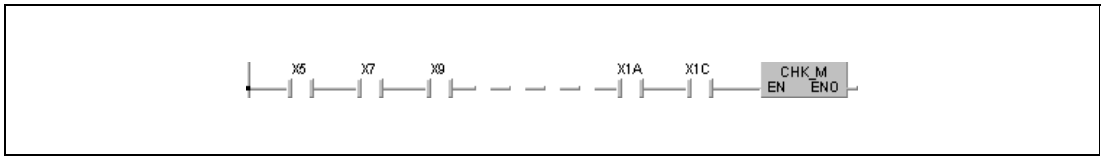


Failure check circuit 6 (coil number 6):

Retract command for reset advance limit switch.



The CHK instruction can designate a maximum of 150 contact numbers for advance limit switches. For the designation of contact numbers any contact number of the retract limit switch is skipped.

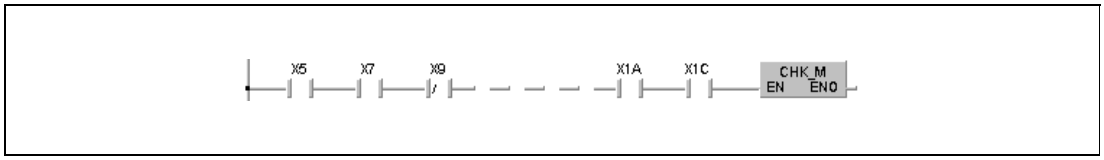


The relay SM80 and the special register SD80 have to be reset after execution of the CHK instruction because they retain their condition after being set. If they are not reset prior to another CHK instruction, the instruction cannot be executed.

The CHKST instruction has to be programmed prior to the CHK instruction. An error will be returned if an instruction other than the LD, LDI, AND or ANI instruction is used between the CHK instruction and the CHKST instruction.

The CHK instruction can be programmed in any program step of the sequence program. The CHK instruction can be used up to two times in all program files being executed. In a single program file a CHK instruction may be used only once.

The coil numbers have to be programmed via a LD or AND instruction prior to the CHK instruction. Other input instructions are not supported. If an LDI or ANI instruction is programmed, the failure check of the CHK instruction cannot be executed. The contact numbers designated for the failure check however can be designated via the LDI and ANI instructions. In the diagram below the switch with the number X9 is ignored because it is an NC contact (normally closed).



The failure detection method depends on the status of the special relay SM710 as follows.

- SM710 is reset (0):

  The failure check is performed in coil number (failure check circuit) sequence from contact 1(limit switch) to contact n (limit switch).

  The first contact is checked from coil number 1 through coil number 6. Then the next contact is checked from coil number 1 through coil number 6. The operation is completed after the nth contact is checked from coil number 1 through coil number 6.

- SM710 ist set (1):

  The failure check is performed in contact number (limit switch) sequence from coil 1 (failure check circuit) through coil 6 (failure check circuit).

  The first coil is checked from contact number 1 through contact number n. Then the next coil is checked from contact number 1 through contact number n. The operation is completed after the 6th coil is checked from contact number 1 through contact number n.

If more than one failure is detected, the number of the first failure detected is stored. Further detected failures are ignored.

The CHK instruction cannot be used by a low speed execution type program. If a low speed execution type program has been set in a program file containing the CHK instruction, an operation error will be returned, and the CPU module operation will be suspended.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- Two failure check input contacts within one failure check circuit are connected in parallel. (Error code 4235)
- There is an NOP instruction. (Error code 4235)
- More than 150 input devices are specified. (Error code 4235)
- A CHKST instruction is not followed by a CHK instruction. (Error code 4235)
- A CHK instruction is executed without a prior CHKST instruction. (Error code 4235)
- The CHKST and CHK instruction are used in a low speed execution type program. (Error code 4235)
- There is an instruction other than the LD, LDI, AND or ANI instruction between the CHK instruction and the CHKST instruction. (Error code 4235)
- The CHK instruction is used at three places or more in all of programs being executed. (Error code 4235)
- The CHK instruction is used at two places or more in a single program. (Error code 4235)

7.10.2 CHKCIR, CHKEND

CPU

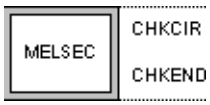
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●		

Devices

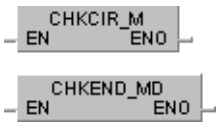
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

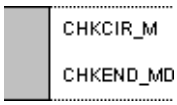
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
—	—	—

Functions **Generating check circuits for the CHK instruction**

CHKCIR, CHKEND **Start and end instructions for a program part with generated check circuits**

The CHKCIR and CHKEND instructions alter check circuits for the CHK instruction. Any required check format can be generated. The actual failure check is performed via the CHKST and CHK instructions.

The failure check is executed via the error check circuits programmed between the CHK and the CHKEND instruction.

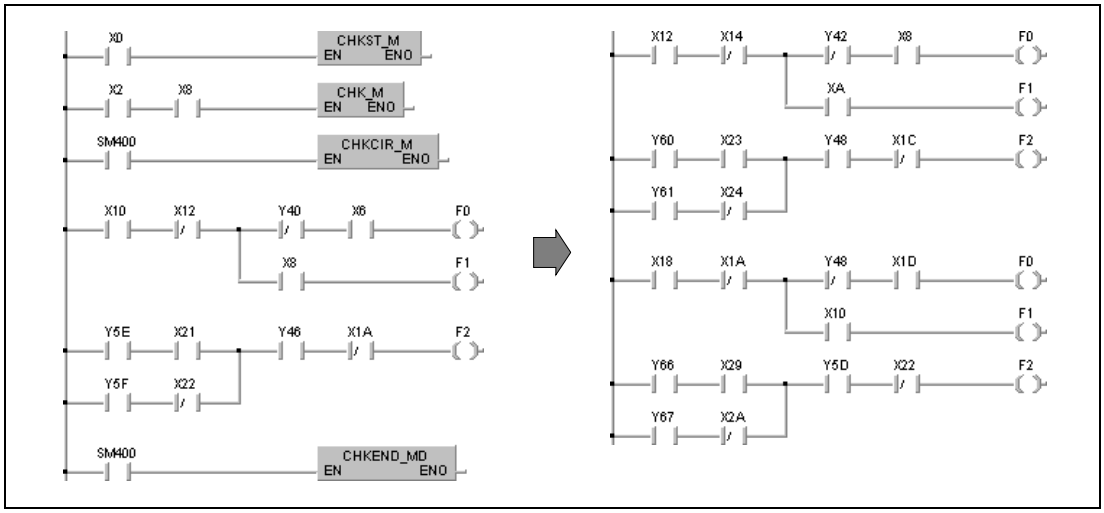
NOTE *If the check circuit format for the CHK instruction was altered via the CHKCIR and CHKEND instructions, connected peripheral devices have to be started up in "General Mode", and a program expansion has to be performed.*

From the error check circuits between the CHKCIR and CHKEND instructions altered error check circuits are generated through index qualification. The error check circuits programmed between these instructions can be assigned 9 annunciators (F1 - F9). Index qualification is performed through the addition of contact numbers designated prior to the CHK instruction and contact numbers of the error check circuits. For example, the contact X10 in the error check circuits shown below will be assigned X12 and X18 in the index qualified check circuits due to the contacts X2 and X8, programmed prior to the CHK instruction.

The error check algorithm depends on the status of the special relay SM710 as follows:

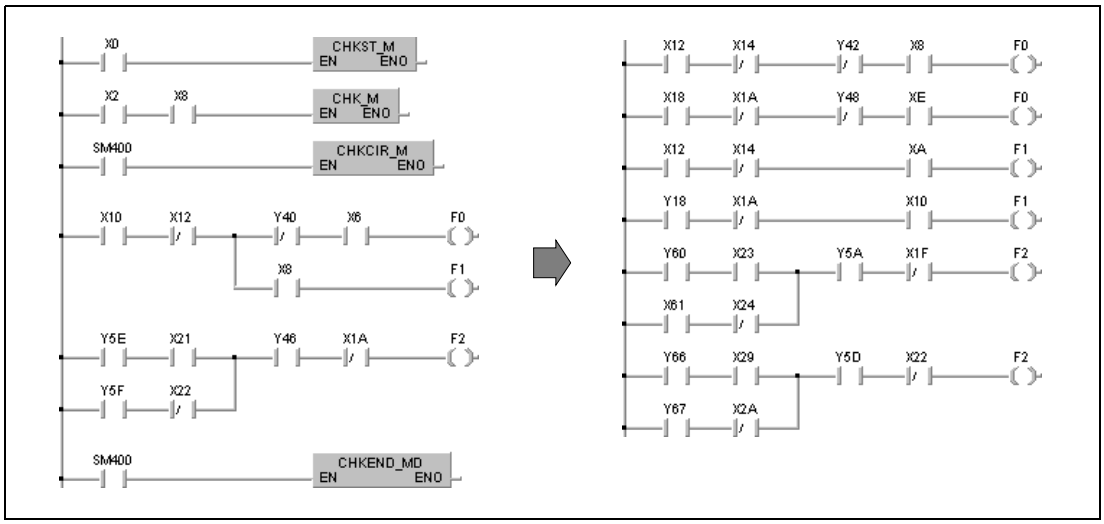
- SM710 is reset (0):

  First in this case, each contact number in the error check circuit programmed between the CHKCIR and CHKEND instruction is index qualified with the first contact number designated prior to the CHK instruction. Then, each programmed check circuit is index qualified again with the second contact number designated prior to the CHK instruction. This operation is completed as for any programmed check circuit with assigned annunciator (F) a total of new check circuits equivalent to the number of input contacts of the CHK instruction exists.



- SM710 is set (1):

  First in this case, the first programmed error check circuit with assigned annunciator is index qualified with all contact numbers programmed prior to the CHK instruction. Then, the following check circuit is index qualified with all contact numbers programmed prior to the CHK instruction. This operation is completed as for any programmed check circuit with assigned annunciator (F) a total of new check circuits equivalent to the number of input contacts of the CHK instruction exists.



During error check of the index qualified error check circuits, the outputs (F) that can only be set via the OUT F instruction are checked for their status. If an output (F) is set, the special relay SM80 is set. The error code consisting of contact number and error check circuit (F1 to F9) is stored in special register SD80 in BCD data format.

The error check circuits between the CHKCIR and CHKEND instruction can be programmed with the following instructions.

- Contacts:

  LD, LDI, AND, ANI, OR, ORI, ANB, ORB, MPS, MPP, MRD, comparison operation instructions.

- Coils:

  OUT F

The inputs X and outputs Y have to be programmed as devices for the contacts.

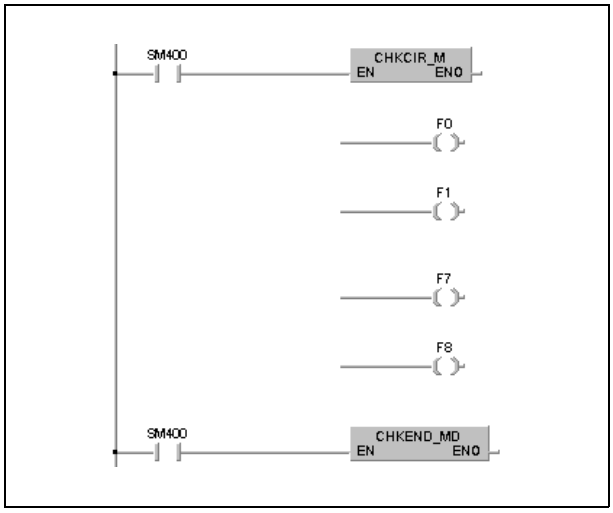
Only annunciators (F) can be programmed as outputs of error check circuits. The error check circuits can be specified any random designation from F0 on, since these outputs are processed as dummy contacts. For this reason, no errors occur with annunciators (F) overlapping.

The status of annunciators (F) can even be checked accurately, if one annunciator (F) is programmed twice beyond the CHK instruction, because both of these annunciator functions are processed separately.

Since the status (0/1) of annunciators (F) applied by the CHK instruction is not updated, the annunciators even remain reset, if they are monitored by a peripheral device.

The error check circuits programmed between the CHKCIR and CHKEND instructions can be created with maximum 256 program steps (contact branches) and 9 outputs (annunciators F1 to F9) addressed by OUT F instructions.

The error check circuits between the CHKCIR and CHKEND instructions are designated from top error check circuit 1 (F0) to bottom error check circuit 9 (F8).



The CHKCIR and CHKEND instructions can be programmed at any program step of the sequence program. In total, these instructions may only exist twice in all program files to be executed and once within one program file.

The CHKCIR and CHKEND instructions cannot be applied in low-speed programs, otherwise an operation error occurs and the CPU terminates processing.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The CHKCIR and CHKEND instructions appear more than twice in all program files. (Error code 4235)
- The CHKCIR and CHKEND instructions appear more than once within one program file. (Error code 4235)
- The CHKEND instruction is not executed after the CHKCIR instruction. (Error code 4230)
- The CHKEND instruction is executed without a preceding CHKCIR instruction. (Error code 4230)
- The CHKCIR and CHKEND instructions are programmed in a low-speed program. (Error code 4235)
- 10 or more annunciators (F) (error check circuits) are addressed. (Error code 4235)
- The created error check circuits contain more than 256 program steps (contact branches). (Error code 4235)
- The error check circuits contain invalid devices. (Error code 4235)
- The error check circuits contain devices already index qualified. (Error code 4235)

Program Example

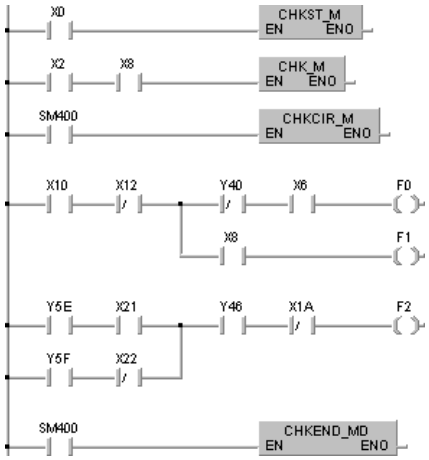
CHKCIR, CHKEND

The following program creates index qualified error check circuits. The operations of this program are illustrated under the topic "functions". In addition, the MELSEC and IEC instruction lists are shown below.

MELSEC Instruction List

```
MELSEC
LD    X0
CHKST
LD    X2
AND   X8
LD    SM400
CHKCIR
LD    X10
ANI   X12
ANI   X6
OUT   F0
LD    X10
ANI   X12
AND   X8
OUT   F1
LD    Y5E
AND   X21
LD    Y5F
ANI   X22
ORB
AND   Y46
ANI   X1A
OUT   F2
LD    SM400
CHKEND
```

Ladder Diagram



IEC Instruction List

```
LD    X0
CHKST_M
LD    X2
AND   X8
LD    SM400
CHKCIR_M
LD    X10
ANDN  X12
ANDN  X6
ST    F0
LD    X10
ANDN  X12
AND   X8
ST    F1
LD    Y5E
AND   X21
OR(
      Y5F
ANDN  X22
)
AND   Y46
ANDN  X1A
ST    F2
LD    SM400
CHKEND_MD
```

7.11 Character string processing instructions

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Conversion of 16-/32-bit binary data into decimal values in ASCII code	BINDA	BINDA_MD
		BINDA _K_MD
		BINDA_S_MD
	BINDAP	BINDA_P_MD
		BINDA_K_P_MD
		BINDA_P_S_MD
	DBINDA	DBINDA_MD
		DBINDA_K_P_MD
		DBINDA_P_S_MD
	DBINDAP	DBINDA_P_MD
		DBINDA_K_P_MD
		DBINDA_P_S_MD
Conversion of BIN 16-/32-bit binary data into ASCII code	BINHA	BINHA_MD
		BINHA_K_MD
		BINHA_S_MD
	BINHAP	BINHA_P_MD
		BINHA_K_P_MD
		BINHA_P_S_MD
	DBINHA	DBINHA_MD
		DBINHA_K_MD
		DBINHA_S_MD
	DBINHAP	DBINHA_P_MD
		DBINHA_K_P_MD
		DBINHA_P_S_MD
Conversion of 4-/8-digit BCD data into ASCII code	BCDDA	BCDDA_MD
		BCDDA_K_MD
		BCDDA_S_MD
	BCDDAP	BCDDA_P_MD
		BCDDA_K_P_MD
		BCDDA_P_S_MD
	DBCDDA	DBCDDA_MD
		DBCDDA_K_MD
		DBCDDA_S_MD
	DBCDDAP	DBCDDA_P_MD
		DBCDDA_K_P_MD
		DBCDDA_P_S_MD

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Conversion of decimal ASCII data into BIN 16-/32-bit binary data	DABIN	DABIN_MD
		DABIN_S_MD
	DABINP	DABIN_P_MD
		DABIN_P_S_MD
	DDABIN	DDABIN_MD
		DDABIN_S_MD
	DDABINP	DDABIN_P_MD
		DDABIN_P_S_MD
Conversion of hexadecimal ASCII data into BIN 16-/32-bit binary data	HABIN	HABIN_MD
		HABIN_S_MD
	HABINP	HABIN_P_MD
		HABIN_P_S_MD
	DHABIN	DHABIN_MD
		DHABIN_S_MD
	DHABINP	DHABIN_P_MD
		DHABIN_P_S_MD
Conversion of decimal ASCII data into 4-/8-digit BCD data	DABCD	DABCD_MD
		DABCD_S_MD
	DABCDP	DABCD_P_MD
		DABCD_P_S_MD
	DDABCD	DDABCD_MD
		DDABCD_S_MD
	DDABCDP	DDABCD_P_MD
		DDABCD_P_S_MD
Read-out of comment data	COMRD	COMRD_MD
		COMRD_S_MD
	COMRDP	COMRD_P_MD
		COMRD_P_S_MD
Detection of character string length	LEN	LEN_E
		LEN_MD
		LEN_S_MD
	LENP	LEN_P_S_MD
Conversion of BIN 16-/32-bit binary data into character string data	STR	STR_MD
		STR_K_MD
		STR_S_MD
	STRP	STR_P_MD
		STR_K_P_MD
		STR_P_S_MD
	DSTR	DSTR_MD
		DSTR_K_MD
		DSTR_S_MD
	DSTRP	DSTR_P_MD
		DSTR_K_P_MD
		DSTR_P_S_MD

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Conversion of character string data into BIN 16-/32-bit binary data	VAL	VAL_MD
		VAL_S_MD
	VALP	VAL_P_MD
		VAL_P_S_MD
	DVAL	DVAL_MD
		DVAL_S_MD
	DVALP	DVAL_P_MD
		DVAL_P_S_MD
Conversion of floating point data into character string data	ESTR	ESTR_M
	ESTRP	ESTRP_M
Conversion of character string data into decimal floating point data	EVAL	EVAL_M
	EVALP	EVALP_M
Conversion of alphanumerical character strings into ASCII code	ASC	ASC_MD
		ASC_K_MD
		ASC_S_MD
	ASCP	ASC_P_MD
		ASC_P_S_MD
		ASC_K_P_MD
Conversion of hexadecimal ASCII values into binary values	HEX	HEX_MD
		HEX_K_MD
		HEX_S_MD
	HEXP	HEX_P_MD
		HEX_K_P_MD
		HEX_P_S_MD
Extraction of character string data (right part of character string)	RIGHT	RIGHT_M
	RIGHTP	RIGHTP_M
Extraction of character string data (left part of character string)	LEFT	LEFT_M
	LEFTP	LEFTP_M
Random extraction of parts from character strings	MIDR	MIDR_M
	MIDRP	MIDRP_M
Selecting and moving parts of character strings into a character string	MIDW	MIDW_M
	MIDWP	MIDWP_M
Search for character strings	INSTR	INSTR_M
	INSTRP	INSTRP_M
	STRINS	
	STRINSP	
	STRDEL	
	STRDELP	
Floating point data conversion with BCD representation	EMOD	EMOD_M
	EMODP	EMODP_M
BCD data conversion with decimal floating point format	EREXP	EREXP_M
	EREXPP	EREXPP_M

7.11.1 BINDA, BINDAP, DBINDA, DBINDAP

CPU

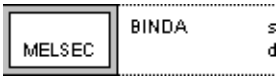
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—

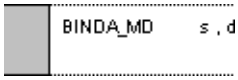
GX IEC Developer

MELSEC Instruction List

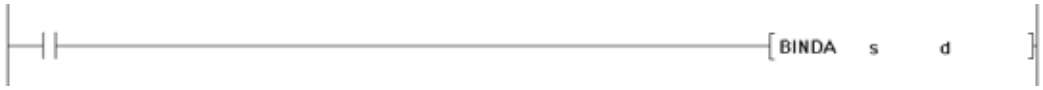


Ladder Diagram

IEC Instruction List



GX Works2



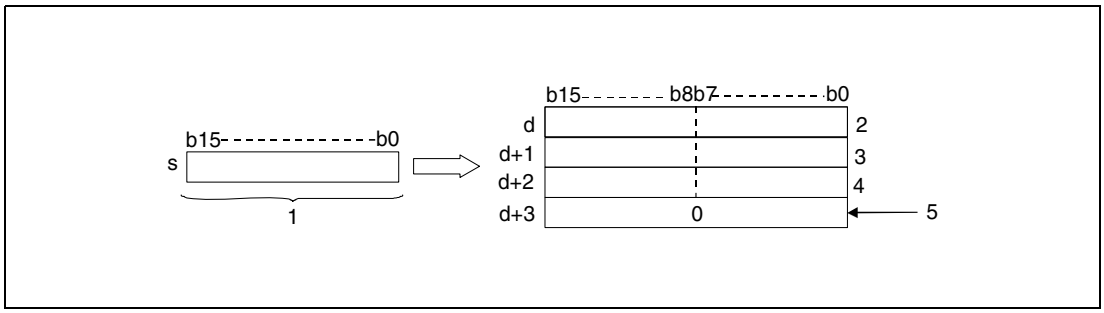
Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	Binary data to be converted into ASCII format	BIN 16-/32-bit	ANY16/32
d	First number of device storing the conversion result	Character string	Array [1..4]/ [1..6] of ANY16

Functions **Conversion of 16-/32-bit binary data into decimal values in ASCII code**

BINDA Conversion of 16-bit binary data

The BINDA instruction converts a 16-bit binary value specified by s into a decimal value in ASCII code and stores it in the device specified in d (Array_d[1]) through d+3 (Array_d[4]).



[1] 16-bit binary data

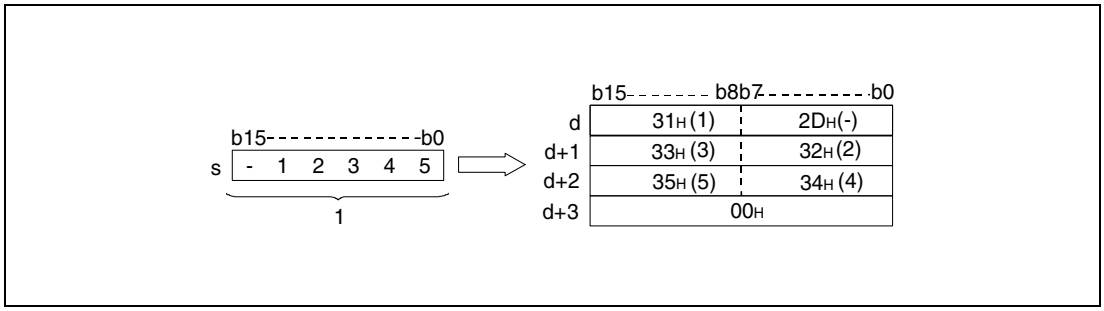
[2] Digit of tenthousands in ASCII code/ sign character

[3] Digit of hundreds in ASCII code/ digit of thousands in ASCII code

[4] Digit of ones in ASCII code/ digit of tens in ASCII code

[5] With the relay SM701 not set

The value specified by s is stored as decimal value in ASCII code beginning from d (Array_d[1]) through d+3 (Array_d[4]).



[1] Binary value

The 16-bit binary value may range from -32768 to 32767.

The results of the conversion operations are stored in d as follows:

– If the 16-bit binary value is positive, the sign character is stored as "20H".

– If the 16-bit binary value is negative, the sign character is stored as "2DH".

The stored sign character "20H" replaces the preceding zeroes.

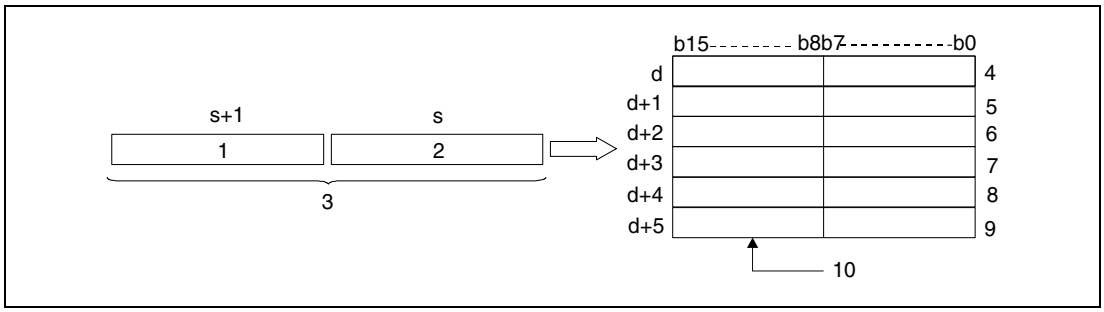
For the value 00325 for example the zeroes of the digits of tenthousands and thousands are replaced by "20H" so that only the actually required digits are stored.

The storage of data in the device specified by d+3 (Array_d[4]) depends on the status of the relay SM701:

- If the relay is not set, a zero "00H" is stored in the area d+3 (Array_d[4]).
- If the relay is set, the value in d+3 (Array_d[4]) remains unchanged.

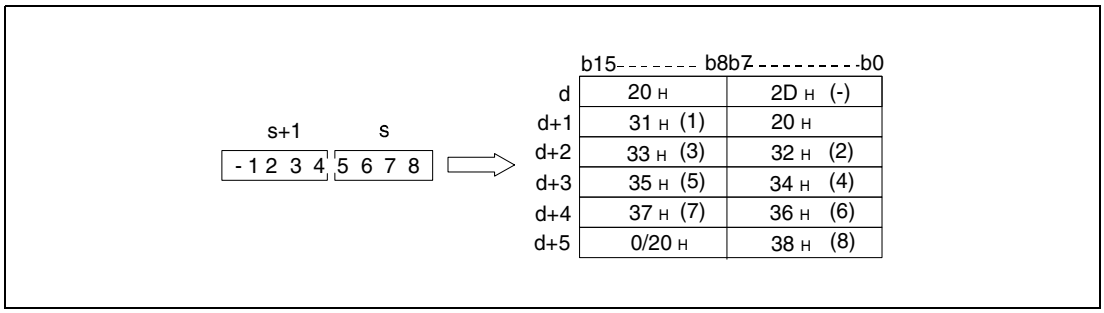
DBINDA Conversion of 32-bit binary data

The DBINDA instruction converts 32-bit binary data specified by s and s+1 into a decimal value in ASCII code and stores it in the device specified in d (Array_d[1]) through d+5 (Array_d[6]).



[1] Upper 16 bits
[2] Lower 16 bits
[3] 32-bit binary data
[4] Sign character/ digit of billions in ASCII code
[5] Digit of ten millions/ digit of one hundred millions in ASCII code
[6] Digit of one hundred thousands/ digit of millions in ASCII code
[7] Digit of thousands/ digit of ten thousands in ASCII code
[8] Digit of tens/ digit of hundreds in ASCII code
[9] 0 or 20H/ digit of ones in ASCII code
[10] With the relay SM701 not set (0)/ with the relay SM701 set (20H)

The value specified by s and s+1 is stored beginning from d (Array_d[1]) through d+5 (Array_d[6]) as decimal value in ASCII code.



The 32-bit binary value specified by s may range from -2147483648 to 2147483647.

The results of the conversion operation are stored in d (Array_d[1]) through d+5 (Array_d[6]) as follows:

– If the binary value is positive, the sign character is stored as "20H".

– If the binary value is negative, the sign character is stored as "2DH".

The stored sign character "20H" replaces the preceding zeroes.

For the value 0012034560 for example the zeroes of the digits of billions and hundred millions are replaced by "20H" so that only the actually required digits are stored.

The storage of data in the upper 8 bits of the device specified by d+5 (Array_d[6]) depends on the status of the relay SM701:

- If this relay is not set, a zero "00H" is stored in the area d+5 (Array_d[6]).
- If this relay is set, a space character (20H) is stored in the area d+5 (Array_d[6]).

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example 1

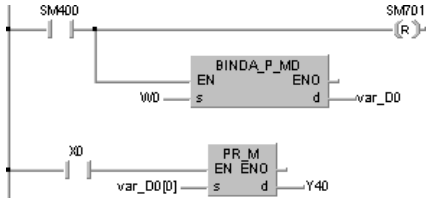
BINDAP

The following program outputs the value of the 16-bit binary data in W0 as decimal value in ASCII code via the BINDAP instruction. The PR instruction outputs the characters at Y40 through Y48.

MELSEC Instruction List

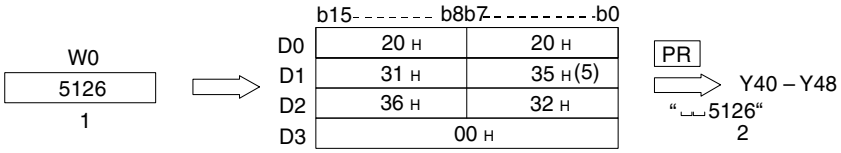
```
MELSEC   LD      SM400
         RST     SM701
         BINDAP  W0
                 D0
         LD      X0
         PR      D0
                 Y40
```

Ladder Diagram



IEC Instruction List

```
LD        SM400
R         SM701
BINDA_P_MD W0 , var_D0
LD        X0
PR_M      var_D0[0] , Y40
```



[1] Binary value

[2] Output

Program Example 2

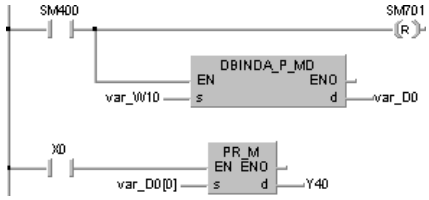
DBINDAP

The following program outputs the value of the 32-bit binary data in W10 and W11 as decimal value in ASCII code via the DBINDAP instruction. The PR instruction outputs the characters at Y40 through Y48.

MELSEC Instruction List

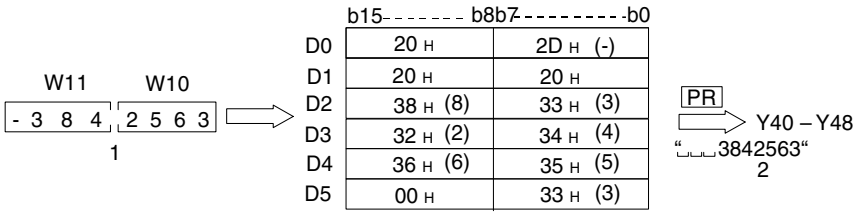
```
MELSEC   LD     SM400
         RST    SM701
         DBINDAP W10
                D0
         LD     X0
         PR     D0
                Y40
```

Ladder Diagram



IEC Instruction List

```
LD              SM400
R               SM701
DBINDA_P_MD     var_W10 , var_D0
LD              X0
PR_M            var_D0[0] , Y40
```



[1] Output

[2] Binary value

NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.11.2 BINHA, BINHAP, DBINHA, DBINHAP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

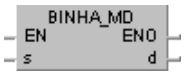
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—

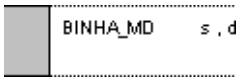
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

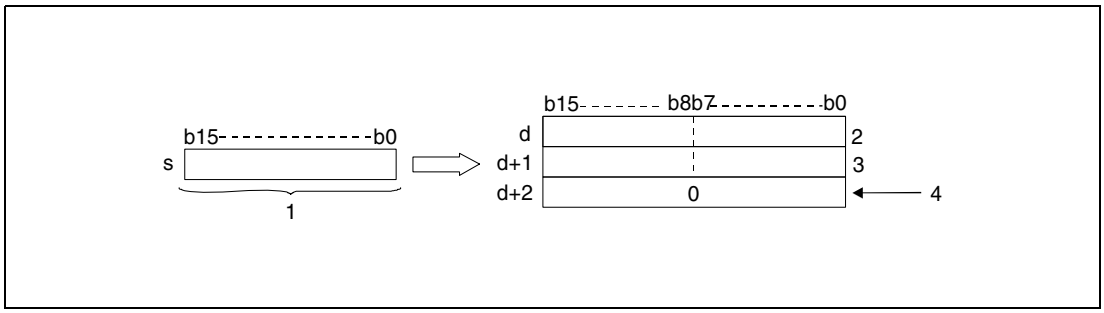
Set Data	Meaning	Data Type	
		MELSEC	IEC
s	Binary data to be converted into ASCII format	BIN 16-/32-bit	ANY16/32
d	First number of device storing the conversion result	Character string	Array [1..3]/ [1..5] of ANY16

Functions

Conversion of 16-/32-bit binary data into hexadecimal values in ASCII code

BINHA Conversion of 16-bit binary data

The BINHA instruction converts 16-bit binary data specified by s into a hexadecimal value in ASCII code and stores it in the devices specified by d (Array_d[1]) through d+2 (Array_d[3]).



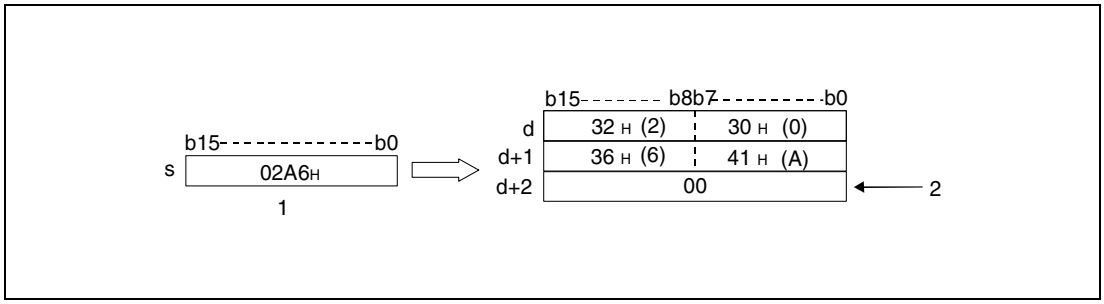
[1] 16-bit binary data

[2] ASCII code of the 3rd digit/ ASCII code of the 4th digit

[3] ASCII code of the 1st digit/ ASCII code of the 2nd digit

[4] With the relay SM701 not set

The value specified by s is stored in ASCII code in d (Array_d[1]) through d+2 (Array_d[3]).



[1] 16-bit binary data

[2] With the relay SM701 not set

The 16-bit binary data specified by s may range from 0H to FFFFH.

The conversion result is stored as 4-digit hexadecimal value in d (Array_d[1]) through d+2 (Array_d[3]).

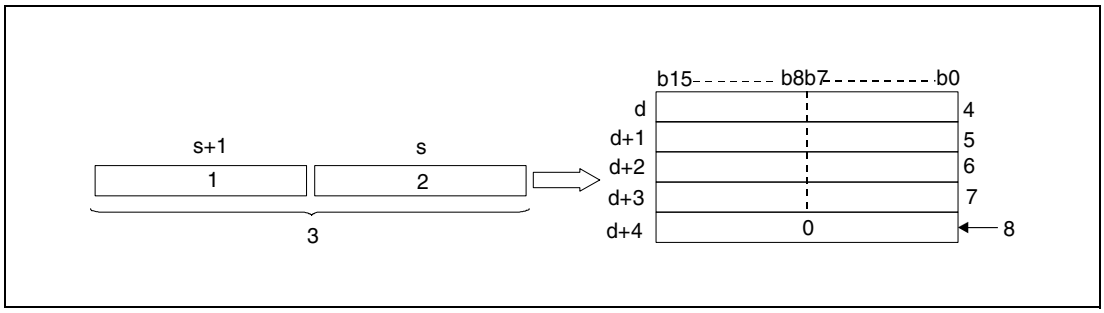
If one of the digits is 0, this digit is processed as value 0 (zeroes are not suppressed).

The storage of the data in the device specified by d+2 (Array_d[3]) depends on the status of the relay SM701 as follows:

- If this relay is not set, a zero "00H" is stored in the area d+2 (Array_d[3]).
- If this relay is set, the value in d+2 (Array_d[3]) remains unchanged.

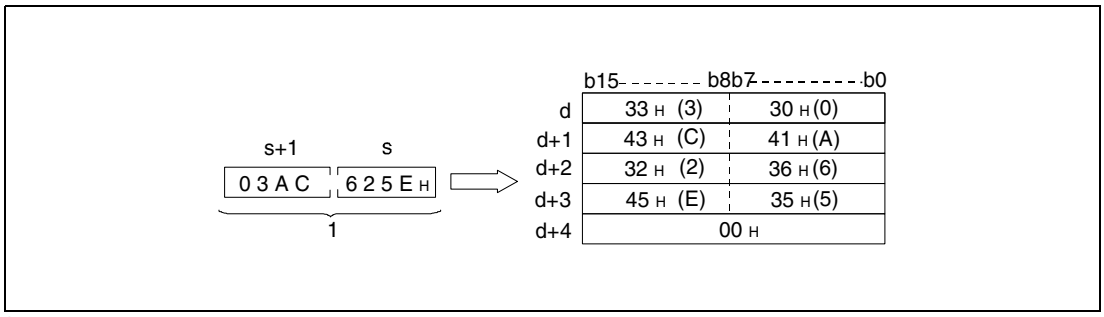
DBINHA Conversion of 32-bit binary data

The DBINHA instruction converts 32-bit binary data specified by s and s+1 into a hexadecimal value in ASCII code and stores it in the devices specified by d (Array_d[1]) through d+4 (Array_d[5]).



[1] Upper 8 bits
[2] Lower 8 bits
[3] 32-bit binary data
[4] ASCII code of the 7th digit/ ASCII code of the 8th digit
[5] ASCII code of the 5th digit/ ASCII code of the 6th digit
[6] ASCII code of the 3th digit/ ASCII code of the 4th digit
[7] ASCII code of the 1st digit/ ASCII code of the 2nd digit
[8] With the relay SM701 not set

The value "03AC625E$_H$" specified in s and s+1 is stored in d as follows:



[1] BIN 32-bit data

The 32-bit binary value specified by s and s+1 may range from 0$_H$ to FFFFFFFF$_H$.

The conversion result is stored as 8-digit hexadecimal value in d (Array_d[1]) through d+4 (Array_d[5]).

If one of the digits is 0, this digit is processed as value 0 (zeroes are not suppressed).

The storage of the data in the device specified by d+4 (Array_d[5]) depends on the status of the relay SM701 as follows:

- If this relay is not set, a zero "00$_H$" is stored in the area d+4 (Array_d[5]).
- If this relay is set, the value in d+4 (Array_d[5]) remains unchanged.

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

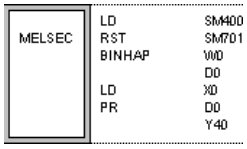
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example 1

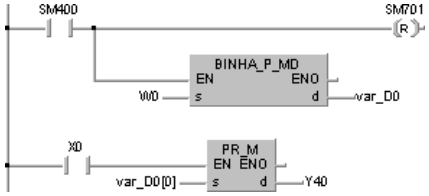
BINHAP

The following program outputs the value of the 16-bit binary data in W0 as decimal value in ASCII code via the BINHAP instruction. The PR instruction outputs the characters at Y40 through Y48.

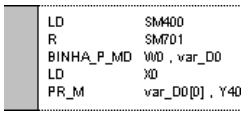
MELSEC Instruction List

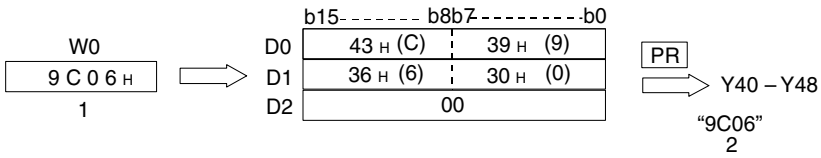


Ladder Diagram



IEC Instruction List





[1] Output

[2] Binary data

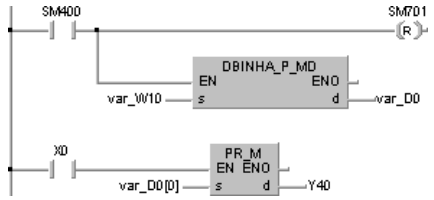
Program Example 2 DBINHAP

The following program outputs the value of the 32-bit binary data in W10 and W11 via the DBINHAP instruction as decimal value in ASCII code. The PR instruction outputs the characters at Y40 through Y48.

MELSEC Instruction List

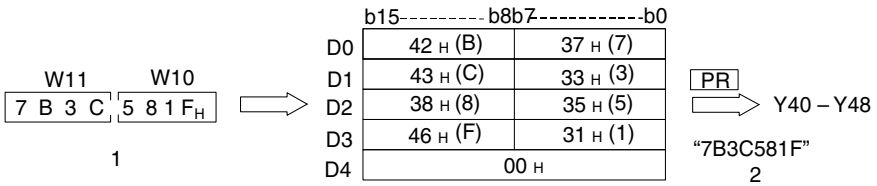
```
MELSEC   LD      SM400
         RST     SM701
         DBINHAP W10
                 D0
         LD      X0
         PR      D0
                 Y40
```

Ladder Diagram



IEC Instruction List

```
         LD              SM400
         R               SM701
         DBINHA_P_MD     var_W10 , var_D0
         LD              X0
         PR_M            var_D0[0] , Y40
```



[1] Output

[2] Binary value

NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.11.3 BCDDA, BCDDAP, DBCDDA, DBCDDAP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

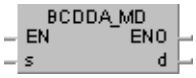
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—

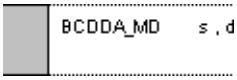
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

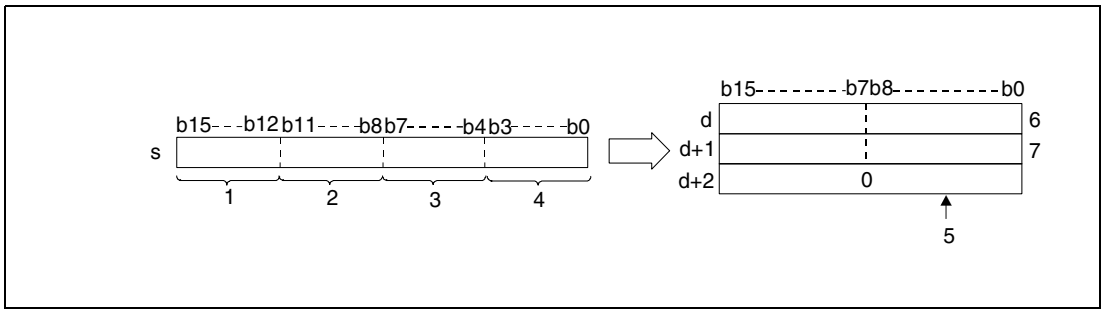
Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	BCD data to be converted into ASCII format	Word	ANY16/32
d	FIrst number of device storing the conversion result	Character string	Array [1..3]/ [1..5] of ANY16

Functions **Conversion of 4-/ 8-digit BCD data into ASCII code**

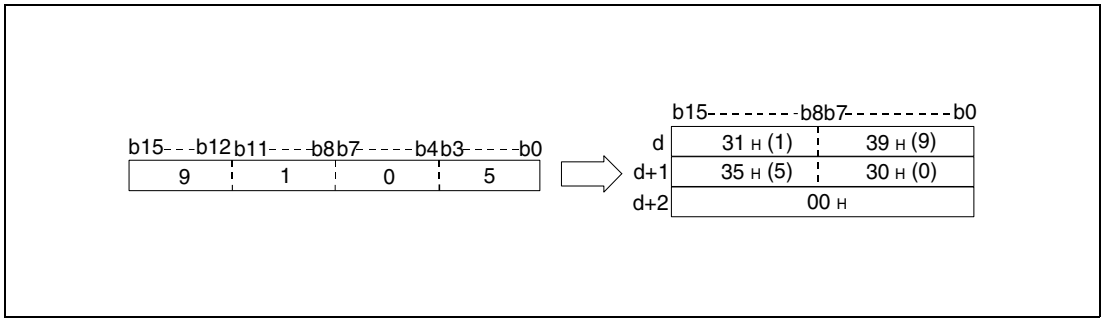
BCDDA Conversion of 4-digit BCD data

The BCDDA instruction converts 4-digit BCD data specified by s into the ASCII format and stores it in the devices specified by d (Array_d[1]) through d+2 (Array_d[3]).



[1] Digit of thousands
[2] Digit of hundreds
[3] Digit of tens
[4] Digit of ones
[5] With the relay SM701 not set
[6] ASCII code of the 3rd digit/ ASCII code of the 4th digit
[7] ASCII code of the 1st digit/ ASCII code of the 2nd digit

The value 9105 specified in s is stored in d as follows:



The BCD value specified in s may range from 0 to 9999.

The conversion result is stored in d (Array_d[1]) through d+2 (Array_d[3]).

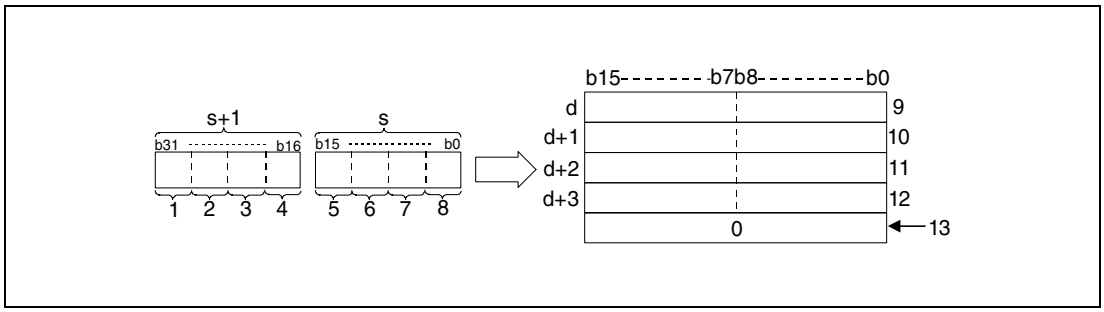
If one of the digits is 0, this digit is processed as "30H" (zeroes are not suppressed).

The storage of the data in the device specified by d+2 (Array_d[3]) depends on the status of the relay SM701 as follows:

- If this relay is not set, a zero "00H" is stored in the area d+2 (Array_d[3]).
- If this relay is set, the value in d+2 (Array_d[3]) remains unchanged.

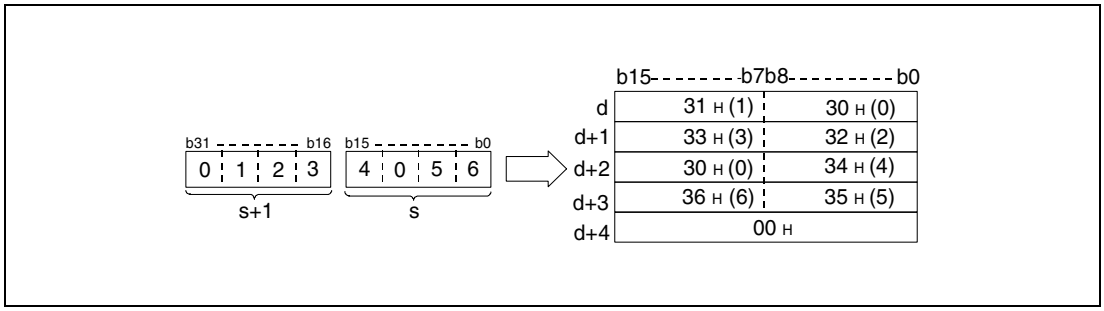
DBCDDA Conversion of 8-digit BCD data

The DBCDDA instruction converts 8-digit BCD data specified by s and s+1 into the ASCII format and stores it in the devices specified by d (Array_d[1]) through d+4 (Array_d[5]).



[1] Digit of ten millions
[2] Digit of millions
[3] Digits of hundred thousands
[4] Digit of ten thousands
[5] Digit of thousands
[6] Digit of hundreds
[7] Digit of tens
[8] Digit of ones
[9] ASCII code of the 7th digit/ ASCII code of the 8th digit
[10] ASCII code of the 5th digit/ ASCII code of the 6th digit
[11] ASCII code of the 3rd digit/ ASCII code of the 4th digit
[12] ASCII code of the 1st digit/ ASCII code of the 2nd digit
[13] With the relay SM701 not set

The value 01234056 specified in s and s+1 is stored in d as follows:



The BCD value specified by s and s+1 may range from 0 to 99999999.

The conversion result is stored in d (Array_d[1]) through d+4 (Array_d[5]).

If one of the digits is 0, this digit is processed as "30H" (zeroes are not suppressed).

The storage of the data in the device specified by d+4 (Array_d[5]) depends on the status of the relay SM701:

- If this relay is not set, a zero "00H" is stored in the area d+4 (Array_d[5]).
- If this relay is set, the value in d+4 (Array_d[5]) remains unchanged.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

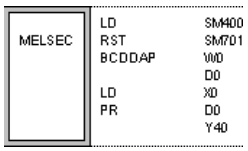
- The BCD data in s exceed the range of 0 to 9999 during the execution of the BCDDA instruction.
  (Error code 4100)
- The BCD data in s exceed the range of 0 to 99999999 during the execution of the DBCDDA instruction.
  (Error code 4100)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example 1

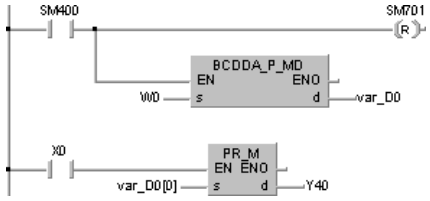
BCDDAP

The following program outputs the value of the 4-digit BCD data in W0 as decimal value in ASCII code via the BCDDAP instruction. The PR instruction outputs the characters at Y40 through Y48.

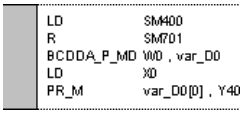
MELSEC Instruction List

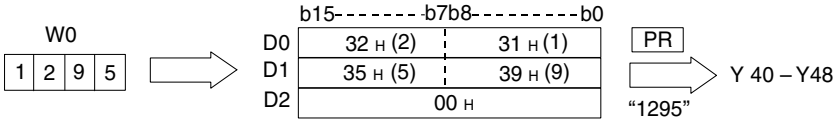


Ladder Diagram



IEC Instruction List





Program Example 2

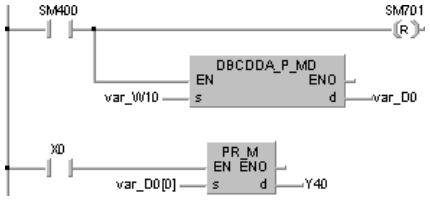
DBCDDAP

The following program outputs the value of the 8-digit BCD data in W10 and W11 as decimal value in ASCII code via the PR instruction. The PR instruction outputs the characters at Y40 through Y48.

MELSEC Instruction List

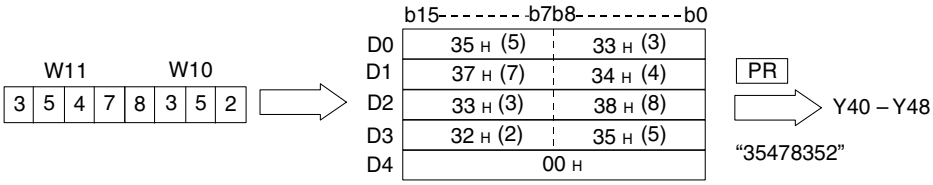
```
MELSEC   LD     SM400
         RST    SM701
         DBCDDAP W10
                D0
         LD     X0
         PR     D0
                Y40
```

Ladder Diagram



IEC Instruction List

```
LD              SM400
R               SM701
DBCDDA_P_MD     var_W10 , var_D0
LD              X0
PR_M            var_D0[0] , Y40
```



NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.11.4 DABIN, DABINP, DDABIN, DDABINP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

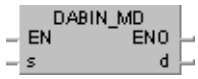
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	●	●	●	●	●	●	●	—	—

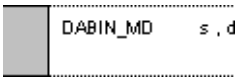
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

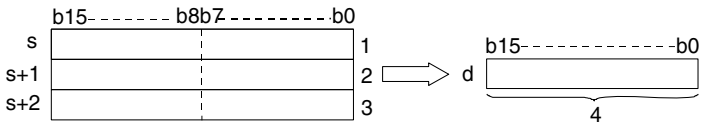
Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	Storage area storing the ASCII data to be converted	Character string	Array [1..3]/ [1..6] of ANY16
d	Storage area storing the conversion result	BIN 16-/32-bit	ANY16/32

Functions **Conversion of decimal ASCII data into BIN 16-/32-bit binary data**

DABIN Conversion of BIN 16-bit binary data

The DABIN instruction converts the decimal ASCII data specified in the area s (Array_s[1]) through s+2 (Array_s[3]) into the BIN 16-bit format and stores it in the devices specified by d.



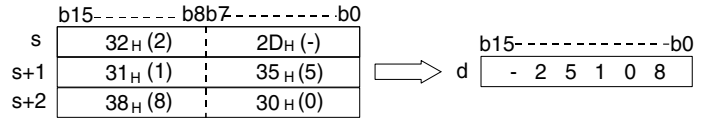
[1] ASCII code of the digit of ten thousands/ sign character

[2] ASCII code of the digit of hundreds/ ASCII code of the digit of thousands

[3] ASCII code of the digit of ones/ ASCII code of the digit of tens

[4] BIN 16-bit binary data

The value specified in the area s (Array_s[1]) through s+2 (Array_s[3]) is stored in d as -25018 as follows:



The ASCII value specified by s (Array_s[1]) through s+2 (Array_s[3]) may range from -32768 to 32767.

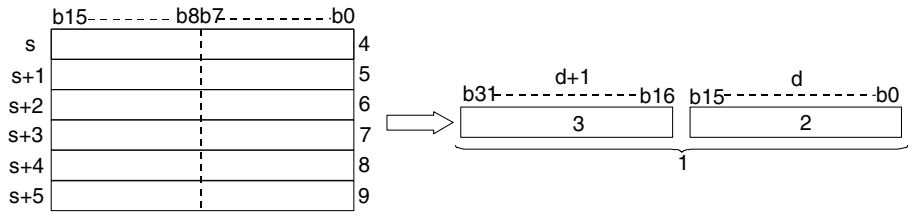
The sign character is stored as "20H" if the binary value is positive. For a negative result the sign character "2DH" is stored.

Each stored digit of the ASCII code may range from "30H" to "39H".

If a digit contains the value "20H" or "00H", this value will be overwritten automatically with the value "30H".

DDABIN Conversion into BIN 32-bit data

The DDABIN instruction converts the decimal ASCII data specified in the area s (Array_s[1]) through s+5 (Array_s[6]) into the BIN 32-bit format and stores it in the devices specified by d and d+1.



[1] BIN 32-bit binary data

[2] Lower 16-bit

[3] Upper 16-bit

[4] ASCII code of the digit of billions / sign character

[5] ASCII code of the digit of ten millions / ASCII code of the digit of hundred millions

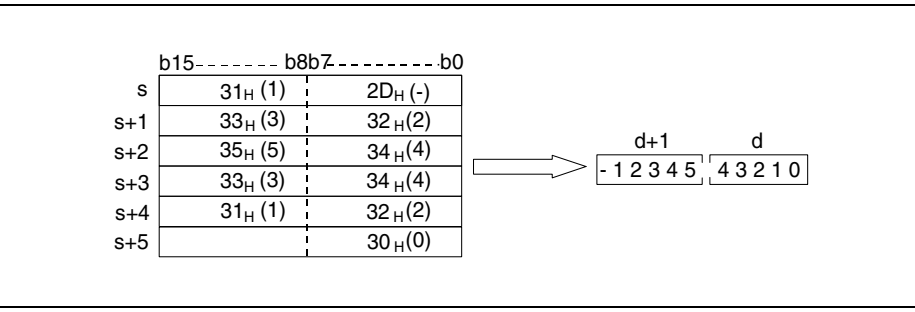
[6] ASCII code of the digit of hundred thousands / ASCII code of the digit of millions

[7] ASCII code of the digit of thousands / ASCII code of the digit of ten thousands

[8] ASCII code of the digit of tens / ASCII code of the digit of hundreds

[9] Is ignored / ASCII code of the digit of tens

The value specified in the area s (Array_s[1]) through s+5 (Array_s[6]) is stored in d as -1234543210 as follows:



The ASCII value specified in s (Array_s[1]) through s+5 (Array_s[6]) may range from -2147483648 to 2147483647.

The sign character is stored as "20H" if the binary value is positive. For a negative result the sign character "2DH" is stored.

Each stored digit of the ASCII code may range from "30H" to "39H".

If a digit contains the value "20H" or "00H", this value will be overwritten automatically with the value "30H".

Operation Errors

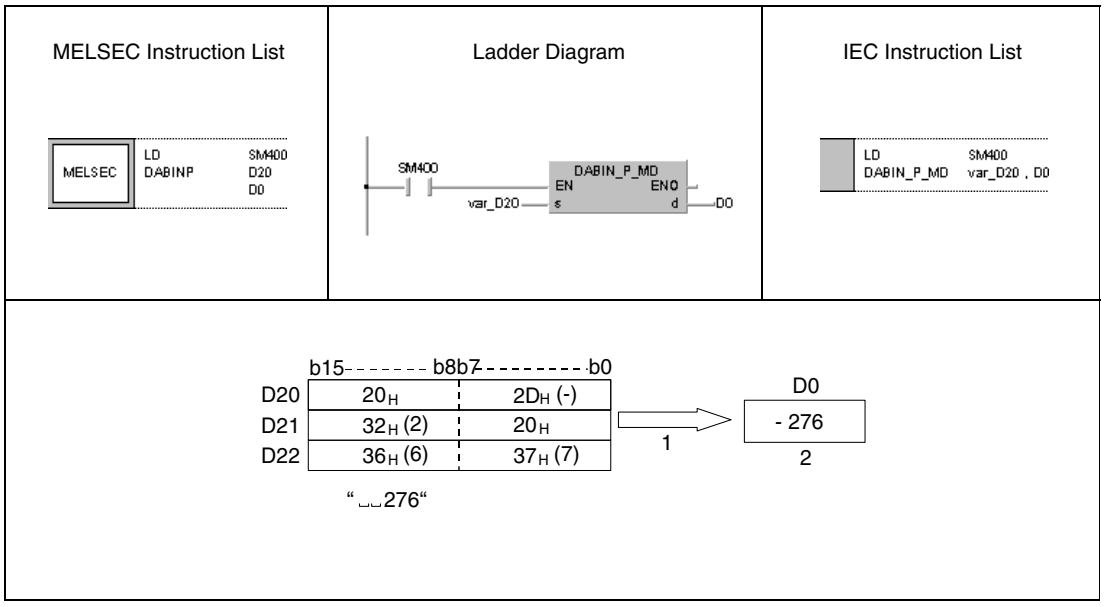
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The sign character stored in the lower 16 bits of the device s (Array_s[1]) contains a value different from "30H" to "39H, "20H" or "00H".
  (Error code 4100)
- The ASCII code stored in the area s (Array_s[1]) through s+5 (Array_s[6]) contains values different from "30H" to "39H, "20H" to "00H".
  (Error code 4100)
- The ASCII code stored in the area s (Array_s[1]) through s+5 (Array_s[6]) exceeds the following range of values:
  For the DABIN instruction -32768 to 32767
  For the DDABIN instruction -2147483648 to 2147483647.
  (Error code 4100)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example 1

DABINP

The following program converts the five-digit decimal ASCII value in D20 (var_D20 Array [0]) through D22 (var_D20 Array [2]) into a binary value and stores it in D0.



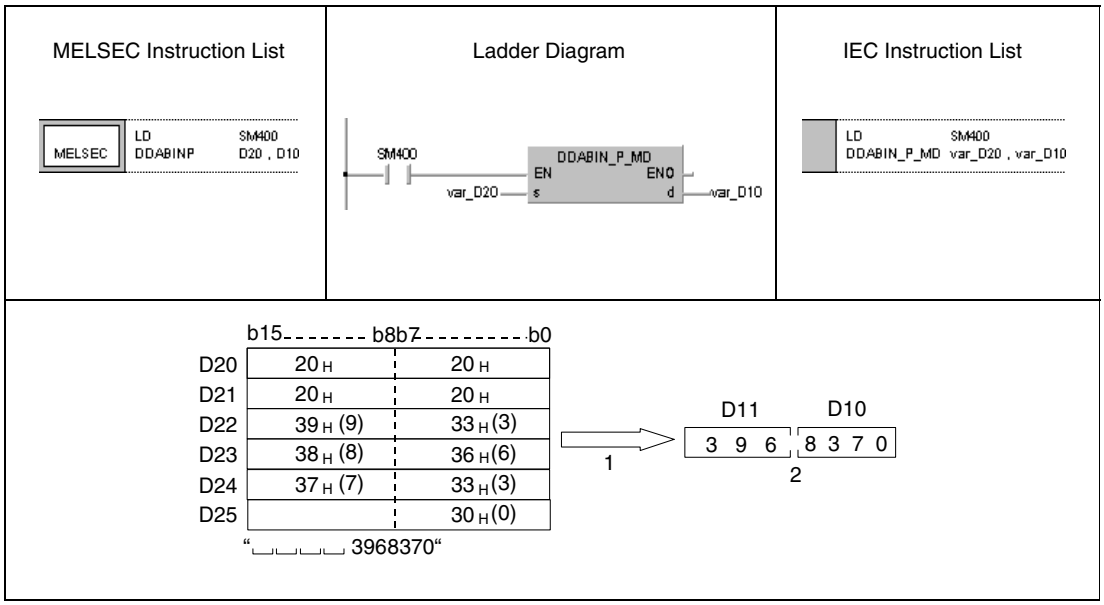
[1] Is read as -00276

[2] Binary value

Program Example 2

DDABINP

The following program converts the ten-digit decimal ASCII value in D20 (var_D20 Array [0]) through D25 (var_D20 Array [5]) into a binary value and stores it in D10 and D11.



[1] Is read as +0003968370

[2] Binary value

NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.11.5 HABIN, HABINP, DHABIN, DHABINP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

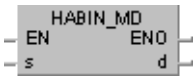
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	●	●	●	●	●	●	●	—	—

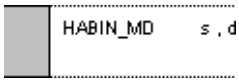
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

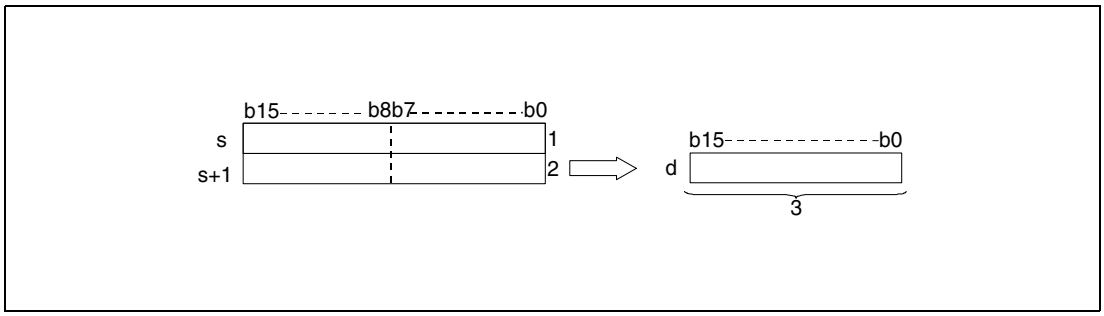
Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	Storage area storing the ASCII data to be converted	Character string	ANY32/Array [1..4] of ANY16
d	Storage area storing the conversion result	BIN 16-/32-bit	ANY16/32

Functions **Conversion of hexadecimal ASCII data into BIN 16-/32-bit binary data**

HABIN Conversion into BIN 16-bit data

The HABIN instruction converts the hexadecimal ASCII data in the device specified by s and s+1 into the BIN 16-bit binary format and stores it in the devices specified by d.

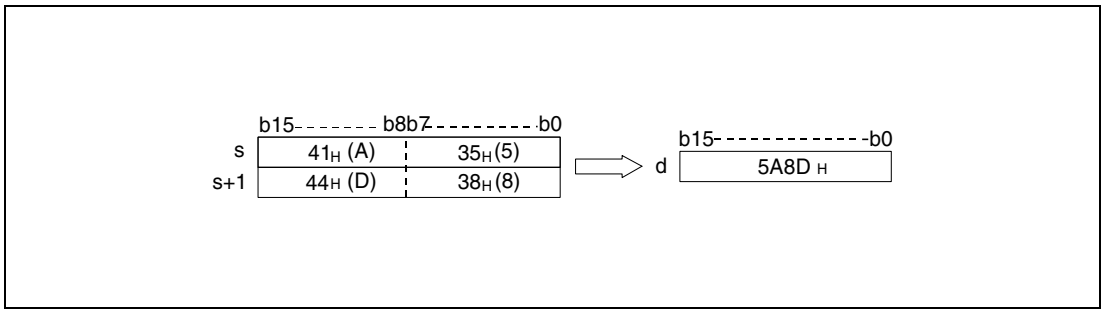


[1] ASCII code for the 3rd digit/ ASCII code for the 4th digit

[2] ASCII code for the 1st digit/ ASCII code for the 2nd digit

[3] BIN 16-bit binary data

The value "5A8D$_H$" specified in s through s+1 is stored in d after being processed as follows:

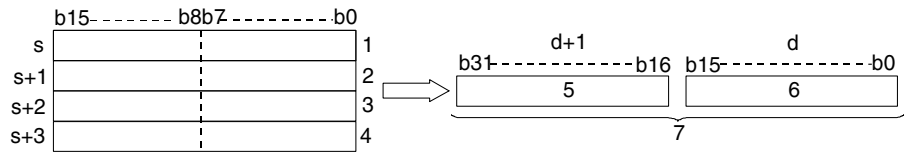


The ASCII value specifed in s through s+1 may range from 0000$_H$ to FFFF$_H$.

Each stored digit of the ASCII code may range from "30$_H$" to "39$_H$" and from "41$_H$" to "46$_H$".

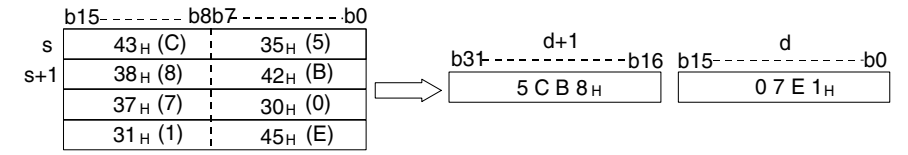
DHABIN Conversion into BIN 32-bit data

The DHABIN instruction converts the hexadecimal ASCII data specified in the area s (Array_s[1]) through s+3 (Array_s[4]) into the BIN 32-bit format and stores it in the devices specified by d and d+1.



[1] ASCII code of the 7th digit / ASCII code of the 8th digit
[2] ASCII code of the 5th digit / ASCII code of the 6th digit
[3] ASCII code of the 3rd digit / ASCII code of the 4th digit
[4] ASCII code of the 1st digit / ASCII code of the 2nd digit
[5] Upper 16 bits
[6] Lower 16 bits
[7] BIN 32-bit binary data

The value "5CB807E1" specified in s (Array_s[1]) through s+3 (Array_s[4]) is stored in d and d+1 after being processed as follows:



The ASCII value specified in s (Array_s[1]) through s+3 (Array_s[4]) may range from 00000000H and FFFFFFFFH.

Each stored digit of the ASCII code may range from "30H" to "39H" and from "41H" to "46H".

Operation Errors

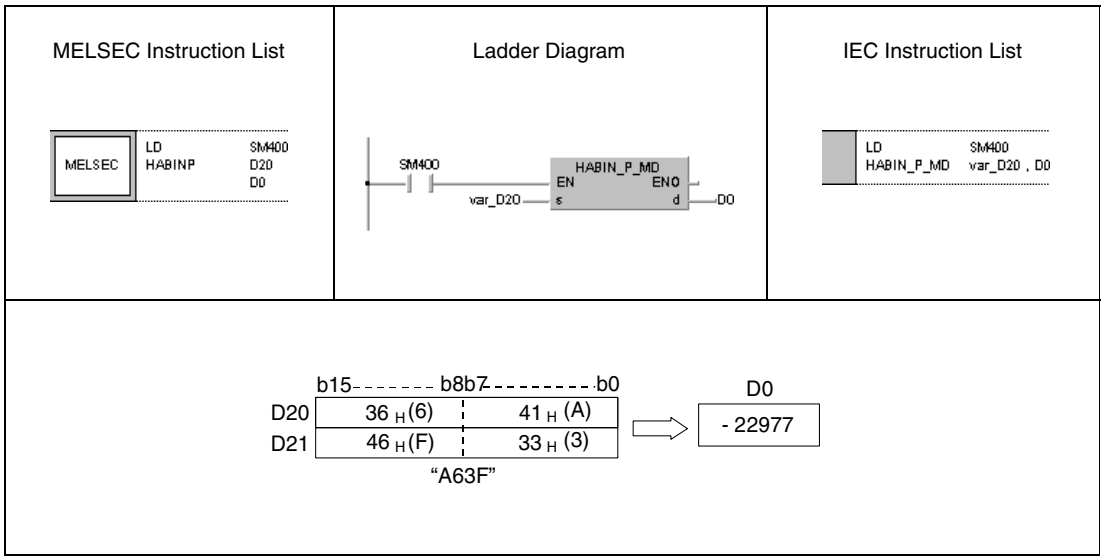
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The ASCII code stored in the area s (Array_s[1]) through s+3 (Array_s[4]) exceeds the relevant range of "30H" to "39H" and from "41H" to "46H".
  (Error code 4100)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example 1

HABINP

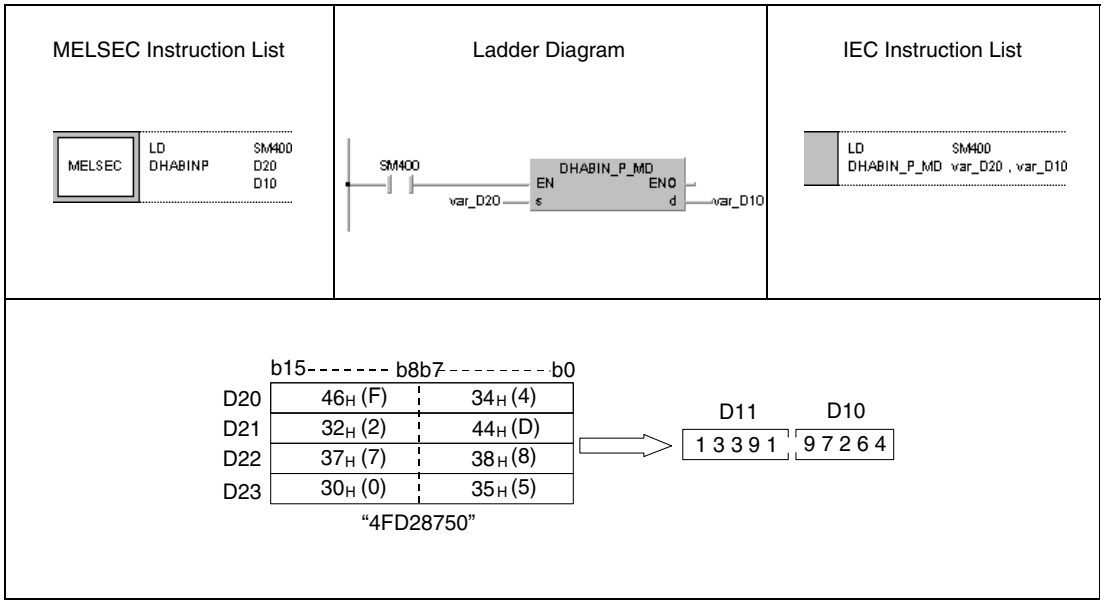
The following program converts the 4-digit ASCII value in D20 (var_D20 Array [0]) through D21 (var_D20 Array [1]) into a binary value and stores it in D0.



Program Example 2

DHABINP

The following program converts the 8-digit ASCII value in D20 (var_D20 Array [0]) through D23 (var_D20 Array [3]) into a binary value and stores it in D10 and D11.



NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.11.6 DABCD, DABCDP, DDABCD, DDABCDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

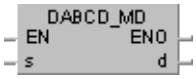
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	●	●	●	●	●	●	●	—	—

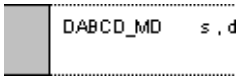
GX IEC Developer

MELSEC Instruction List

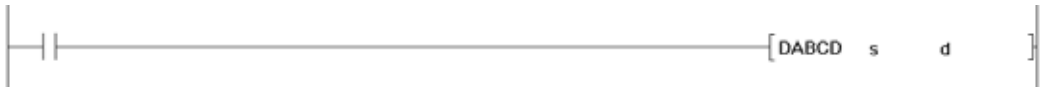
Ladder Diagram



IEC Instruction List



GX Works2



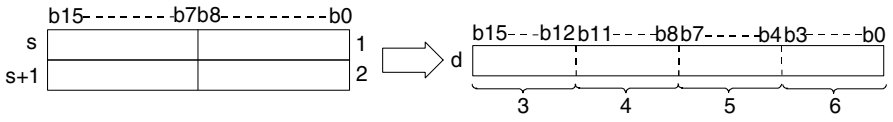
Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	ASCII data to be converted or first number of the devices where the ASCII data is stored	Character string	ANY32/ Array [1..4] of ANY16
d	Storage area storing the conversion result	4-/8-digit BCD data	ANY16/ 32

Functions **Conversion of decimal ASCII data into 4-/8-digit BCD data**

DABCD Conversion into 4-digit BCD data

The DABCD instruction converts the decimal ASCII data in s and s+1 into the 4-digit BCD data format and stores it in the devices specified by d.



[1] ASCII code of the digit of hundreds / ASCII code of the digit of thousands

[2] ASCII code of the digit of ones / ASCII code of the digit of tens

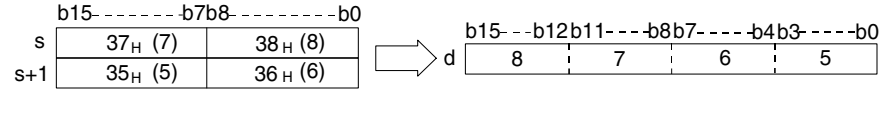
[3] Digit of thousands

[4] Digit of hundreds

[5] Digit of tens

[6] Digit of ones

The value 8765 specified in s and s+1 is stored in d as follows:



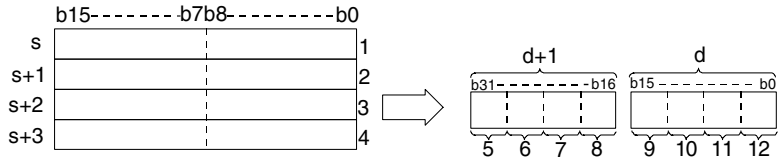
The ASCII value specified in s through s+1 may range from 0 to 9999.

Each stored digit of the ASCII code may range from "30H" to "39H".

If a digit contains the value "20H" or "00H", this value will be overwritten automatically with the value "30H".

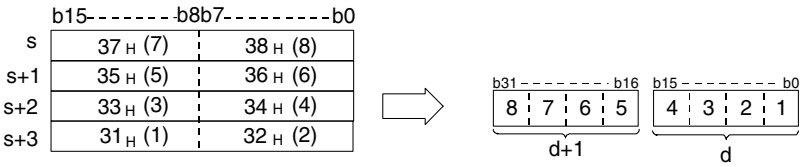
DDABCD Conversion into 8-digit BCD data

The DDABCD instruction converts the ASCII data specified in the area s (Array_s[1]) through s+3 (Array_s[4]) into the 8-digit BCD format and stores it in the devices specified in d and d+1.



[1] ASCII code of the digit of millions / ASCII code of the digit of ten millions
[2] ASCII code of the digit of ten thousands / ASCII code of the digit of hundred thousands
[3] ASCII code of the digit of hundreds / ASCII code of the digit of thousands
[4] ASCII code of the digit of ones / ASCII code of the digit of tens
[5] Digit of ten millions
[6] Digit of millions
[7] Digit of hundred thousands
[8] Digit of ten thousands
[9] Digit of thousands
[10] Digit of hundreds
[11] Digit of tens
[12] Digit of ones

The value 87654321 specified in s (Array_s[1]) through s+3 (Array_s[4]) is stored in d and d+1 as follows:



The ASCII value specified in s (Array_s[1]) through s+3 (Array_s[4]) may range from 0 to 99999999.

Each stored digit of the ASCII code may range from "30H" to "39H".

If a digit contains the value "20H" or "00H", this value will be overwritten automatically with the value "30H".

Operation Errors

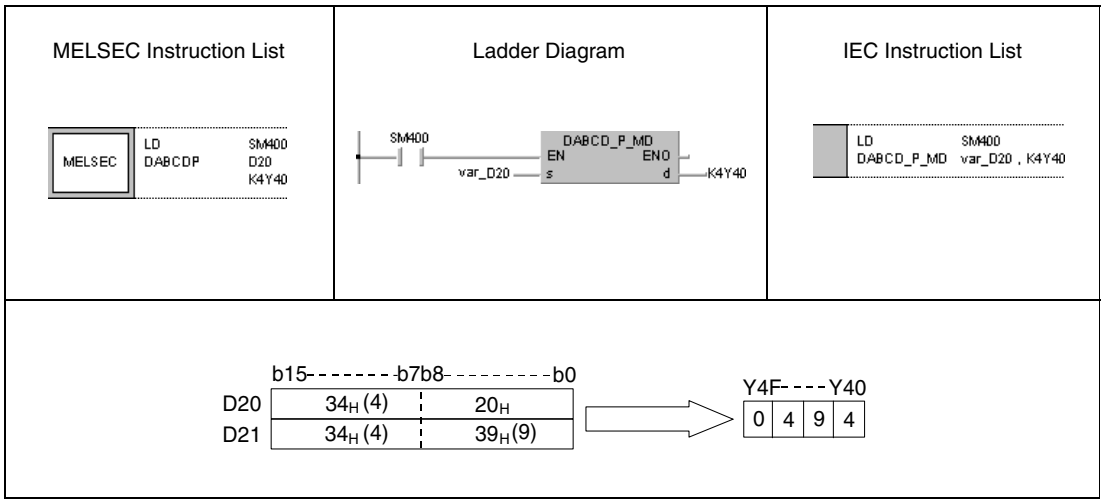
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- There are characters within the data at s (Array_s[1]) to s+3 (Array_s[4]) that are outside the range from "30_H" to "39_H".
  (Error code 4100)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example 1

DABCDP

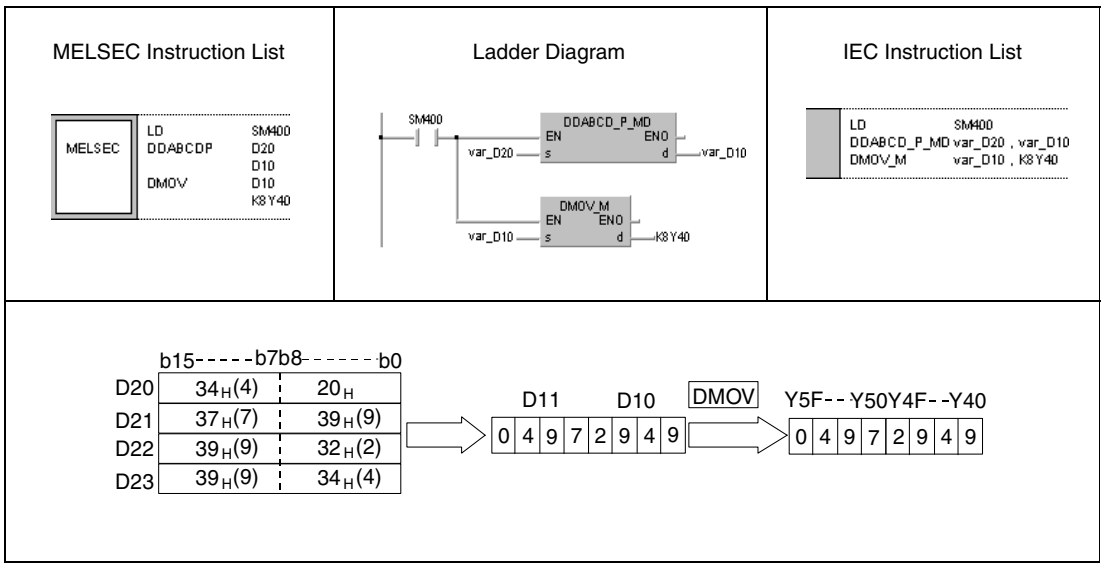
The following program converts the ASCII value in D20 (var_D20 Array [0]) through D21 (var_D20 Array [1]) into a 4-digit BCD value and outputs it at Y40 through Y4F.



Program Example 2

DDABCDP

The following program converts the ASCII value in D20 (var_D20 [0]) through D23 (var_D20 [3]) into an 8-digit BCD value, stores the result in D10 and D11, and outputs it at Y40 through Y5F.



7.11.7 COMRD, COMRDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

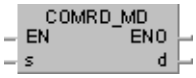
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				BL\S, BL\TR, BL, P, I, J, U
s	●	●	●	●	●	●	—	—	●
d	—	●	●	—	—	—	—	—	—

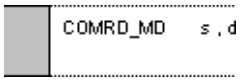
GX IEC Developer

MELSEC Instruction List

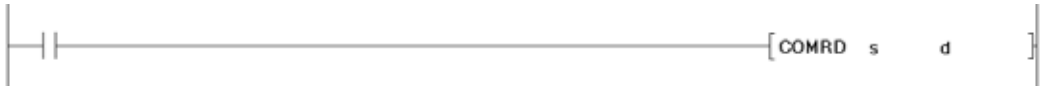
Ladder Diagram



IEC Instruction List



GX Works2



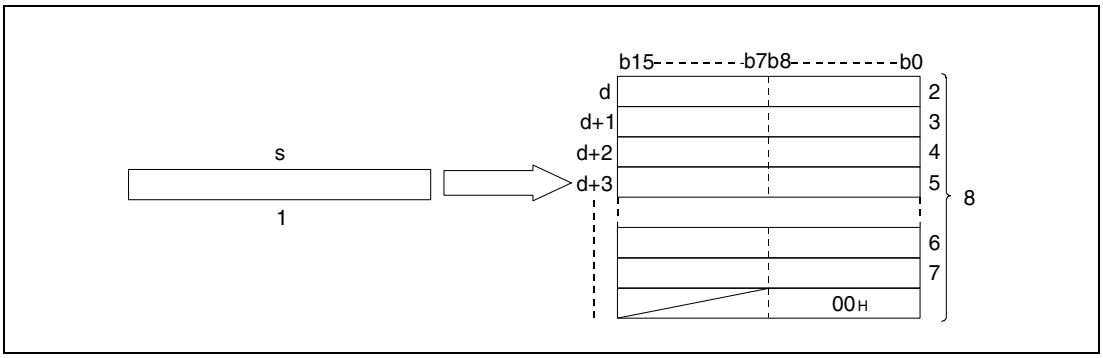
Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	First number of device storing comment data to be read	Device number	ANY16
d	First number of device to store read comment data	Character string	Array [1..8] of ANY16

Functions **Reading device comment data**

COMRD Read instruction

The COMRD instruction reads comment data from the device specified by s and stores it as ASCII code in the area d (Array_d[1]) through d+7 (Array_d[8]).



[1] Comment data

[2] ASCII code of the 2nd character / ASCII code of the 1st character

[3] ASCII code of the 4th character / ASCII code of the 3rd character

[4] ASCII code of the 6th character / ASCII code of the 5th character

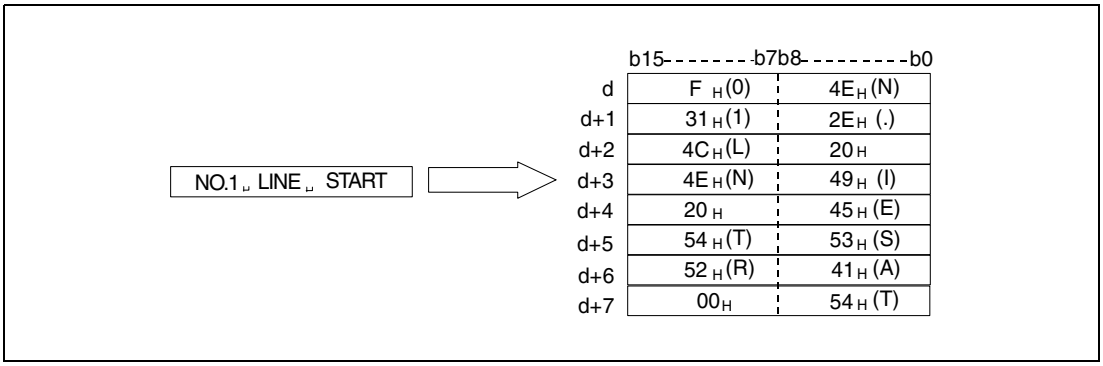
[5] ASCII code of the 8th character / ASCII code of the 7th character

[6] ASCII code of the 30th character / ASCII code of the 29th character

[7] ASCII code of the 32th character / ASCII code of the 31th character

[8] Stores at maximum 32 characters.

For example, the comment data stored in s with the character string "NO.1 LINE START" will be stored from d (Array_d[1]) on, as follows:



The address area of the devices specified by s must be located within the address area for comment data.

If no comment is specified by s, the characters are converted into blank characters.

A comment must not exceed the maximum length of 32 characters.

The content of the byte following the last character depends on the status of the special relay SM701 as follows:

- If SM701 is not set, a zero is stored
- If SM701 is set, no changes are made.

SM720 is set for one scan after the execution of the COMRD instruction has been finished. SM721 is ON during the execution of the COMRD instruction. If SM721 is already set, when the COMRD instruction is started, no processing will be performed.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The comment is not registered to the device number specified by s.
  (Error code 4100)
- The device number specified by d is not a word device.
  (Error code 4101)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

NOTE

The device comment used in the COMRD(P) instruction uses a comment file stored in a memory card and the standard ROM. Comment files stored in the program memory cannot be used.

Set the comment file used for the COMRD(P) instruction in "PLC file setting" in the PLC parameter dialog box. If the comment file to be used is not set in the PLC file setting, device comments cannot be output with the COMRD(P) instruction.
When a comment file is set in the "PLC File" tab of the PLC Parameter dialog box, but the file does not exist at power-on or reset, "FILESET ERROR" (error code: 2400) will occur.

The COMRD(P) instruction cannot be executed during the interrupt program. No operation if executed.

The processing of the COMRD (P) completes after several scans.

Two or more file comments cannot be accessed simultaneously.

The starting signal (command) of the COMRD(P) instruction is disabled when it is turned ON before an other COMRD(P) instruction is completed (while SM721 is ON). Execute the COMRD(P)/PRC instruction when SM721 is OFF.

The following instructions cannot be executed simultaneously because they use SM721 in common:

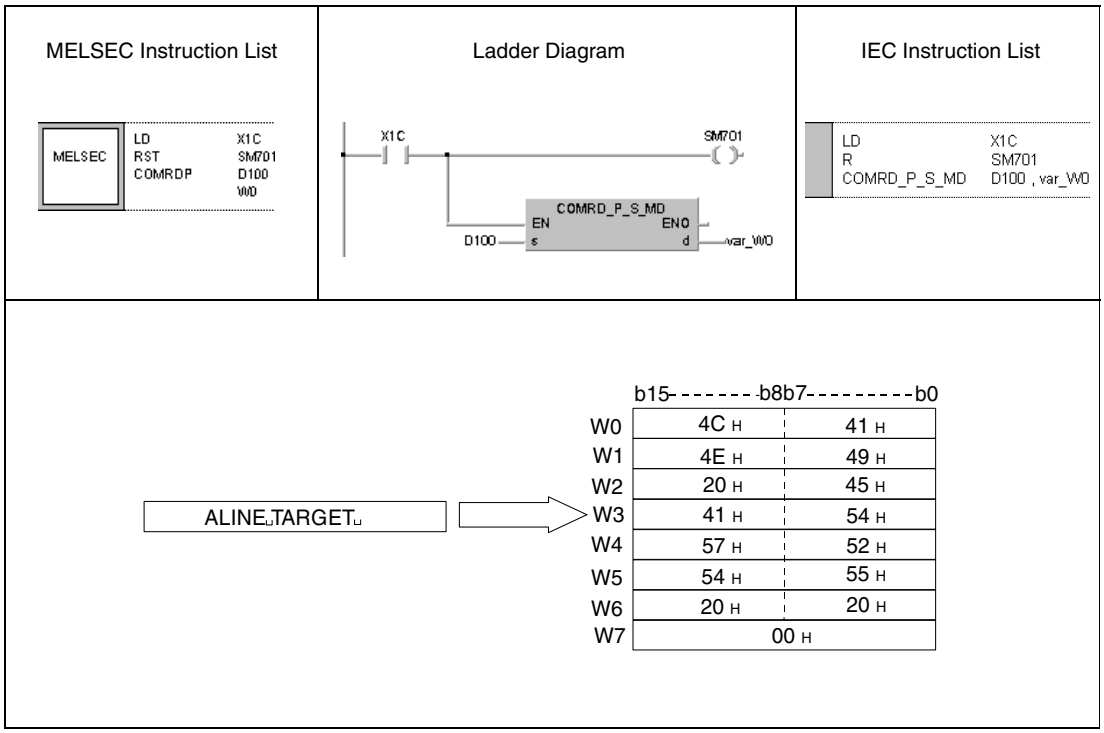
Instruction	ON during execution	ON for one scan after the execution of the instruction is complete	ON after the execution of the instruction is complete with error
SP.FREAD SP.FWRITE	SM721	Bit designated by instruction	Bit designated by instruction + next Bit
PRC COMRD		SM720	—

For the LCPU, when a comment file stored on an SD memory card is used, this instruction cannot be executed while SM606 (SD memory card forced disable instruction) is ON. Even if the instruction is attempted to be executed, the command will be ignored.

Program Example

COMRDP

With leading edge from X1C, the following program stores a comment specified in D100, as ASCII code in W0 (var_W0 Array [0]) through W7 (var_W0 Array [7]).



NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.11.8 LEN, LENP

CPU

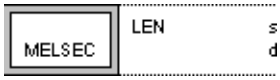
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

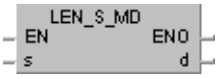
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer

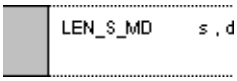
MELSEC Instruction List



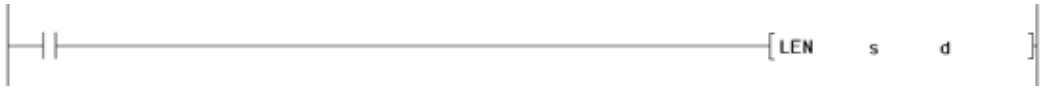
Ladder Diagram



IEC Instruction List



GX Works2



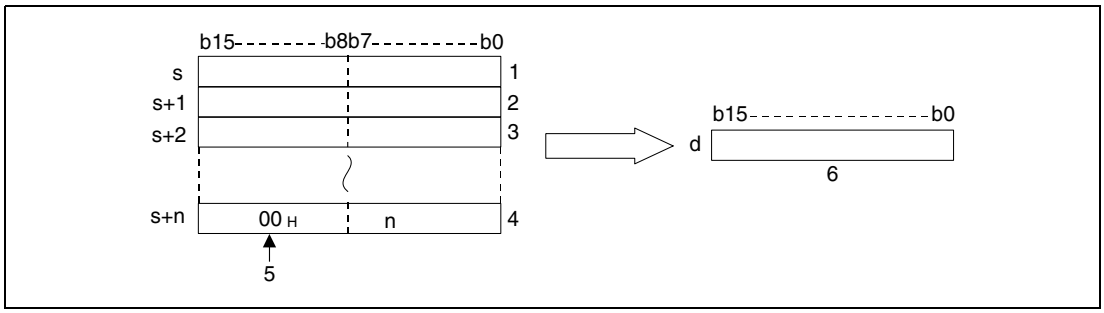
Variables

Set Data	Meaning	Data Type
s	Character string or first number of device storing a character string of which the length is to be detected	Character string
d	Address area storing the detected length of the character string	BIN 16-bit

Functions **Detecting the length of character strings**

LEN Length detection

The length instruction detects the length of a character string specified in s and stores the result in the device specified by d.



[1] 2nd character / 1st character

[2] 4th character / 3rd character

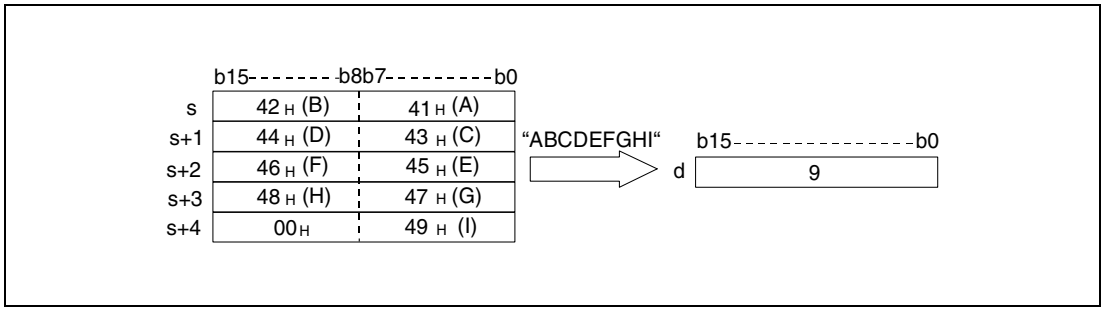
[3] 6th character / 5th character

[4] nth character

[5] End of character string

[6] Length of character string

For example, the character string "ABCDEFGHI" stored in s is stored in d as "9" as follows:



The character string stored in s is being processed until the character code "00H" is read.

The result is stored in d.

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

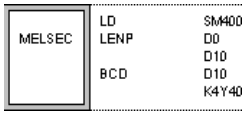
- The character code "00H" is missing in the last byte in s.
  (Error code 4101)

Program Example

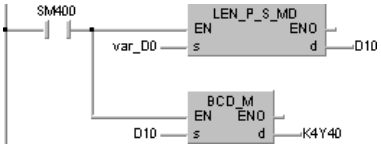
LENP

The following program processes the character string stored in D0, detects its length and outputs the character string as 4-digit BCD data at Y40 through Y4F.

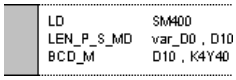
MELSEC Instruction List

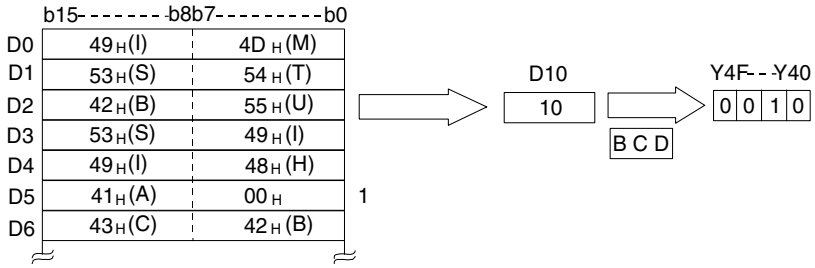


Ladder Diagram



IEC Instruction List





[1] Characters following the character code "00_H" are omitted
(only the length of the character string "MITSUBISHI" is detected)

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.11.9 STR, STRP, DSTR, DSTRP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
● [1)]	●	●	●	●	●

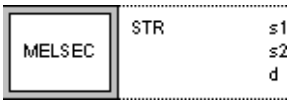
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

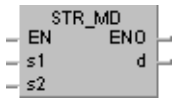
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	—	—
s2	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

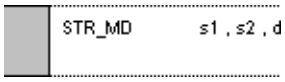
MELSEC Instruction List



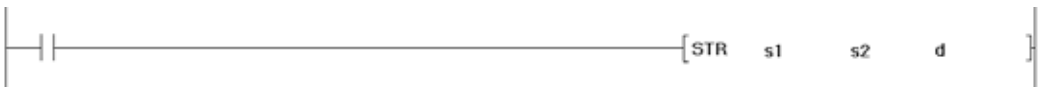
Ladder Diagram



IEC Instruction List



GX Works2



Variables

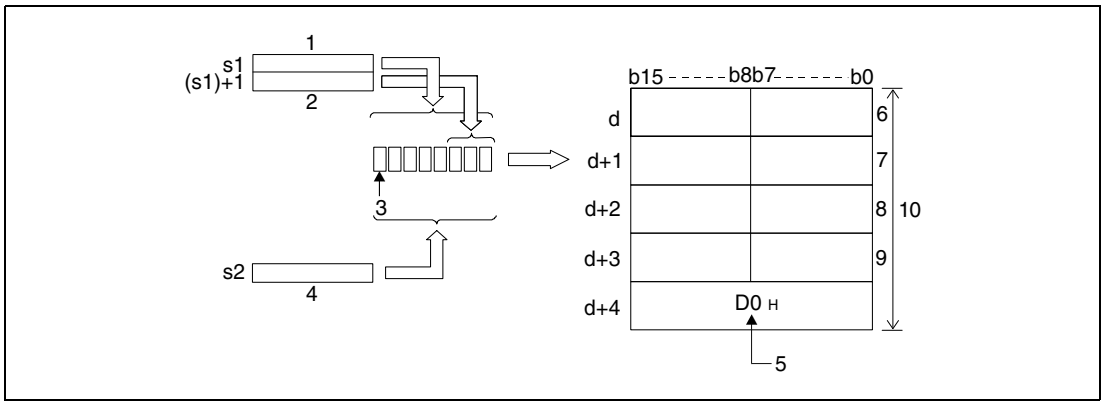
Set Data	Meaning	Data Type	
		MELSEC	IEC
s1	First number of device storing the number of digits of the numerical value to be converted	BIN 16-bit	ANY32
s2	Binary data to be converted	BIN 16-/32-bit	ANY16/32
d	First number of device storing the converted character string	Character string	Array [1..5]/ [1..6] of ANY16

Functions

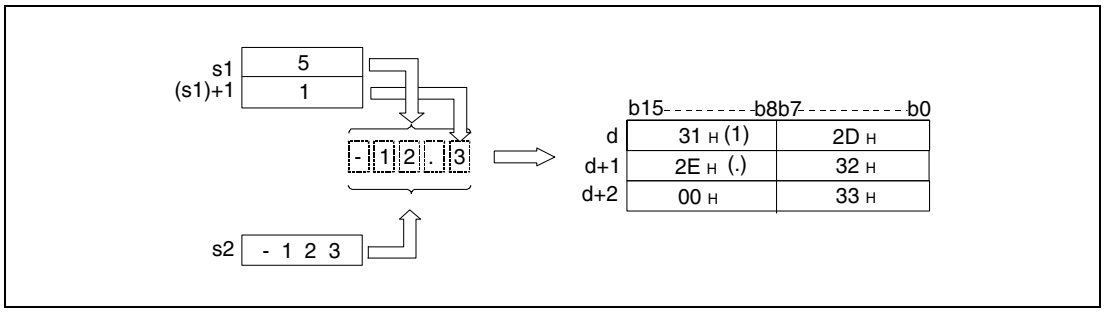
Conversion of BIN 16-/32-bit binary data into character strings

STR Conversion of BIN 16-bit binary data

The STR instruction adds a decimal point to the BIN 16-bit binary value in the device specified by s2 to the digit specified by the devices s1 and (s1)+1, converts the data into a character string, and stores it in the area of the devices specified by d (Array_d[1]) through d+4 (Array_d[5]).



[1] Total of all digits
[2] Decimal places
[3] Sign
[4] Binary value
[5] End of character string indication, automatically placed.
[6] ASCII code for the (total number of digits -1)th digit / ASCII code of the sign
[7] ASCII code for the (total number of digits -3)th digit / ASCII code for the (total number of digits -2)th digit
[8] ASCII code for the (total number of digits -5)th digit / ASCII code for the (total number of digits -4)th digit
[9] ASCII code for the (total number of digits -7)th digit / ASCII code for the (total number of digits -6)th digit
[10] Total of all digits



The number of digits that can be stored in the device specified by s1 ranges from 2 to 8.

The number of decimal places that can be stored in the devices specified by (s1)+1 ranges from 0 to 5 and must not exceed the number of digits minus 3.

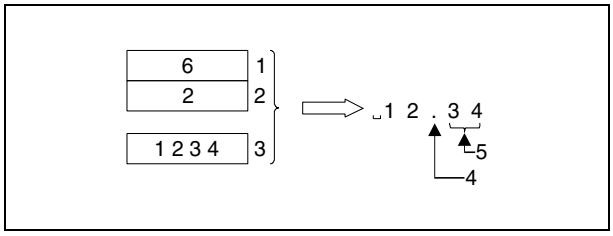
The BIN 16-bit data that can be stored in the device specified by s2 must range from -32768 to 32767.

After the conversion into a character string, the string is stored in the devices specified by d (Array_d[1]) through d+4 (Array_d[5]) as follows:

– A positive sign of the binary data is stored as ASCII character "20H" (blank).

– A negative sign of the binary data is stored as ASCII character "2DH" ("minus"- character).

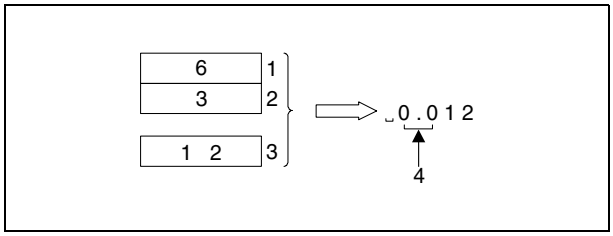
If the number of decimal places is greater than zero, the decimal point "2E$_H$" (.) is placed automatically before the first digit specified.



[1]Total of all digits
[2]Number of decimal places
[3]Binary value
[4]Decimal point placed automatically
[5]Decimal places

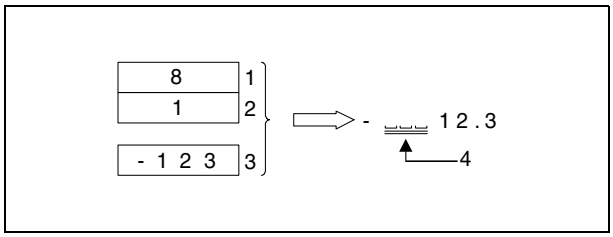
If the number of decimal places equals zero, the decimal point character "2D$_H$" (.) is not placed.

If the number of decimal places is greater than the number of digits of the binary value, the missing digits are replaced by zeroes, the binary value is shifted to the right, and the decimal point is placed accordingly (0.□□□□□).



[1]Total of all digits
[2]Number of decimal places
[3]Binary value
[4]Zeroes and decimal point placed automatically

If the number of digits, sign and decimal point included, is greater than the number of digits in the binary value, the missing digits between sign and numerical value are replaced by "20$_H$" (blanks) automatically.

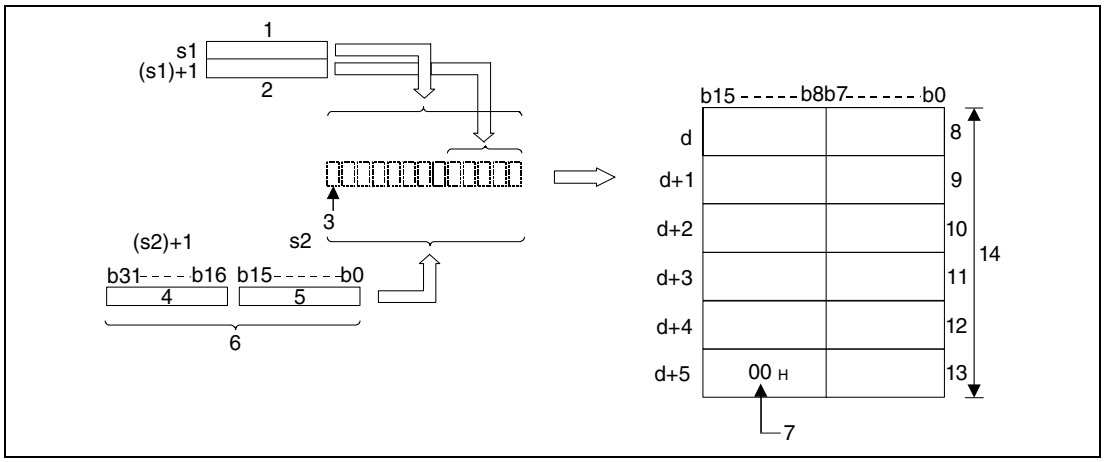


[1]Total of all digits
[2]Number of decimal places
[3]Binary value
[4]Blank characters placed automatically.

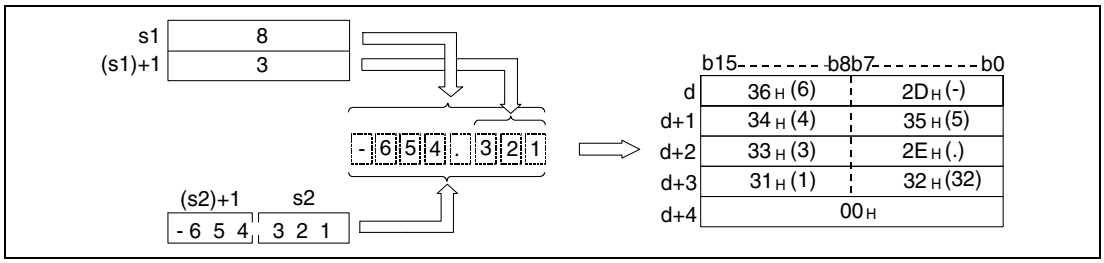
At the end of the converted character string the character code "00$_H$" is stored automatically.

DSTR Conversion of BIN 32-bit data

The DSTR instruction adds a decimal point to the BIN 32-bit binary value in the device specified by s2 and (s2)+1 to the digit specified by the devices s1 and (s1)+1, converts the data into a character string, and stores it in the area of the devices specified by d (Array_d[1]) through d+5 (Array_d[6]).



[1] Total of all digits
[2] Decimal places
[3] Sign
[4] Upper 16 Bit
[5] Lower 16 Bit
[6] Binary value
[7] End of character string indication, automatically placed.
[8] ASCII code for the (total number of digits -1)th digit / ASCII code of the sign
[9] ASCII code for the (total number of digits -3)th digit / ASCII code for the (total number of digits -2)th digit
[10] ASCII code for the (total number of digits -5)th digit / ASCII code for the (total number of digits -4)th digit
[11] ASCII code for the (total number of digits -7)th digit / ASCII code for the (total number of digits -6)th digit
[12] ASCII code for the (total number of digits -9)th digit / ASCII code for the (total number of digits -8)th digit
[13] End of character string indication / ASCII code for the (total number of digits -10)th digit
[14] Total of all digits



The number of digits that can be stored in the device specified by s1 ranges from 2 to 13.

The number of decimal places that can be stored in the devices specified by (s1)+1 ranges from 0 to 10 and must not exceed the number of digits minus 3.

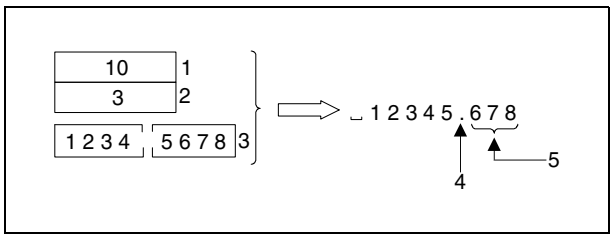
The BIN 32-bit data that can be stored in the device specified by s2 and (s2)+1 must range from -2147483648 and 32147483647.

After the conversion into a character string, the string is stored in the devices specified by d (Array_d[1]) to d+5 (Array_d[6]) as follows:

– A positive sign of the binary data is stored as ASCII character "20H" (blank).

– A negative sign of the binary data is stored as ASCII character "2DH" ("minus"- character).

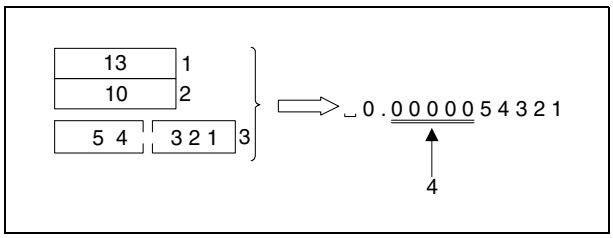
If the number of decimal places is greater than zero, the decimal point "2EH" (.) is placed automatically before the first digit specified.



[1]Total of all digits
[2]Number of decimal places
[3]Binary value
[4]Decimal point placed automatically
[5]Decimal places

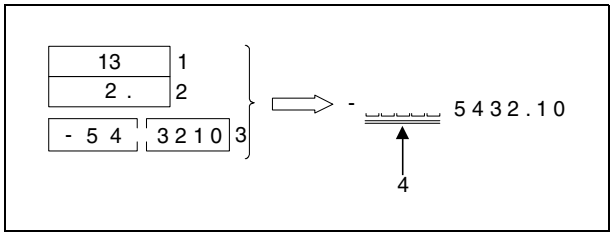
If the number of decimal places equals zero, the decimal point character "2DH" (.) is not placed.

If the number of decimal places is greater than the number of digits of the binary value, the missing digits are replaced by zeroes, the binary value is shifted to the right, and the decimal point is placed accordingly (0.□□□□).



[1]Total of all digits
[2]Decimal places
[3]Binary value
[4]Zeroes and decimal point placed automatically

If the number of digits, sign and decimal point included, is greater than the number of digits in the binary value, the missing digits between sign and numerical value are replaced by "20H" (blanks) automatically.



[1]Total of all digits
[2]Number of decimal places
[3]Binary value
[4]Blank characters placed automatically.

At the end of the converted character string the character code "00H" is stored automatically.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of digits stored in s1 exceeds the range of values specified below:
  (Error code 4100)
  Range of values for the STR instruction: ..... 2 to 8
  Range of values for the DSTR instruction: ..... 2 to 13
- The number of decimal places stored in (s1)+1 exceeds the range of values specified below:
  (Error code 4100)
  Range of values for the STR instruction: ..... 0 to 5
  Range of values for the DSTR instruction: ..... 0 to 10
- The values stored in s1 and (s1)+1 do not correspond to the following relation:
  The total of all digits minus 3 is greater than or equal to the number of decimal places.
  (Error code 4100)
- The number of digits stored in s1 and (s1)+1 is less than the digits of the binary values in s2 and (s2)+1.
  (Error code 4100)
- The area storing the character string specified from d (Array_d[1]) onwards exceeds the relevant device range.
  (Error code 4100)

Program Example 1

STRP

With leading edge from X0, the following program converts the binary value specified by D10 corresponding to the number of digits specified in D0 and D1. The result is stored in the area from D20 (var_D20 Array [1]) through D23 (var_D20 Array [4]).

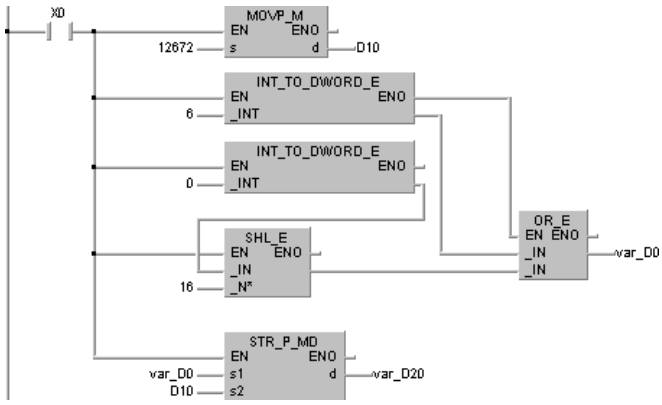
MELSEC Instruction List

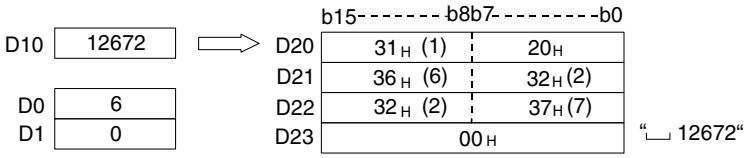
```
MELSEC
LD      X0
MOVP    K12672
        D10
MOVP    K6
        D0
MOVP    K0
        D1
STRP    D0
        D10
        D20
```

IEC Instruction List

```
LD          X0
MOVP_M      12672 , D10
INT_TO_DWORD_E  6 , var_Merker
INT_TO_DWORD_E  0 , var_D0
SHL_E       var_D0 , K16 , var_D0
LD          var_Merker
OR          var_D0
ST          var_D0
LD          X0
STR_P_MD    var_D0 , D10 , var_D20
```

Ladder Diagram





Program Example 2

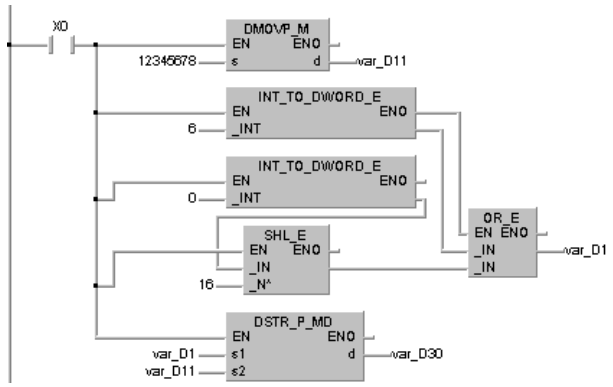
DSTRP

With leading edge from X0, the following program converts the binary value specified in D10 and D11corresponding to the number of digits specified in D0 and D1. The result is stored in the area from D20 (var_D20 Array [1]) through D26 (var_D20 Array [7]).

MELSEC Instruction List

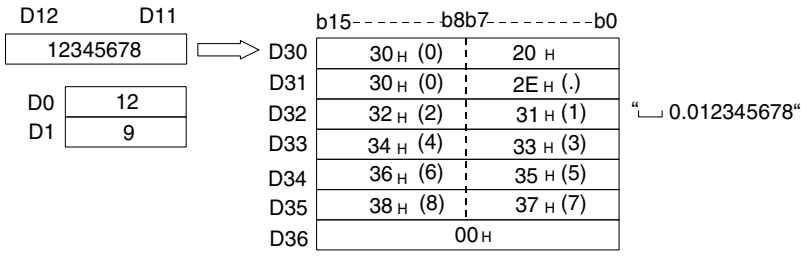
```
MELSEC   LD      X0
         DMOVP   K12345678
                 D11
         MOVP    K6
                 D2
         MOVP    K0
                 D3
         DSTRP   D2
                 D11
                 D30
```

Ladder Diagram



IEC Instruction List

```
LD              X0
DMOVP_M         12345678 , var_D11
INT_TO_DWORD_E  6 , var_Merker
INT_TO_DWORD_E  0 , var_D1
SHL_E           var_D1 , K16 , var_D1
LD              var_Merker
OR              var_D1
ST              var_D1
LD              X0
DSTR_P_MD       var_D1 , var_D11, var_D30
```



NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.11.10 VAL, VALP, DVAL, DVALP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
● [1)]	●	●	●	●	●

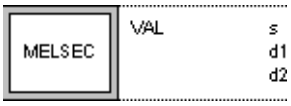
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

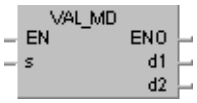
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d1	●	●	●	—	—	—	—	—	—
d2	●	●	●	●	●	●	●	—	—

GX IEC Developer

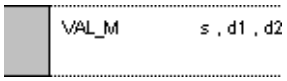
MELSEC Instruction List



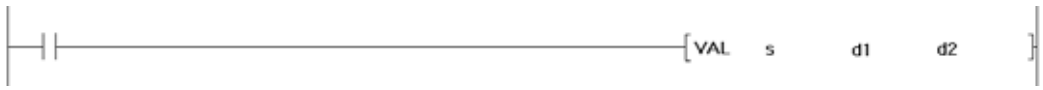
Ladder Diagram



IEC Instruction List



GX Works2



Variables

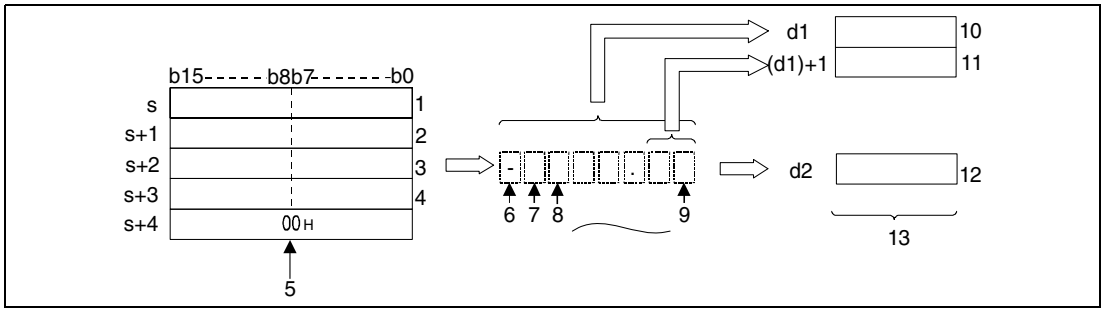
Set Data	Meaning	Data Type	
		MELSEC	IEC
s	Character string or first number of device storing the character string of the binary data to be converted	Character string	Array [1..5]/ [1..7] of ANY16
d1	First number of device storing the number of digits of the binary data after conversion	BIN 16-bit	ANY32
d2	Initial number of device storing the converted binary data	BIN 16-/32-bit	ANY16/32

Functions **Conversion of character strings into BIN 16-/32-bit binary data**

VAL Conversion into BIN 16-bit binary data

The VAL instruction converts the character strings stored in the area s (Array_s[1]) through s+4 (Array_s[5]) into BIN 16-bit data. The number of digits and the binary value are stored in d1, (d1)+1, and d2.

For the conversion into the BIN 16-bit data format all data in the area s (Array_s[1]) through s+4 (Array_s[5]) is recognized as character string up to the character code "00H".



[1] ASCII code for the 1st character / ASCII code for the sign

[2] ASCII code for the 3rd character / ASCII code for the 2nd character

[3] ASCII code for the 5th character / ASCII code for the 4th character

[4] ASCII code for the 7th character / ASCII code for the 6th character

[5] Indicates the end of the character string

[6] Sign character

[7] 1st character

[8] 2nd character

[9] 7th character

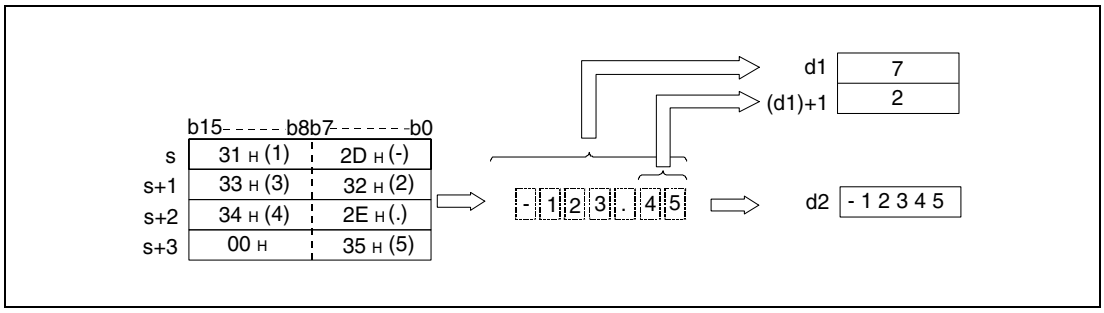
[10] Total of all digits

[11] Number of decimal places

[12] Integer value, the decimal point is not processed

[13] BIN 16-bit

The character string "-123.45" in the area s (Array_s[1]) through s+4 (Array_s[5]) is to be converted. The result will be stored in d1, (d1)+1 and d2 as follows:



The number of all characters stored in s (Array_s[1]) through s+4 (Array_s[5]) may range from 2 to 8.

The number of possible decimal places stored in the area s (Array_s[1]) through s+4 (Array_s[5]) may range from 0 to 5. In general the number of decimal places must not exceed the total of all digits minus 3.

The numerical value of a character string to be converted with the decimal point ignored must range from -32768 to 32767. The numerical value of the ASCII character string with the sign character and decimal point ignored must range from "30H" to "39H".

A positive sign of the binary data is stored as ASCII character "20H" (blank).

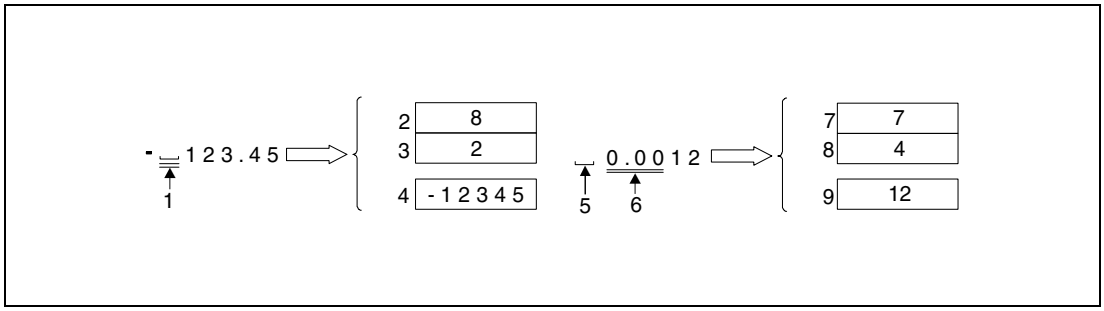
A negative sign of the binary data is stored as ASCII character "2DH" ("minus"- character).

The ASCII character "2EH" is stored as decimal point.

The total of all digits stored in d1, (d1)+1, and d2 contains all characters that represent the numerical value as well as the sign character d1 and the decimal places (d1)+1.

In the binary data stored in d2 after the conversion the decimal point is ignored.

If the characters "20H" (blank) or "30H" (zero) are stored between character sign and first numerical value, these are ignored for the conversion.

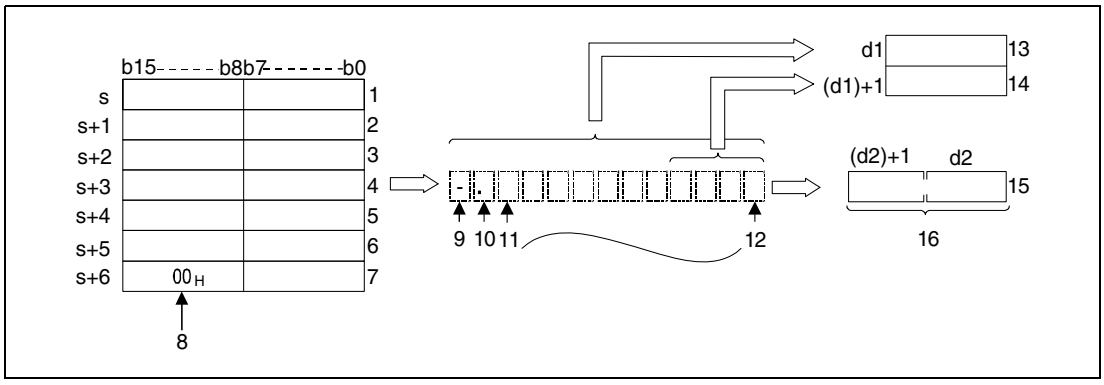


[1] These characters are not processed
[2] Total of all digits
[3] Number of decimal places
[4] Binary value
[5] Sign character
[6] These characters are not processed
[7] Total of all digits
[8] Number of decimal places
[9] Binary value

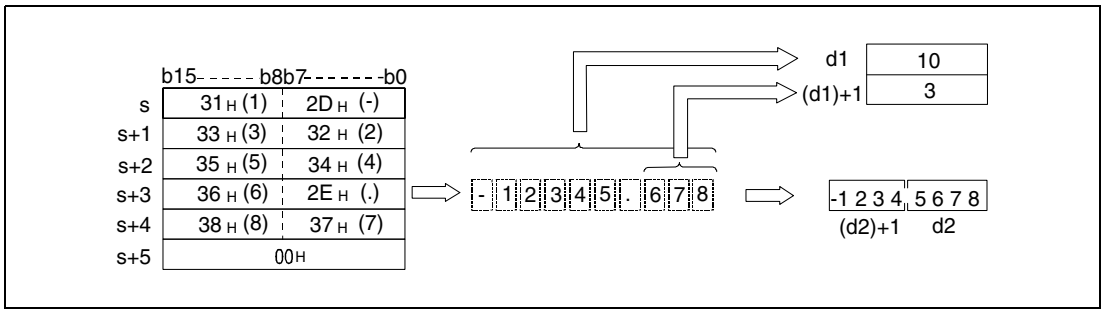
DVAL Conversion into BIN 32-bit data

The DVAL instruction converts the character strings stored in s (Array_s[1]) through s+6 (Array_s[7]) into BIN 32-bit data. The number of digits and the binary value are stored in d1, (d1)+1, d2 and (d2)+1.

For the conversion into the BIN 32-bit binary format all data in the area s (Array_s[1]) through s+6 (Array_s[7]) up to the character code "00H" are recognized as character string.



[1] ASCII code for the 1st character / ASCII code for the sign character
[2] ASCII code for the 3rd character / ASCII code for the 2nd character
[3] ASCII code for the 5th character / ASCII code for the 4th character
[4] ASCII code for the 7th character / ASCII code for the 6th character
[5] ASCII code for the 9th character / ASCII code for the 8th character
[6] ASCII code for the 11th character / ASCII code for the 10th character
[7] ASCII code for the zero character / ASCII code for the 12th character
[8] Indicates the end of the character string
[9] Sign character
[10] 1st character
[11] 2nd character
[12] 12th character
[13] Total of all digits
[14] Number of decimal places
[15] Integer value, the decimal point is not processed
[16] BIN 32-bit



The total of all characters stored in s (Array_s[1]) through s+6 (Array_s[7]) may range from 2 to 13.

The number of possible decimal places stored in the area s (Array_s[1]) through s+6 (Array_s[7]) may range from 0 to 10. In general the number of decimal places must not exceed the total of all digits minus 3.

The numerical value of a character string to be converted with the decimal point ignored must range from -2147483648 to 2147483647. The numerical value of the ASCII character string with the sign character and decimal point ignored must range from "30H" to "39H".

A positive sign of the binary data is stored as ASCII character "20H" (blank).

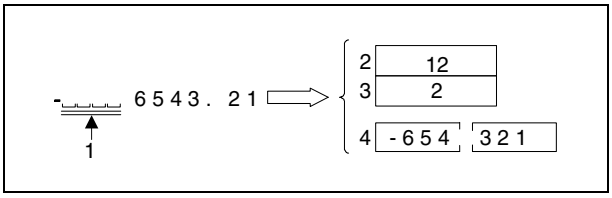
A negative sign of the binary data is stored as ASCII character "2DH" ("minus"- character).

The ASCII character "2EH" is stored as decimal point.

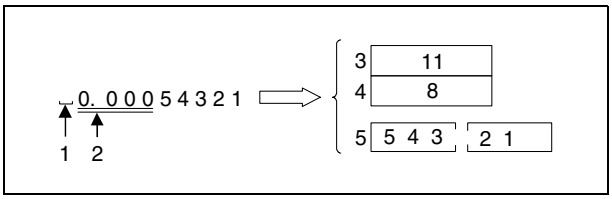
The total of all digits stored in d1, (d1)+1, d2, and (d2)+1 contains all characters that represent the numerical value as well as the sign character d1 and the decimal places (d1)+1.

In the binary data stored in d2 and (d2)+1 after the conversion the decimal point is ignored.

If the characters "20H" (blank) or "30H" (zero) are stored between character sign and first numerical value, these are ignored for the conversion.



[1] These characters are not processed
[2] Total of all digits
[3] Number of decimal places
[4] BIN 32-bit binary value



[1] Sign character
[2] These characters are not processed
[3] Total of all digits
[4] Number of decimal places
[5] BIN 32-bit binary value

Operation Errors

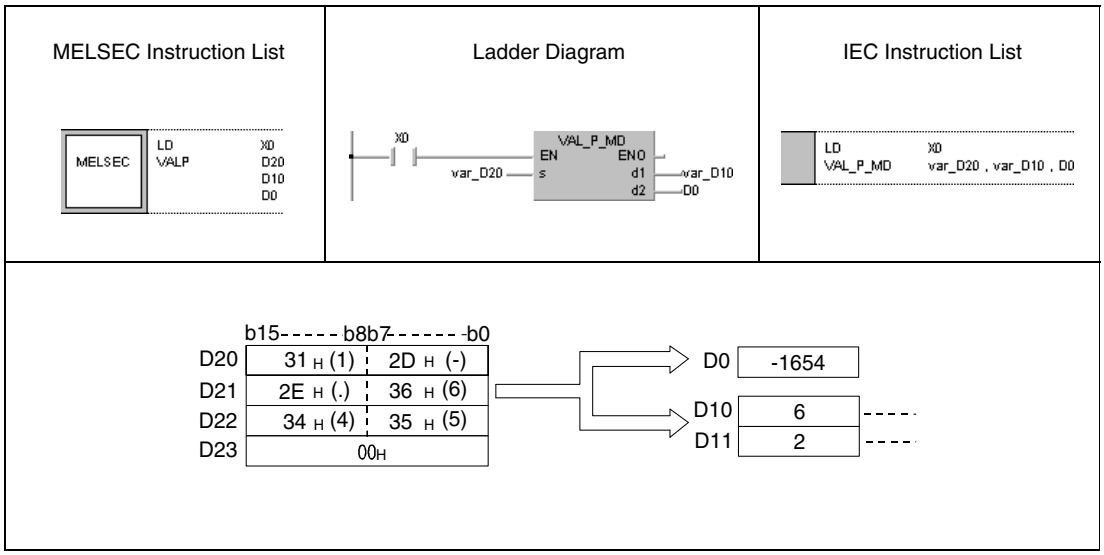
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The total of all digits stored from s (Array_s[1]) onwards exceeds the range of values from 2 to 8 (VAL) or 2 to 13 (DVAL) respectively. (Error code 4101)
- The number of decimal places stored in (d1)+1 exceeds the range of values from 0 to 5 (VAL) or 0 to 10 (DVAL) respectively. (Error code 4100)
- The total of all digits minus 3 is greater than or equal to the number of decimal places. (Error code 4100)
- An ASCII code other than "20H" or "2DH" were stored for the character sign. (Error code 4100).
- An ASCII code other than from "30H" to "39H", or "2EH" were stored as a digit for one of the individual numbers. (Error code 4100)
- More than one decimal point is stored in one value. (Error code 4100)
- The binary value exceeds the range of values from -32768 to 32767 (VAL) or -2147483648 to 2147483647 (DVAL) after the conversion. (Error code 4100)
- The ASCII character "00H" is placed to the wrong digit. (Error code 4100)

Program Example 1

VALP

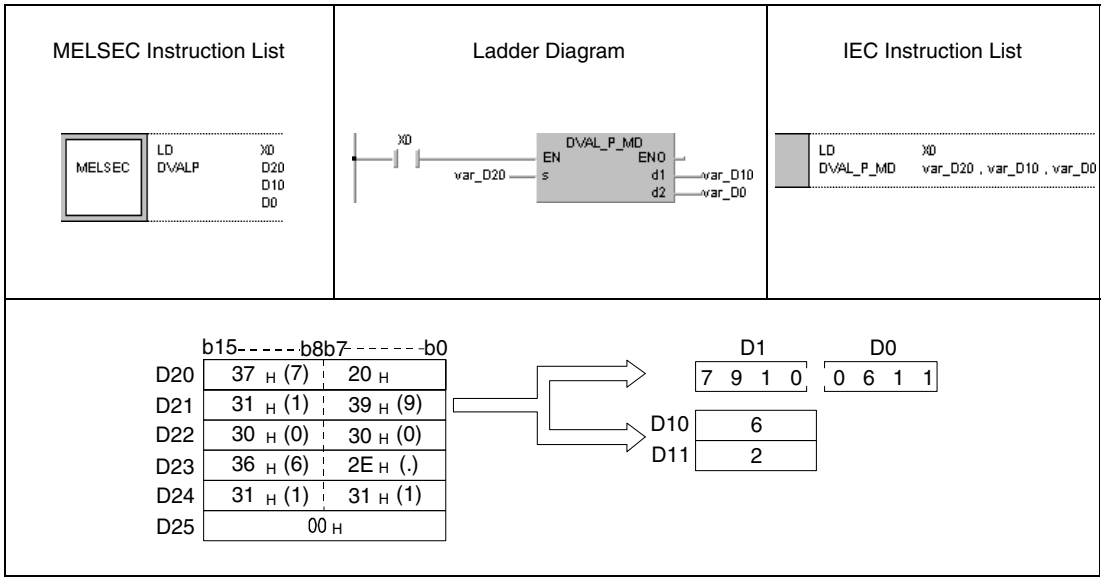
With leading edge from X0, the following program converts the character string stored in the area D20 (var_ D20 Array [1]) through D23 (var_ D20 Array [4]) into an integer value, converts this value into a BIN 16-bit binary value, and stores it in D0.



Program Example 2

DVALP

With leading edge from X0, the following program converts the character string stored in the area D20 (var_ D20 Array [1]) through D24 (var_ D20 Array [5]) into an integer value, converts this value into a BIN 32-bit value, and stores it in D0 and D1.



NOTE

These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.11.11 ESTR, ESTRP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
● [1)]	●	●	●	●	●

[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

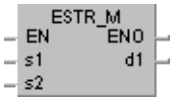
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□N□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	●	●	● [1)]	●	—
s2	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—

[1] Available only in multiple Universal model QCPU and LCPU

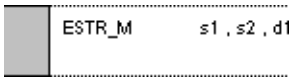
GX IEC Developer

MELSEC Instruction List

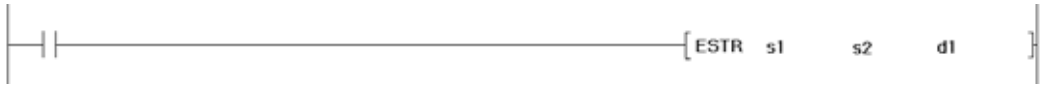
Ladder Diagram



IEC Instruction List



GX Works2



Variables

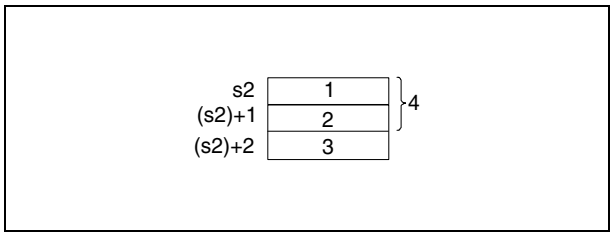
Set Data	Meaning	Data type	
		MELSEC	IEC
s1	Floating point data to be converted or initial number of device storing such data	Real number	Real number
s2	First number of device storing the data format of the numeric data to be converted	BIN 16-bit	Array [1..3] of ANY16
d	First number of device storing the converted data	Character string	Character string

Functions **Conversion of floating point data into character string data**

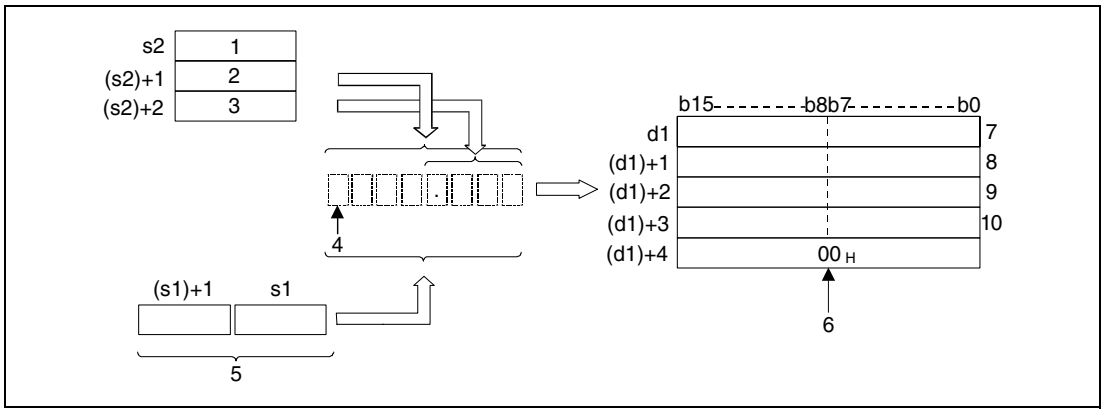
ESTR Conversion of floating point data

The ESTR instruction converts the floating point data (real numbers) in s1 and (s1)+1 into character string data. The data format of the character string is specified in s2 (Array_s2[1]) through (s2)+2 (Array_s2[3]). The result is stored from d onwards.

The data format after the conversion depends on the data format in s2 (Array_s2[1]) through (s2)+2 (Array_s2[3]).



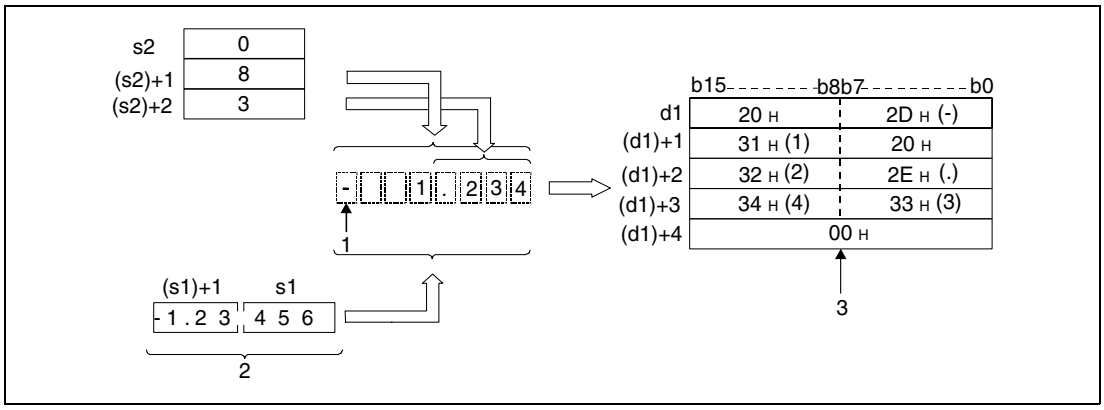
[1] Data format (decimal format "0"/ exponential format "1")
[2]Total of all digits
[3]Number of decimal places



[1] Data format (decimal format "0" / exponential format "1")
[2] Total of all digits
[3] Number of decimal places
[4] Sign character
[5] Floating point data (real number)
[6] End of character string, placed automatically
[7] ASCII code for the (total number of digits -1)th digit / ASCII code of the sign
[8] ASCII code for the (total number of digits -3)th digit / ASCII code for the (total number of digits -2)th digit
[9] ASCII code for the (total number of digits -5)th digit / ASCII code for the (total number of digits -4)th digit
[10]ASCII code for the (total number of digits -7)th digit / ASCII code for the (total number of digits -6)th digit

Decimal format

The real number -1.23456 is converted into a character string with a total of 8 digits (3 decimal places included). The result is stored from d onwards.



[1] Sign character

[2] Floating point number (real number)

[3] End of character string, automatically placed

The total number of all digits of the number in (s2)+1 (Array_s2[2]) to be converted is represented as follows:

– If the number of decimal places is zero, the total number of digits is >= 2.

– If the number of the decimal places is a different value, the total number of all digits is 3 plus the number of decimal places.

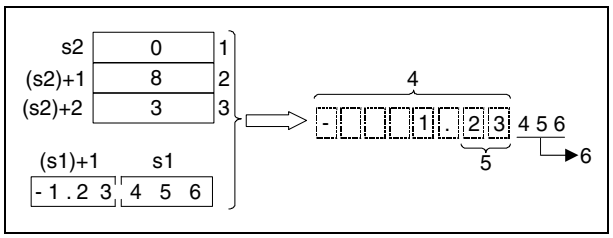
The number of decimal places that has to be specified must range within 0 and 7. In general, the number of decimal places must be less than or equal to the total number of all digits minus 3.

After the conversion the character string in d is stored as follows:

– A positive sign of the floating point data is stored as ASCII character "20H" (blank).

– A negative sign of the floating point data is stored as ASCII character "2DH" ("minus"-character).

In cases where the actual number of decimal places of the floating point data exceeds the specified number of decimal places, the surplus digits are cut off.



[1] Data format (decimal format "0"/ exponential format "1")

[2] Total of all digits

[3] Number of decimal places

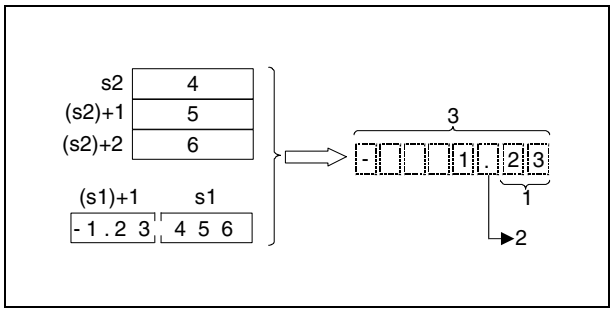
[4] Total of all digits

[5] Number of decimal places

[6] These digits are cut off

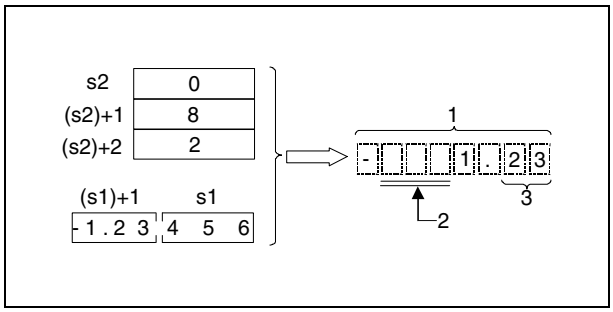
If the number of decimal places is specified a value different from zero, the decimal point "2EH" (.) is placed automatically in the specified digit.

If the number of decimal places is specified zero the decimal point "2EH" (.) is not placed.



[1] Number of decimal places
[2] Decimal point is placed and stored automatically
[3] Total of all digits

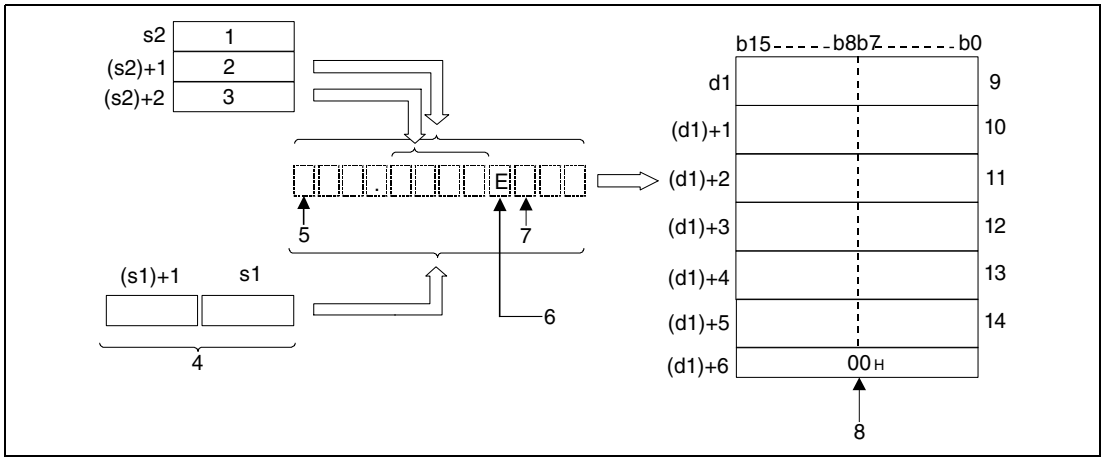
If the total number of digits, excluding the sign, the decimal point and the decimal fraction part, is greater than the integer part of the 32-bit floating point type real number data, "20H (space)" will be stored between the sign and the integer part.



[1]Total of all digits
[2]Blanks "20H" are stored
[3]Number of decimal places

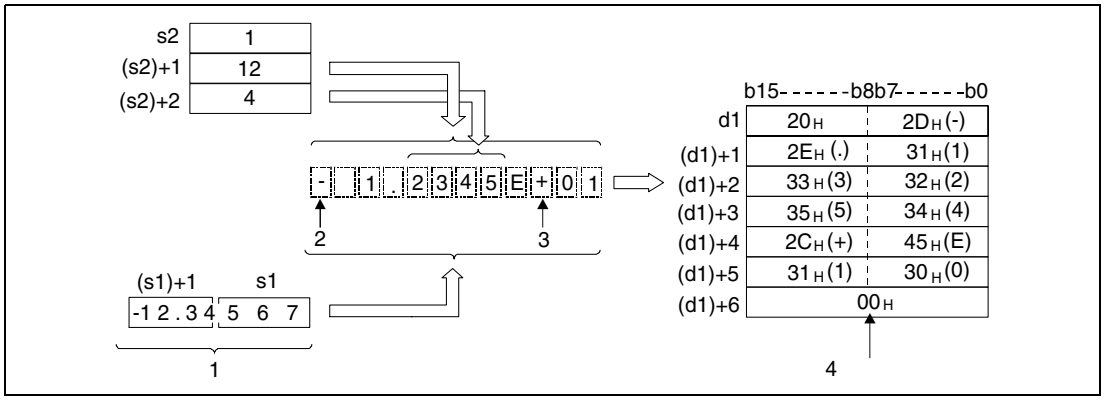
The character code "00H" is stored automaticallly at the end of the character string.

Exponential format



[1] Data format (Exponential format) (1)
[2] Total number of all digits
[3] Number of decimal places
[4] Floating point number (real number)
[5] Sign of the integer value
[6] The "E" is placed automatically
[7] Sign of the exponent
[8] End of character string indication, placed automatically
[9] ASCII code for the (total number of digits -1)th digit / ASCII code of the sign
[10] ASCII code for the (total number of digits -3)th digit / ASCII code for the (total number of digits -2)th digit
[11] ASCII code for the (total number of digits -5)th digit / ASCII code for the (total number of digits -4)th digit
[12] ASCII code for the (total number of digits -7)th digit / ASCII code for the (total number of digits -6)th digit
[13] Sign of the exponent/ 45_H (E)
[14] ASCII code for the (total number of digits -11)th digit (exponent)/ASCII code for the (total number of digits -10)th digit (exponent)

Example The real number -12.34567 is to be represented in exponential notation. The total number of all digits is 12. The number of decimal digits is specified 4. The result is stored from d1 onwards.



[1] Floating point number (real number)

[2] Sign of the integer value

[3] Sign of the exponent

[4] End of character string indication, placed automatically

The total number of all digits of the number in (s2)+1 (Array_s2[2]) to be converted is represented as follows:

– If the number of decimal places is zero, the total number of digits is >= 2.

– If the number of the decimal places is a different value, the total number of all digits is 7 plus the number of decimal places.

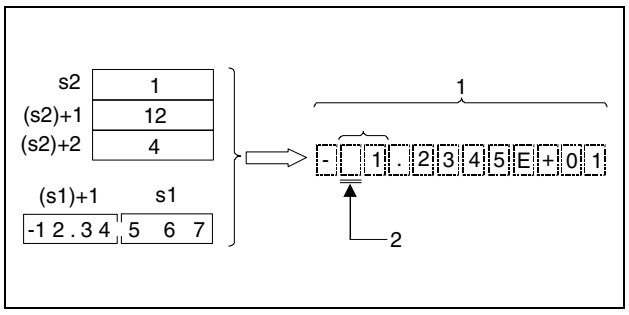
The number of decimal places that has to be specified must range within 0 and 7. In general, the number of decimal places must be less than or equal to the total number of all digits minus 7.

After the conversion the character string in d is stored as follows:

– A positive sign of the floating point data is stored as ASCII character "20H" (blank).

– A negative sign of the floating point data is stored as ASCII character "2DH" ("minus"-character).

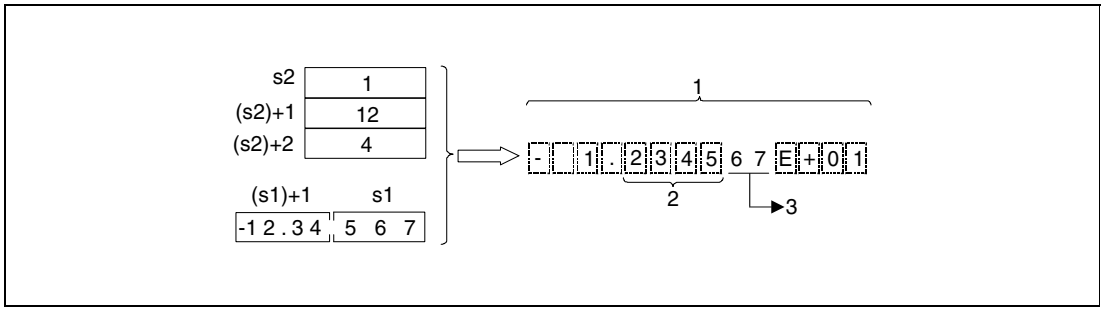
The integer range is fixed to 2 digits. If the integer range contains one digit only, a blank in ASCII code is placed and stored between the sign character and the integer digit.



[1] Total of all digits (12)

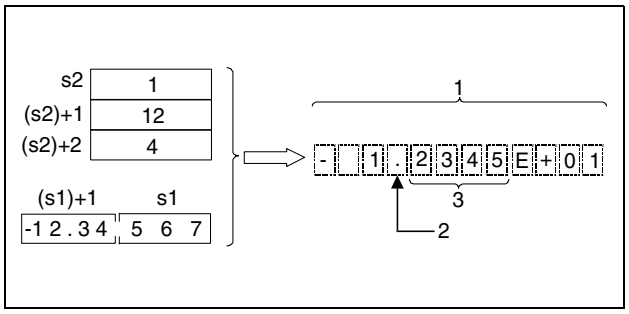
[2] Becomes a blank

If the floating point value of the decimal range is longer than the relevant storage range, the digits that cannot be stored are cut off.



[1] Total of all digits (12)
[2] Number of digits in the decimal range (4)
[3] These digits are cut off

If the number of decimal places is specified a value different from zero, the decimal point "2EH" (.) is placed automatically in the specified digit.

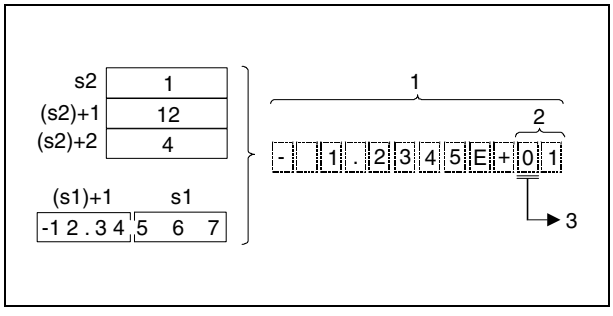


[1] Total of all digits (12)
[2] Is placed automatically
[3] Number of digits in the decimal range (4)

If the number of decimal places is specified zero the decimal point "2EH" (.) is not placed.

The ASCII code "2CH" (+) is placed and stored for a positive exponent.
The ASCII code "2DH" (-) is placed and stored for a negative exponent.

The exponential range is fixed to 2 digits. If the exponential range contains one digit only, the ASCII code "30H" (0) is placed and stored between the exponent sign and the exponent.



[1] Total of all digits (12)
[2] Is fixed to 2 digits
[3] Is set to zero automatically

The character code "00H" is stored automaticallly at the end of the character string.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The devices specified in s1 and (s1)+1 are not within the following range:
  $0, \pm 2^{-126} \leq s1 < \pm 2^{128}$.
  (Error code 4100)
- The format in s2 (Array_s2[1]) is neither 0 nor 1.
  (Error code 4100)
- The total number of digits in (s2)+1 (Array_s2[2]) exceeds the range of values:
  (Error code 4100)

  For the decimal format
  The number of decimal places is zero (total number of digits $\geq$ 2).
  The number of decimal places is different from zero
  (total number of digits $\geq$ (number of decimal places + 3)).

  For the exponential format
  The number of decimal places is zero (total number of digits $\geq$ 2).
  The number of decimal places is different from zero
  (total number of digits $\geq$ (number of decimal places + 7)).
- The number of digits in (s2)+2 (Array_s2[3]), forming the decimal part exceeds the range of values:
  (Error code 4100)

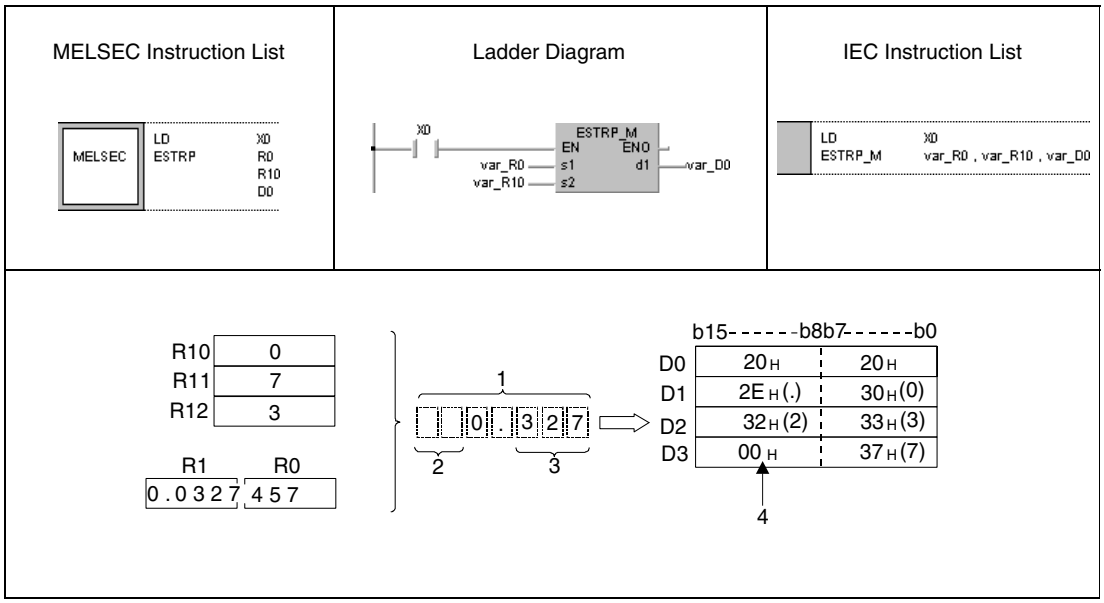
  For the decimal format
  The number of digits forming the decimal part is less than or equal to the total number of digits minus 3.

  For the exponential format
  The number of digits forming the decimal part is less than or equal to the total number of digits minus 7.
- The value whose total digits exceeds "24" is specified.
  (Error code 4100)
- The storage range in d exceeds the relevant storage device range.
  (Error code 4101)
- The device specified by s2 exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU.)
  (Error code 4101)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example 1

ESTRP

With leading edge from X0, the following program converts a floating point value (real number) specified by the devices R0 and R1 into the format specified by R10 (var_R10 Array [1]) through R12 (var_R10 Array [3]) and stores the result in D0 through D3.



[1] Total number of digits

[2] Blanks

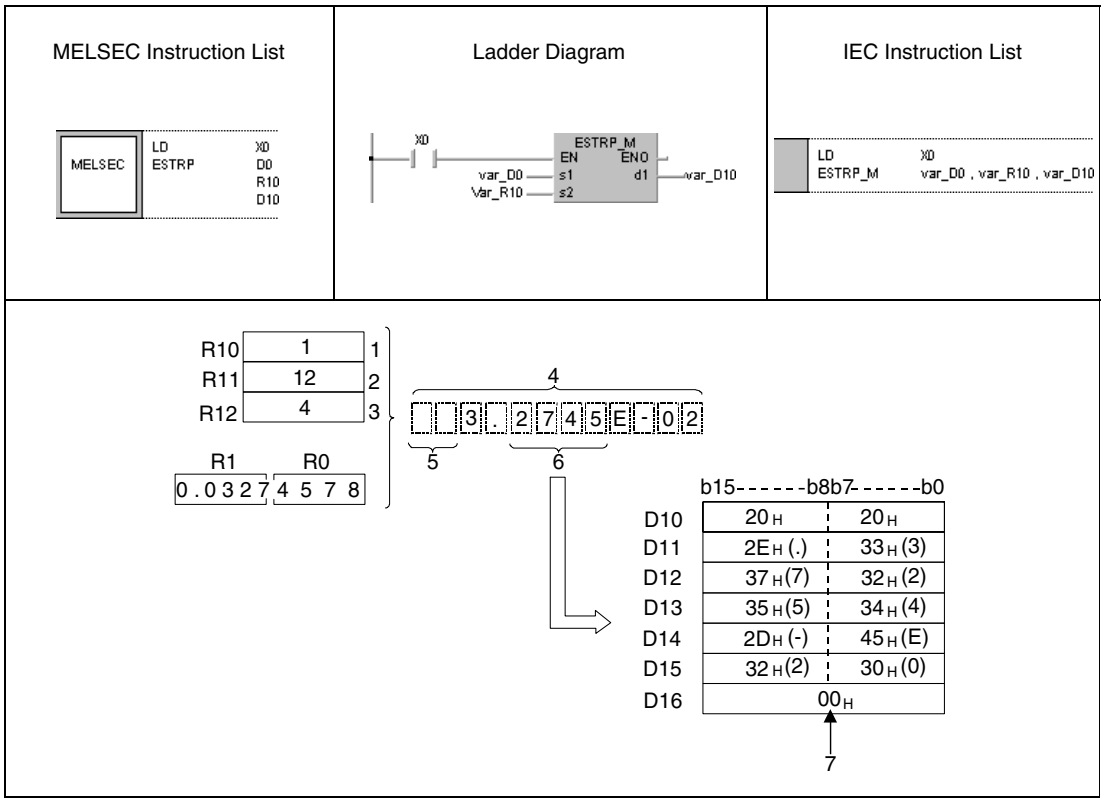
[3] Number of decimal places

[4] Is stored automatically

Program Example 2

ESTRP

With leading edge from X0, the following program converts a floating point value (real number) specified by D0 and D1 into the format specified by R10 (var_R10 Array [1]) through R12 (var_R10 Array [3]) and stores the result in D10 through D16.



[1] Data format (Exponential representation) (1)

[2] Total number of all digits

[3] Number of decimal places

[4] Total number of all digits

[5] Blanks

[6] Number of decimal places in the decimal part

[7] Is stored automatically

7.11.12 EVAL, EVALP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
● [1)]	●	●	●	●	●

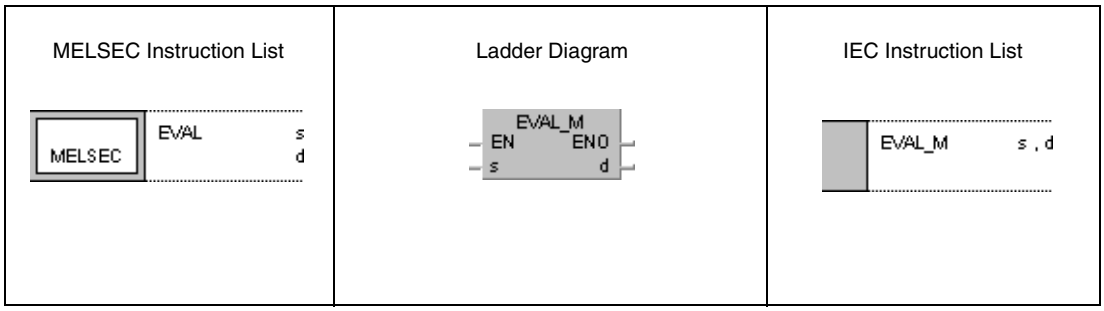
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

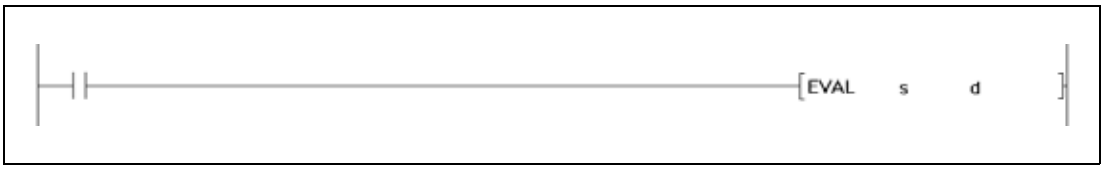
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	● [1)]	● [1)]	—	—	—

[1] Available on Universal model QCPU and LCPU

GX IEC Developer



GX Works2



Variables

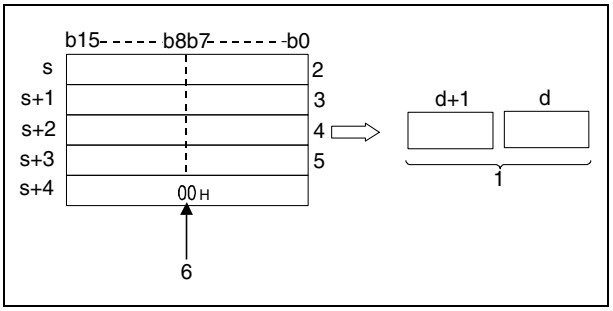
Set Data	Meaning	Data Type
s	Character string data to be converted into a floating point number (data type REAL) or initial number of device storing such data	Character string
d	First number of device storing the converted decimal floating point number (data type REAL)	REAL

Functions **Conversion of character string data into decimal floating point data**

EVAL Conversion of character strings

The EVAL instruction converts the character string in s through s+4 into a decimal floating point number (real number). The result is stored in d.

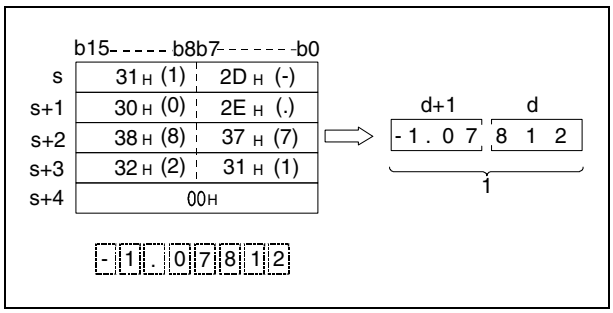
The characer string can be converted into decimal floating point format as well as into the exponential format.



[1] Decimal floating point data (data type REAL)
[2] ASCII code of the 1st character/ ASCII code of sign character
[3] ASCII code of the 3rd character/ ASCII code of the 2nd character
[4] ASCII code of the 5th character/ ASCII code of the 4th character
[5] ASCII code of the 7th character/ ASCII code of the 6th character

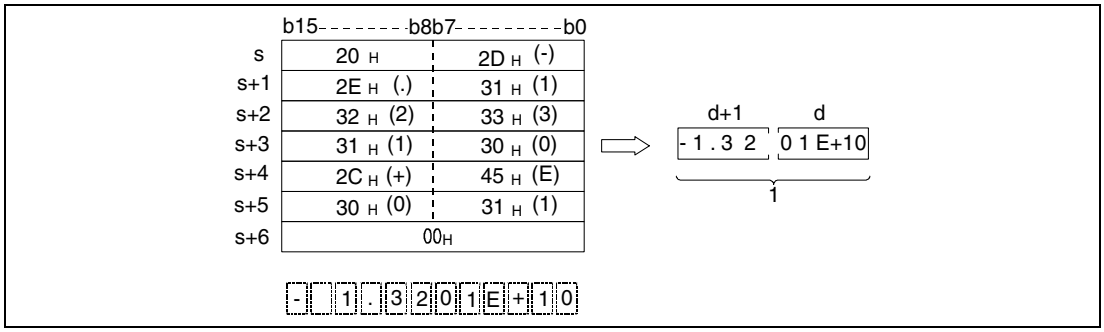
[6] Indicates the end of character string

Decimal format



[1] Decimal floating point data (real number)

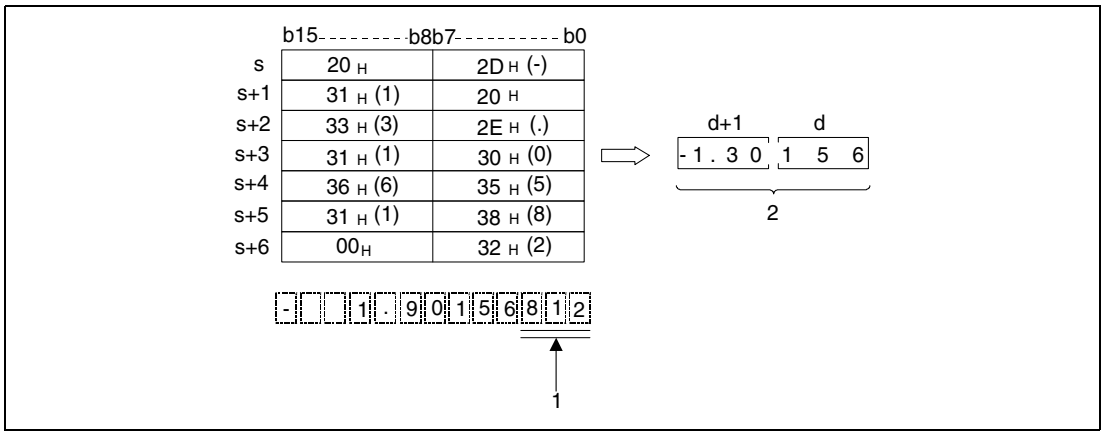
Exponential format



[1] Decimal floating point data (data type REAL)

In the example below, six digits (without sign, decimal point, and exponent digits of the result) of the character string from s onwards are converted into a decimal floating point number. The digits from the 7th digit on are cut off from the result.

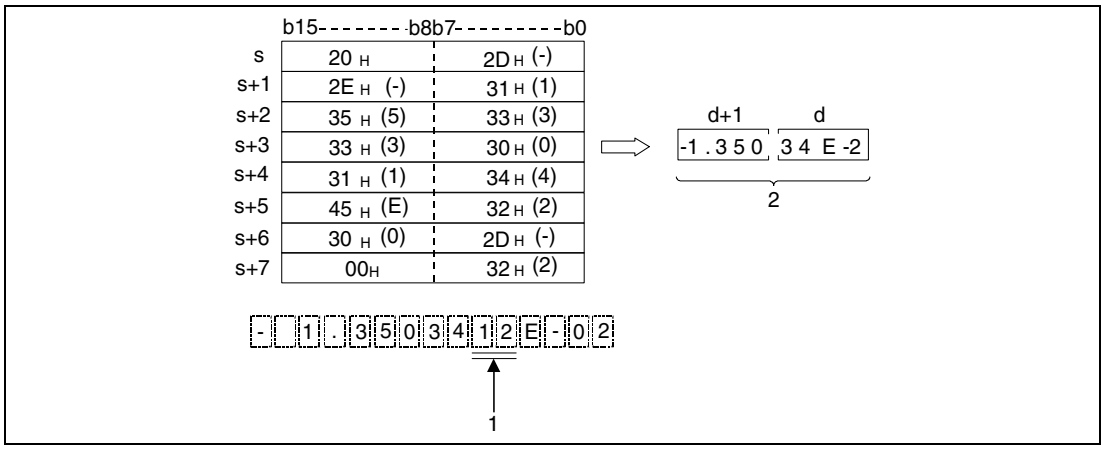
Decimal format



[1] These digits are omitted

[2] Decimal floating point data (data type REAL)

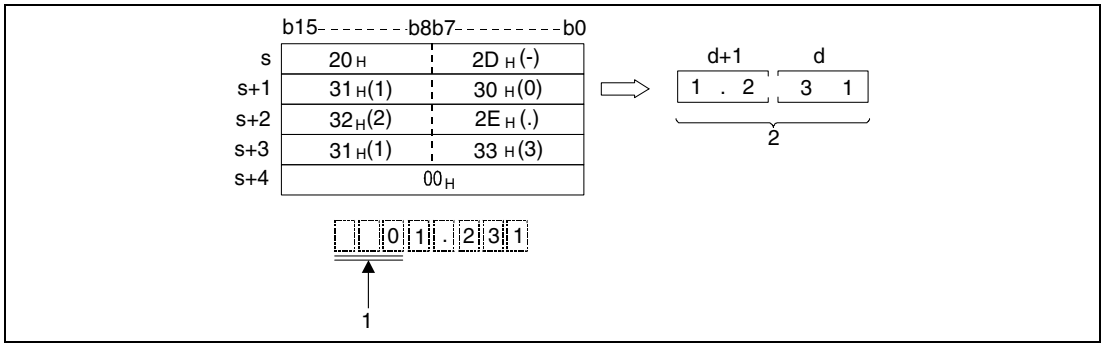
Exponential format



[1] These digits are omitted

[2] Decimal floating point data (data type REAL)

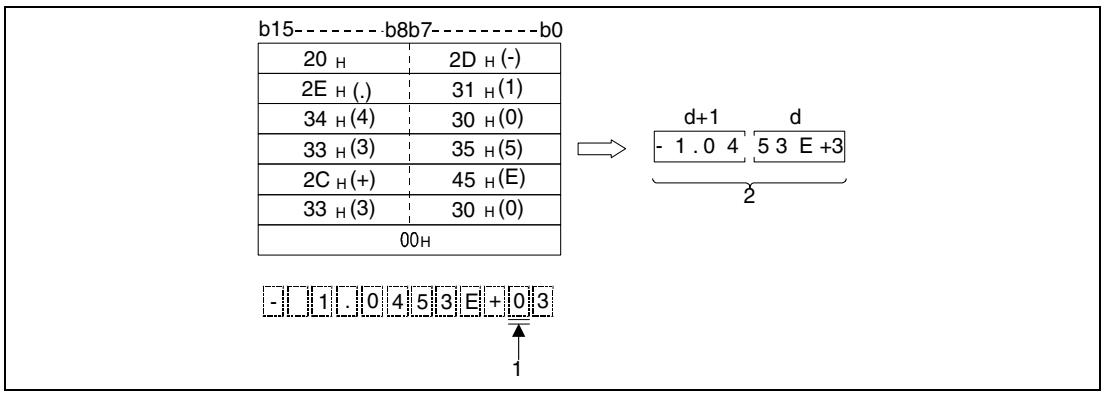
Leading blanks (ASCII code "20H") or zeroes (ASCII code "30H") in the character string from s onwards are ignored by the conversion, except for the initial zero (e.g. 0.123).



[1] These characters are ignored by the conversion

[2] Decimal floating point data (data type REAL)

If the ASCII code "30H" (zero) is placed between the character "E" and the character string for the exponential format, this character is ignored by the conversion.



[1] These characters are ignored by the conversion

[2] Decimal floating point data (data type REAL)

A character string to be converted may contain a maximum of 24 characters.

Operation Errors

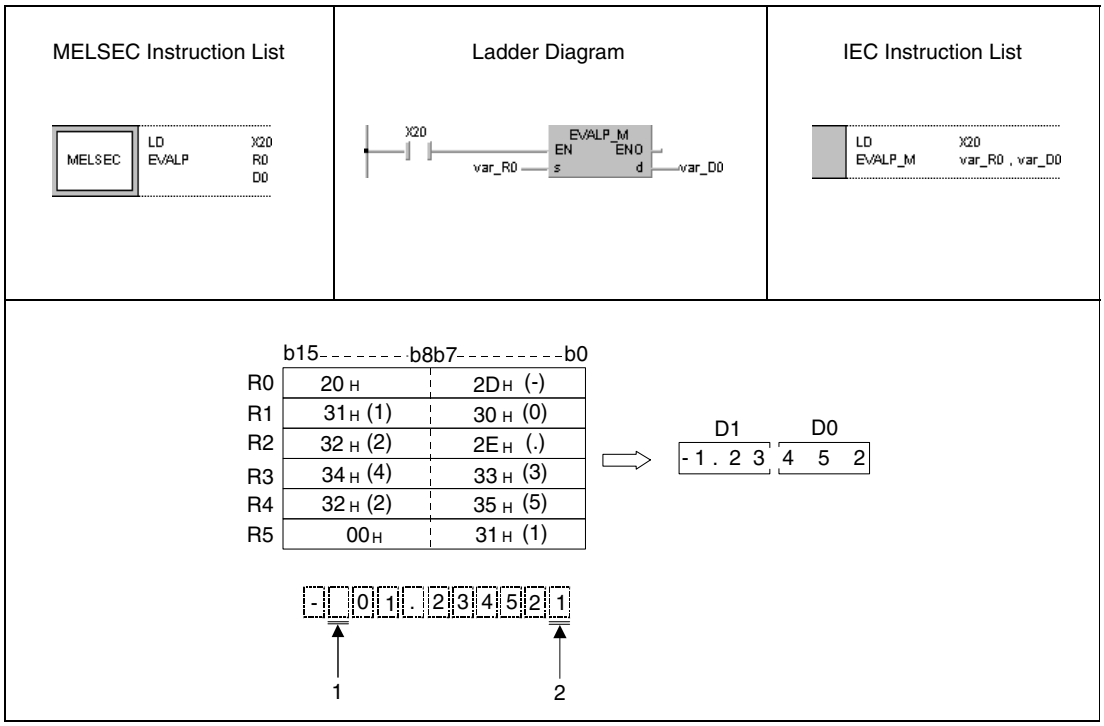
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The digits prior to the decimal point or the decimal places contain characters exceeding the range of values from "30H" (0) to "39H" (9).
  (Error code 4100)
- The character "2EH" is used more than once within the character string. (Error code 4100)
- The exponent part contains characters different from "45H (E), 2BH (+)" or "45H (E), 2DH (-)" or contains more than one exponent portion.
  (Error code 4100)
- Data after conversion is not within the following range:
  0, + 2^{-126} < (data after conversion) < + 2^{128}
  (Error code 4100)
- The end of string indicator "00H" exceeds the relevant storage device range.
  (Error code 4100)
- The number of characters in the string is 0 or greater than 24.
  (Error code 4100)

Program Example 1

EVALP

With leading edge from X20, the following program converts the character string specified in R0 through R5 into a decimal floating point number (real number) and stores the result in D0 and D1.



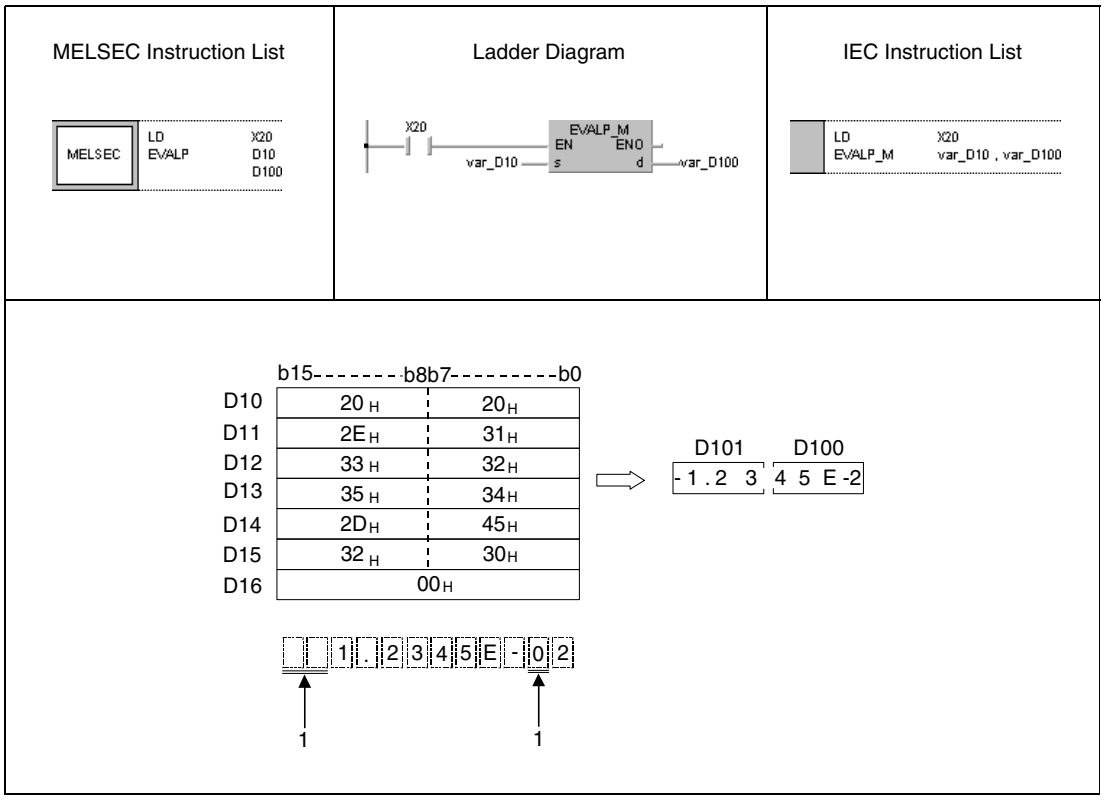
[1] This digit is not processed

[2] This number is cut off

Program Example 2

EVALP

With leading edge from X20, the following program converts the character string specified in D10 through D16 into a floating point number (data type REAL) and stores the result in D100 and D101.



[1] These digits are not processed

NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.11.13 ASC, ASCP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

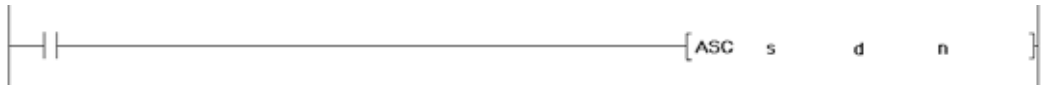
MELSEC Instruction List	Ladder Diagram	IEC Instruction List
MELSEC ASC_MD s d n		

NOTE

The ASC and the ASCP instructions do not work with the IEC editors. The only way to program these instructions is by using the MELSEC instruction list.

Remedy: Move the hexadecimal ASCII format direct into the target registers.

GX Works2



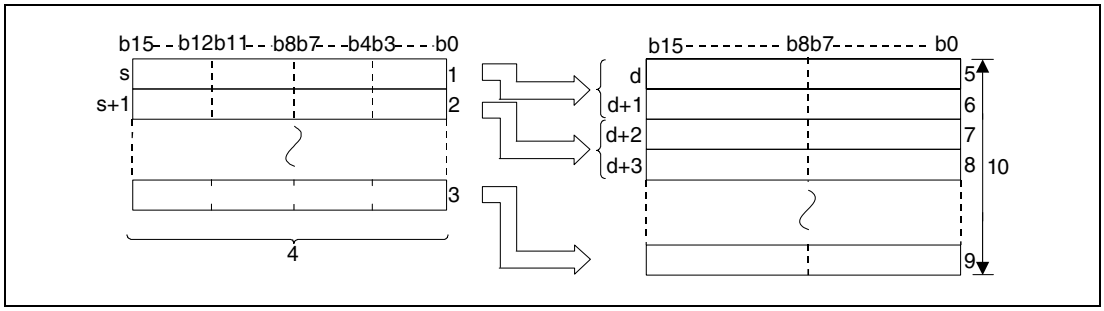
Variables

Set Data	Meaning	Data Type
s	Head number of the devices where BIN data to be converted to a character string is stored	BIN 16-bit
d	First number of device storing converted character string	Character string
n	Number of characters to be stored	BIN 16-bit

Functions **Conversion of BIN 16-bit data into ASCII code**

ASC/ASCP Conversion instruction

The ASCII instruction converts the 16-bit binary data stored from s onwards into the hexadecimal ASCII format and stores the result considering the number of characters specified by n from d onwards.



[1] First digit / second digit / third digit / fourth digit

[2] First digit / second digit / third digit / fourth digit

[3] First digit / second digit / third digit / fourth digit

[4] Binary data

[5] ASCII code of the 1st digit / ASCII code of the 2nd digit

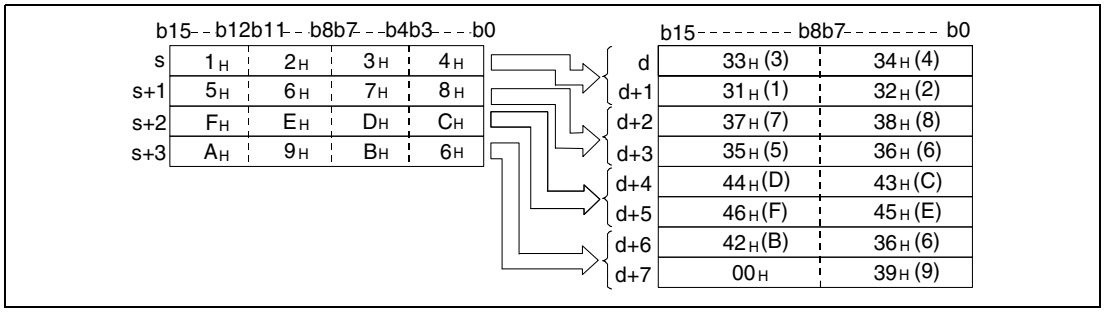
[6] ASCII code of the 3rd digit / ASCII code of the 4th digit

[7] ASCII code of the 5th digit / ASCII code of the 6th digit

[8] ASCII code of the 7th digit / ASCII code of the 8th digit

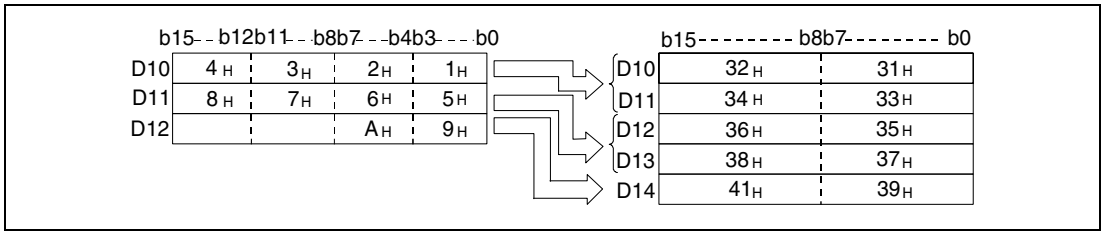
[9] ASCII code of the 9th digit / ASCII code of the 10th digit

[10] Number of digits specified in n

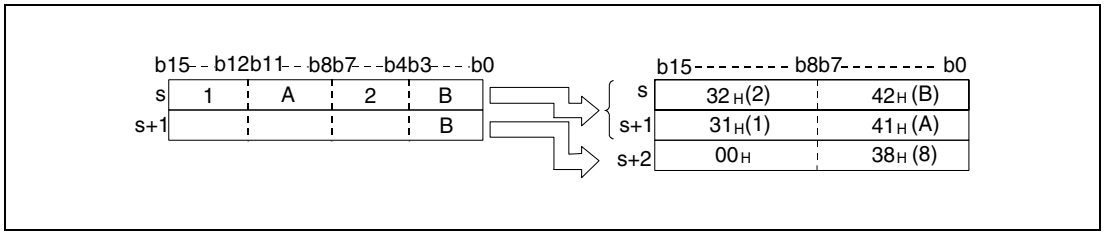


The number of characters specified in n determines the ranges of values of the devices specified from s and d onwards. The devices specified from s onwards contain the binary data to be converted. The converted character string is stored in the devices specified from d onwards.

The program is even processed accurately and without an error message, if the storage area of the binary data to be converted overlaps with that of the converted ASCII data.



If n specifies an odd number of characters, the ASCII character "00H" is placed automatically into the upper 8 bits of the highest address of the area, storing the character string.



If the number of characters specified by n is zero, the program will not be executed.

Operation Errors

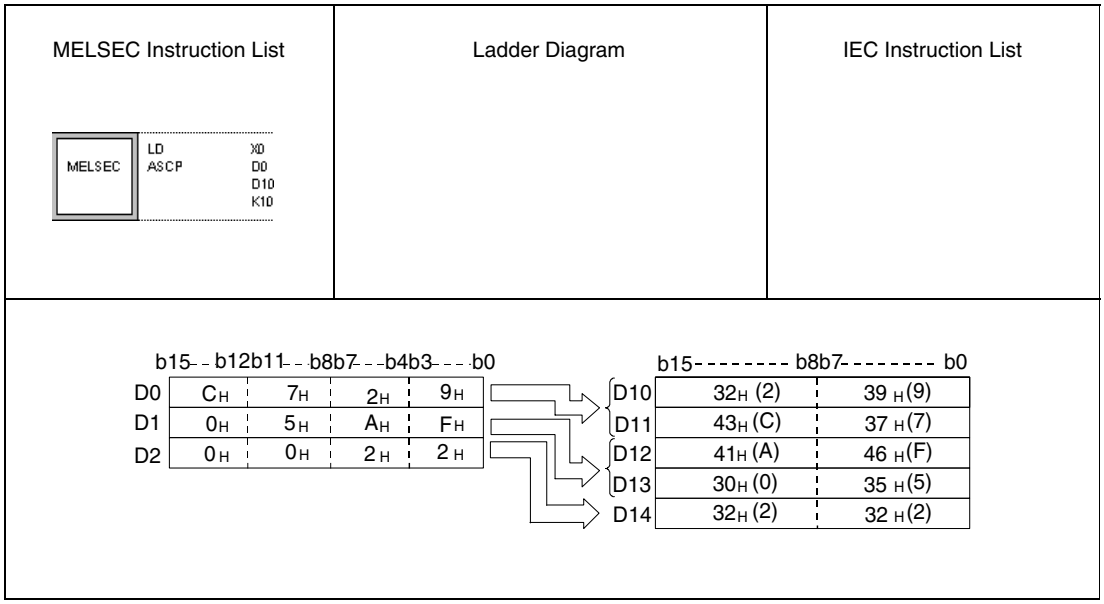
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of characters specified by n and therefore the required number of registers from s onwards exceeds the relevant storage device range.
  (Error code 4101)
- The number of characters specified by n and therefore the required number of registers from d onwards exceeds the relevant storage device range.
  (Error code 4101)

Program Example

ASCP

With leading edge from X0, the following program reads in the binary data stored in D0 as hexadecimal values and converts it into a character string. The result is stored in D10 through D14.



7.11.14 HEX, HEXP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

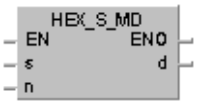
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—

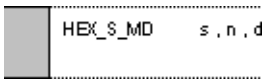
GX IEC Developer

MELSEC Instruction List

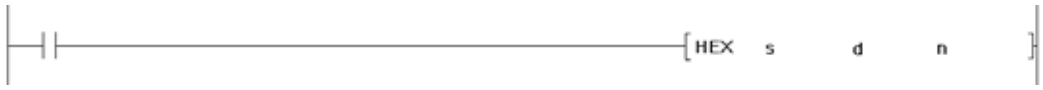
Ladder Diagram



IEC Instruction List



GX Works2



Variables

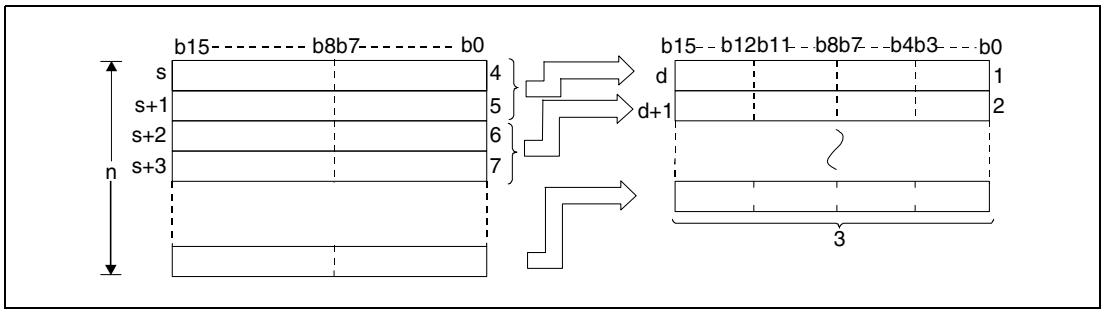
Set Data	Meaning	Data Type
s	First number of device storing character string to be converted	Character string
d	First address of area storing the converted binary data	BIN 16-bit
n	Number of characters to be converted	

Functions

Conversion of hexadecimal ASCII values into binary values

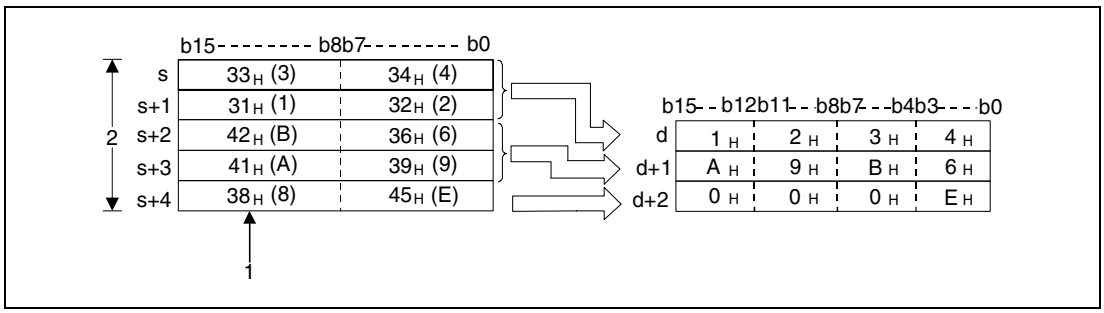
HEX Conversion of hexadecimal ASCII values

The HEX instruction converts the hexadecimal ASCII characters from s onwards into binary values. The result is stored from d onwards.



[1] 4th digit / 3rd digit / 2nd digit / 1st digit
[2] Binary data
[3] ASCII code of the 2nd digit / ASCII code of the 1st digit
[4] ASCII code of the 4th digit / ASCII code of the 3rd digit
[5] ASCII code of the 2nd digit / ASCII code of the 1st digit
[6] ASCII code of the 4th digit / ASCII code of the 3rd digit

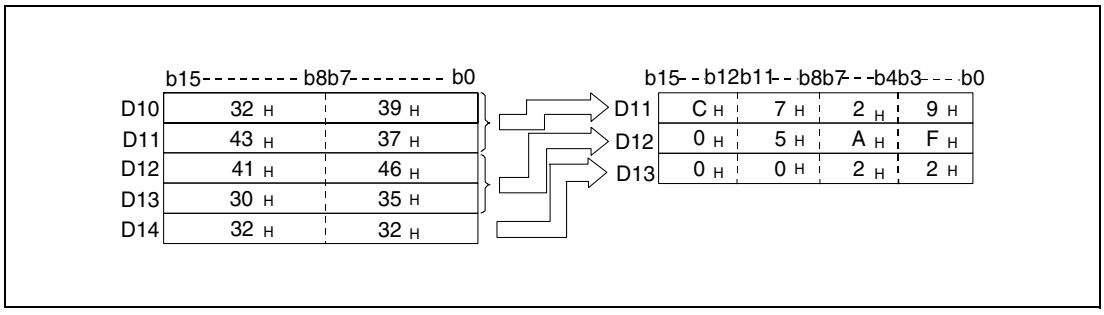
The number of characters in n is 9.



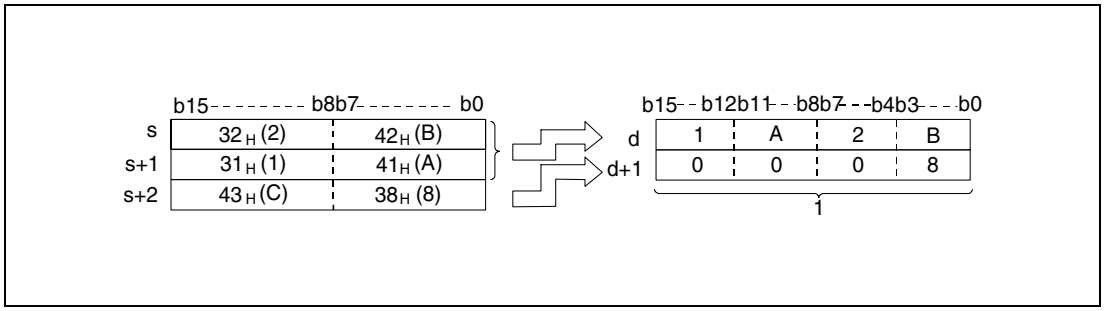
[1] Since the character string contains 9 characters, the "38H" is not changed or moved.
[2] n = 9

The number of characters specified in n determines the range of values of the character string from s and of the binary data from d onwards automatically.

Although the range of values of the ASCII code to be converted and that of the converted binary values overlap, this instruction processes the data accurately.



If the number of characters in n is not divisible by 4, a zero is written after the specified number of characters automatically to the highest registers storing the converted binary values.



[1] The value zero is stored automatically

If the number of characters in n is zero, the conversion will not be executed.

The ASCII code from s onwards may range from "30H" through "39H" and from "41H" through "46H".

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

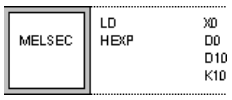
- The devices specified from s onwards contain characters exceeding the ranges from "30H" through "39H", or from "41H" to "46H".
  (Error code 4100)
- The number of characters specified by n and therefore the required number of registers from s onwards exceeds the relevant storage device range.
  (Error code 4101)
- The number of characters specified by n and therefore the required number of registers from d onwards exceeds the relevant storage device range.
  (Error code 4101)
- The value n is negative.
  (Error code 4101)

Program Example

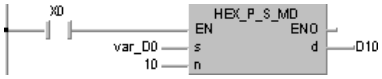
HEXP

With leading edge from X0, the following program converts the character string "6B52A71379" stored in D0 through D4 into binary data. The result is stored in D10 through D14.

MELSEC Instruction List

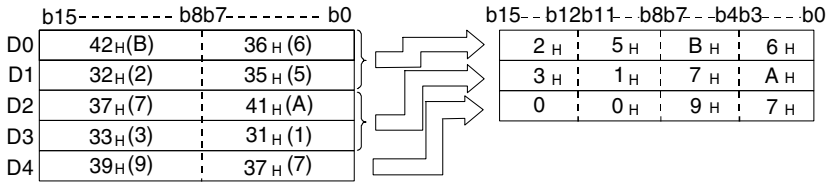


Ladder Diagram



IEC Instruction List

```
LD      X0
HEX_P_S_MD  var_D0 , 0 , D10
```



NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.11.15 RIGHT, RIGHTP, LEFT, LEFTP

CPU

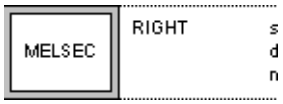
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

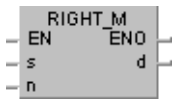
	Usable Devices									
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant		Other
	Bit	Word		Bit	Word			K, H (16#)	$	
s	—	●	●	—	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—	—

GX IEC Developer

MELSEC Instruction List

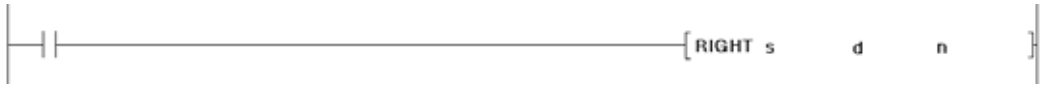


Ladder Diagram



IEC Instruction List

GX Works2



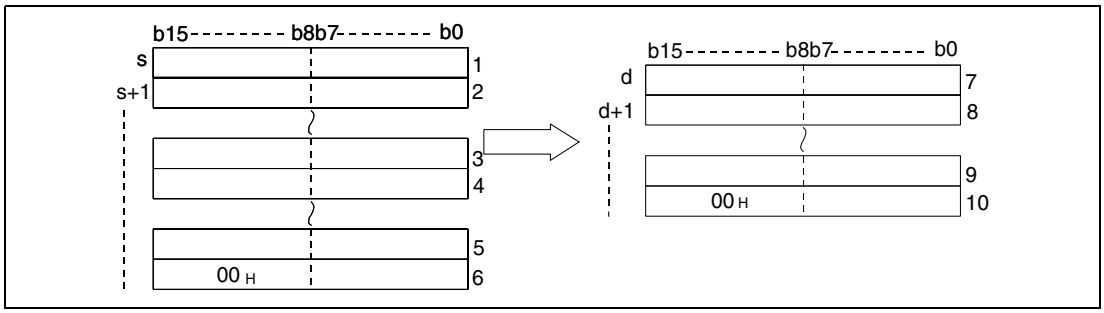
Variables

Device	Meaning	Data Type
s	First number of device storing the character string	Character string
d	First number of device area storing the determined characters of the character string	
n	Number of characters stored on the left or on the right	BIN 16-bit

Functions **Extraction of character string data from the right or from the left**

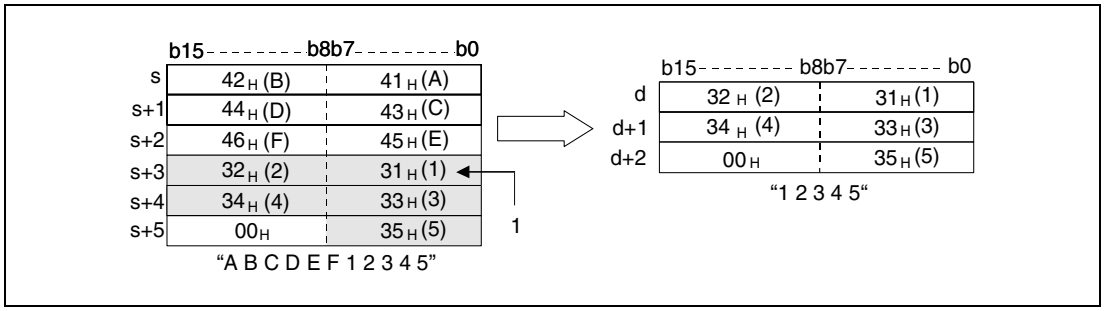
RIGHT Extract character string data from the right

The RIGHT instruction stores n characters from the right side of the character string (end of character string) from s onwards. The characters are stored from d onwards.



[1] ASCII code of the 2nd characters / ASCII code of the 1st chracter
[2] ASCII code of the 4th character / ASCII code of the 3rd character
[3] ASCII code of the last character minus n+2 / ASCII code of the last character minus n+1
[4] ASCII code of the last character minus n+4 / ASCII code of the last character minus n+3
[5] ASCII code of the last character minus 1 / ASCII code of the last character minus 2
[6] "00_H" / ASCII code of the last character
[7] ASCII code of the last character minus n+2 / ASCII code of the last character minus n+1
[8] ASCII code of the last character minus n+4 / ASCII code of the last character minus n+3
[9] ASCII code of the last character minus 1 / ASCII code of the last character minus 2
[10] "00_H" / ASCII code of the last character

– With n = 5

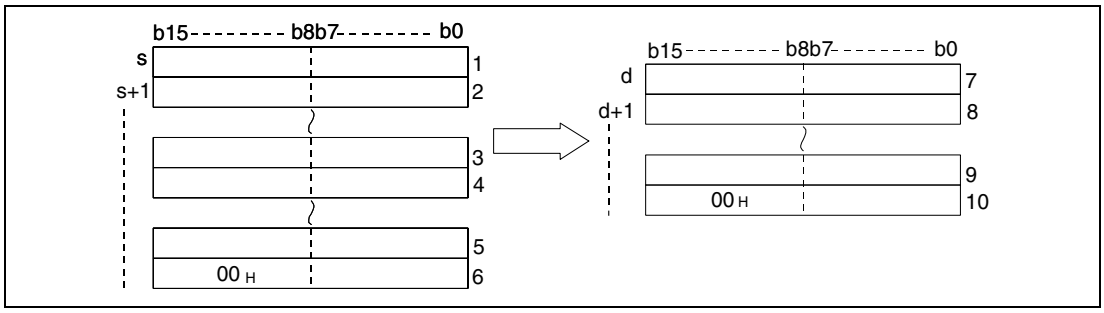


[1] ASCII code for the 5th character

If the number of characters in n is zero, the character code "00_H" is stored from d onward.

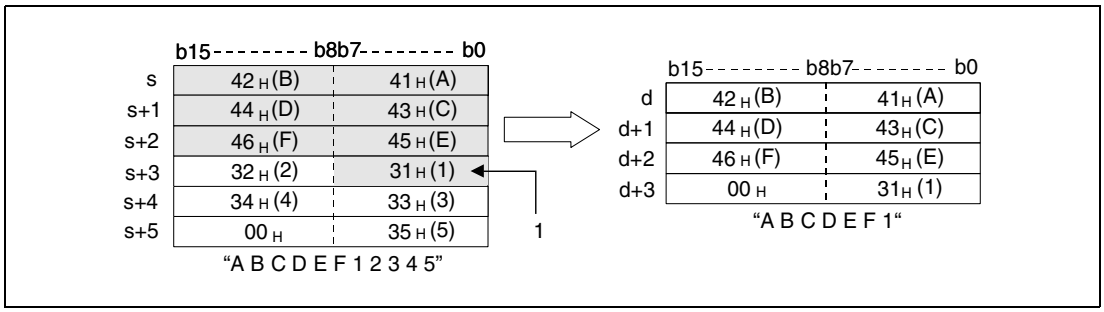
LEFT **Extract character string data from the left**

The LEFT instruction stores n characters from the left side of the character string (beginning of character string) from s onwards. The characters are stored from d onwards.



[1] ASCII code of the 2nd character / ASCII code of the 1st character
[2] ASCII code of the 4th character / ASCII code of the 3rd character
[3] ASCII code of the character n-1 / ASCII code of the character n-2
[4] ASCII code of the character n+1 / ASCII code of the nth character
[5] "00_H" / ASCII code of the last character
[6] ASCII code of the 2nd character / ASCII code of the 1st character
[7] ASCII code of the 4th character / ASCII code of the 3rd character
[8] ASCII code of the character n-1 / ASCII code of the character n- 2
[9] "00_H" / ASCII code of the nth character

– With n=7



[1] ASCII code of the 7th character

If the number of characters in n is zero, the character code "00_H" is stored from d onwards.

Operation Errors

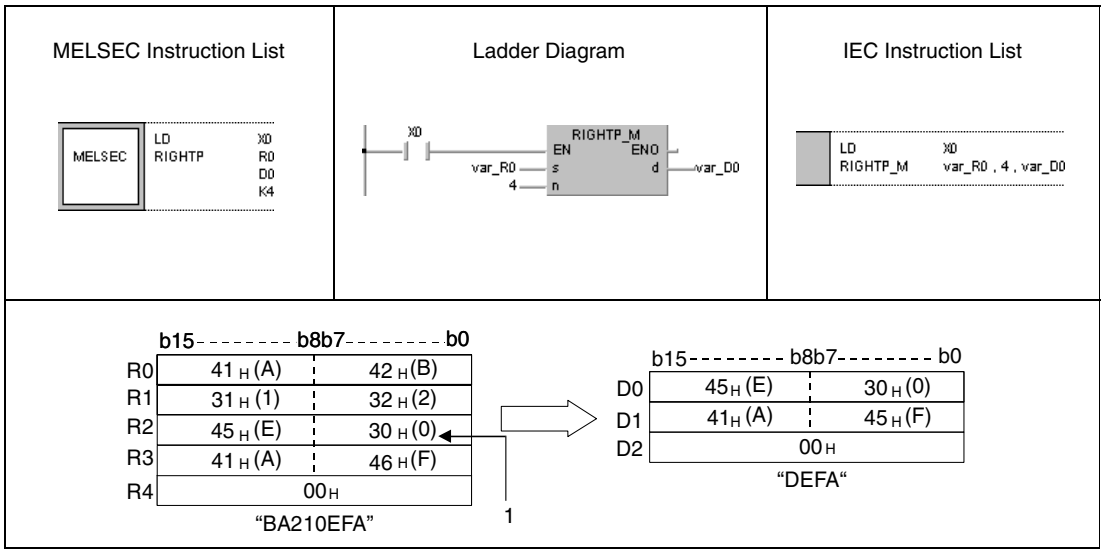
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value in n exceeds the number of existing characters stored from s onwards. (Error code 4101)
- The area specified by n exceeds the relevant device range of the device specified by d. (Error code 4101)

Program Example 1

RIGHTP

With leading edge from X0, the following program extracts 4 characters of the data from the right side of the character string stored in R0 through R4 and stores it in D0 through D2.

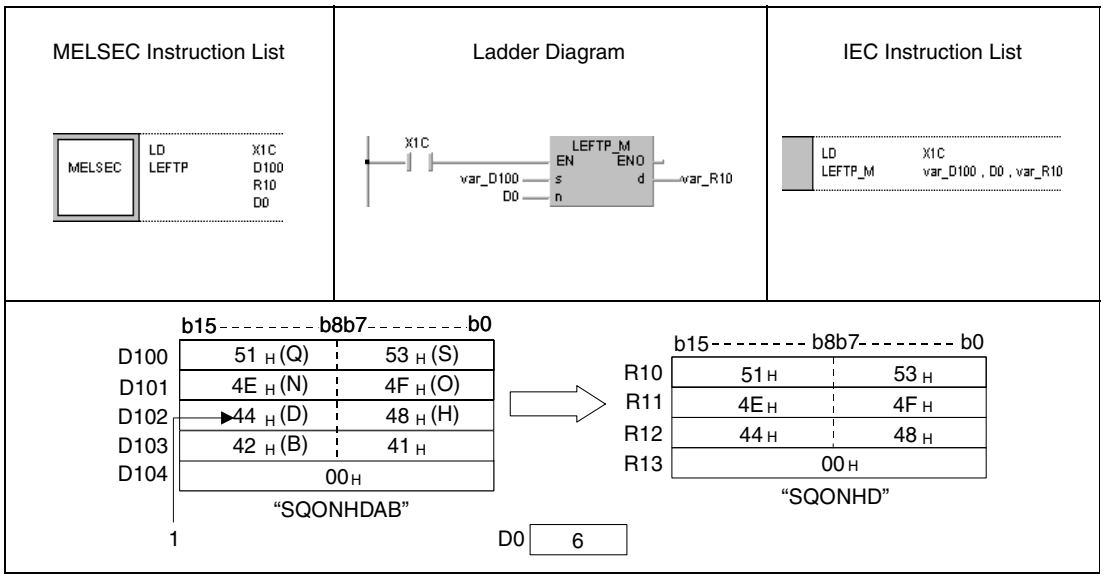


1 ASCII code of the 4th character

Program Example 2

LEFTP

With leading edge from X1C, the following program extracts the number of characters specified in D0 from the left side of the character string specified in D100 through D104. The result is stored in R10 through R13.



1 ASCII code of the 6th character

NOTE

These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.11.16 MIDR, MIDRP, MIDW, MIDWP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

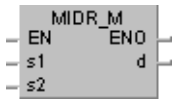
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
s2	●	●	●	●	●	●	●	—	—

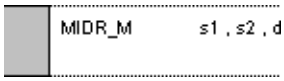
GX IEC Developer

MELSEC Instruction List

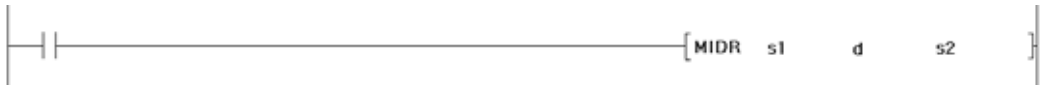
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s1	Character string or first number of device storing character string data	Character string	Character string
d	First number of device storing the operation result		
s2	First number of device storing the 1st character and the number of characters. (s2)+0: Position of the 1st character (s2)+1: Number of characters	BIN 16-bit	Array [1..2] of ANY16

Functions

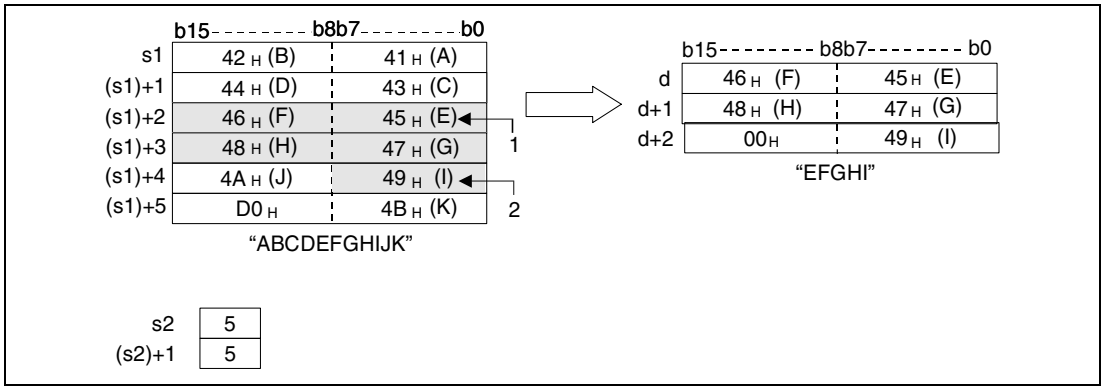
Storing and moving parts of character strings

MIDR Storing specified parts of character strings

The MIDR instruction stores a part specified from s onwards of the character string stored from d onwards.

The first character of part to be stored is specified in s2 (Array_s[1]) and is counted beginning from the left part of the character string (lower byte of s1).

The length of the part to be stored is specified in s2+1 (Array_s[2]).

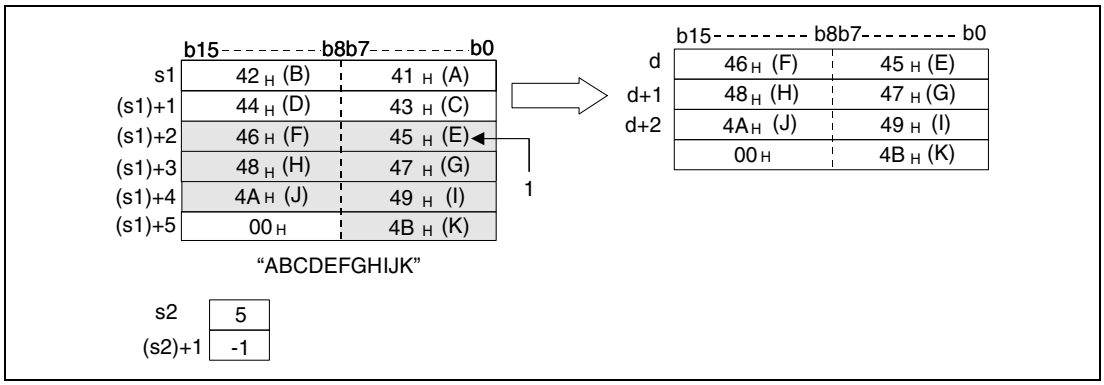


[1] Position of the 1st character (s2), 5th character of the character string

[2] Position of the last character to be stored

No operation is processed, if the number of characters in (s2)+1 (Array_s[2]) is zero.

If the value "–1" is stored in (s2)+1 (Array_s2[2]), the characters from s2 (Array_s2[1]) onwards are stored.



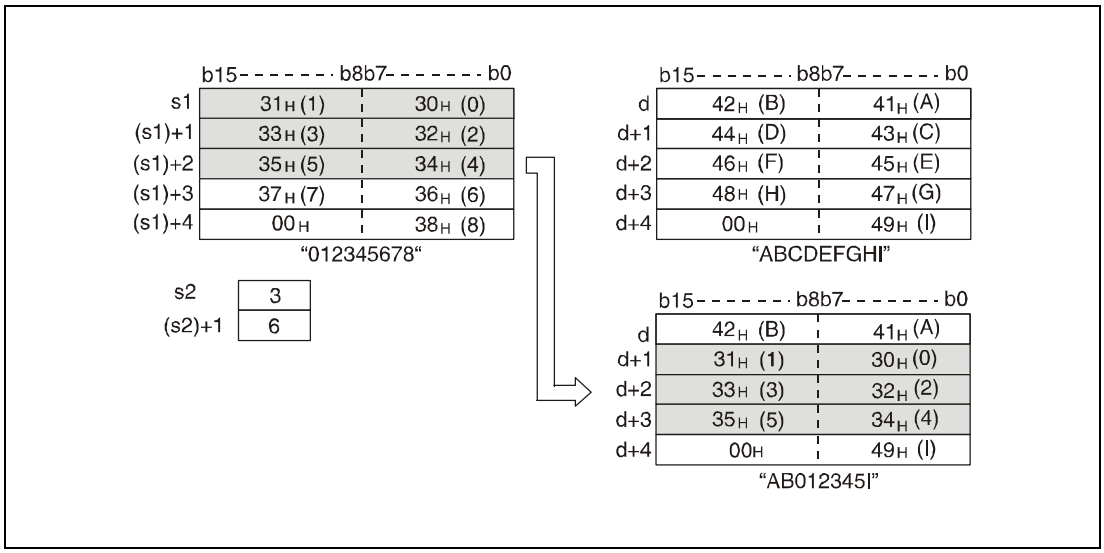
[1] Position of the 1st character (s2), 5th character of the character string

MIDW Moving parts of character string to a defined area

The MIDW instruction stores a part of specified length of the character string stored from s1 onwards in the area specified in d and d+1.

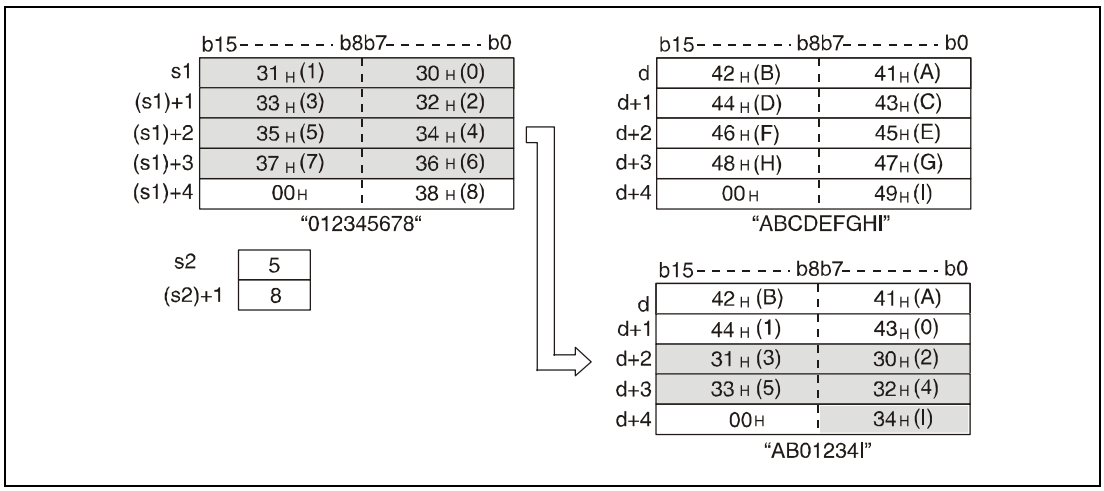
The first address of the storage area in d through d+n is specified in s2 (Array_s2[1]) and is counted beginning from the left part of the character string (lower byte of d).

The length of the part of string to be stored is specified in s2+1 (Array_s2[2]).

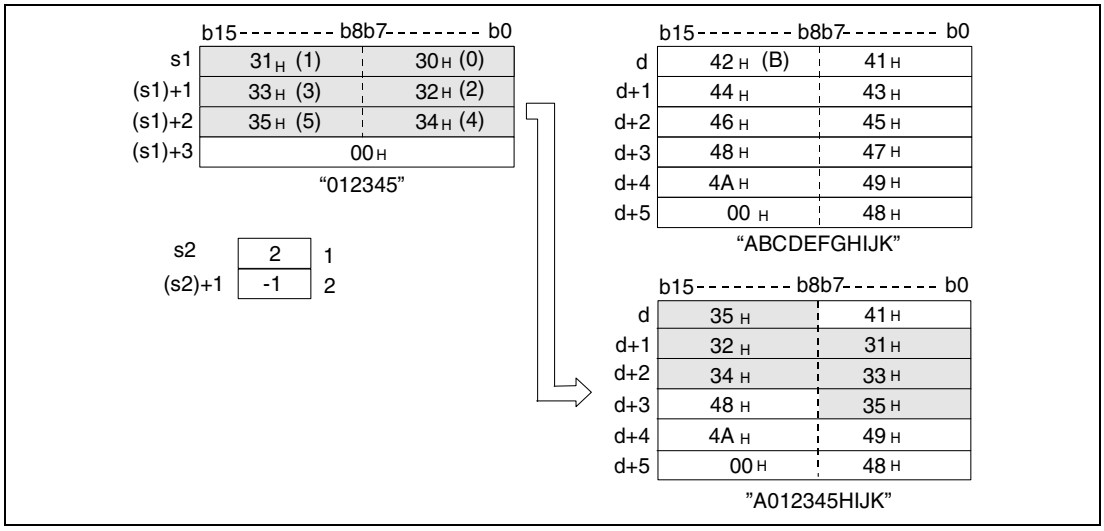


No operation is processed, if the number of characters in (s2)+1 (Array_s2[2]) is zero.

If the number of characters specified in (s2)+1 (Array_s2[2]) exceeds the storage area specified from d onwards, the remaining characters are cut off. In the following diagram the characters "35H" through "37H" are not stored.



If the value -1 is stored in (s2)+1 (Array_s2[2]), the characters are stored from s1 onwards.



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

For the MIDR instruction

- The initital device number of the characters to be stored specified in s2 (Array_s2[1]) exceeds the range from s1 to (s1)+n.
  (Error code 4101)
- The initital device number of the characters to be stored specified in (s2)+1 (Array_s2[2]) exceeds the range from d to d+n.
  (Error code 4101)
- The s2+0 value is 0.
  (Error code 4101)
- "00H" does not exist in the specified devices that follow the device specified for s1.
  (Error code 4101)

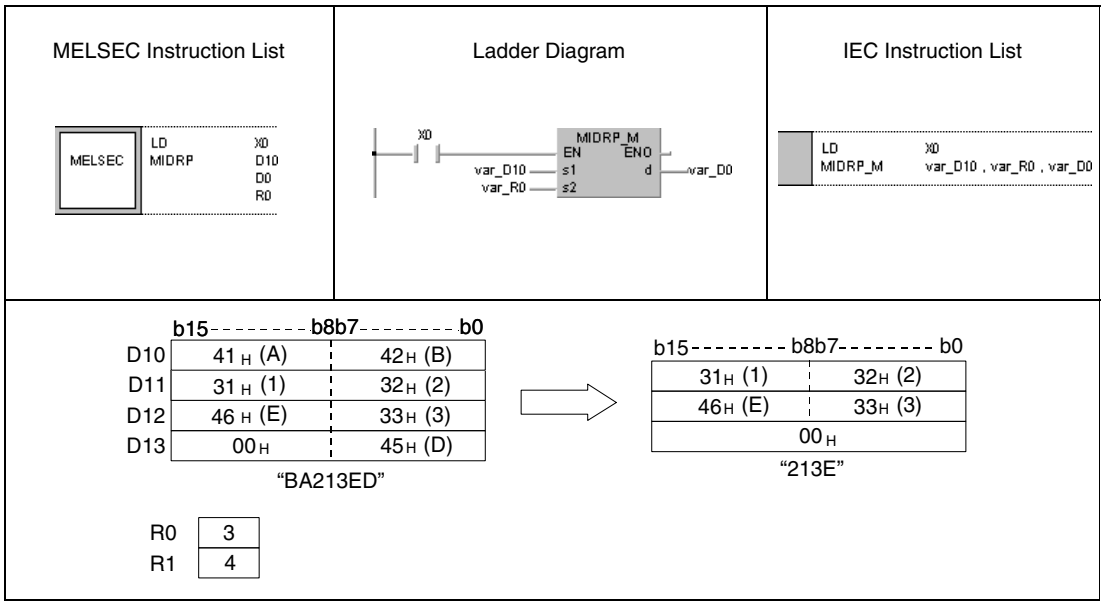
For the MIDW instruction

- The initital device number of the characters to be stored specified in (s2) (Array_s2[1]) exceeds the range from d to d+n.
  (Error code 4101)
- The initital device number of the characters to be stored specified in (s2)+1 (Array_s2[2]) exceeds the storage range in s1 through (s1)+n.
  (Error code 4101)
- The s2+0 value is 0.
  (Error code 4101)
- "00H" does not exist in the specified devices that follow the device specified for s1.
  (Error code 4101)

Program Example 1

MIDRP

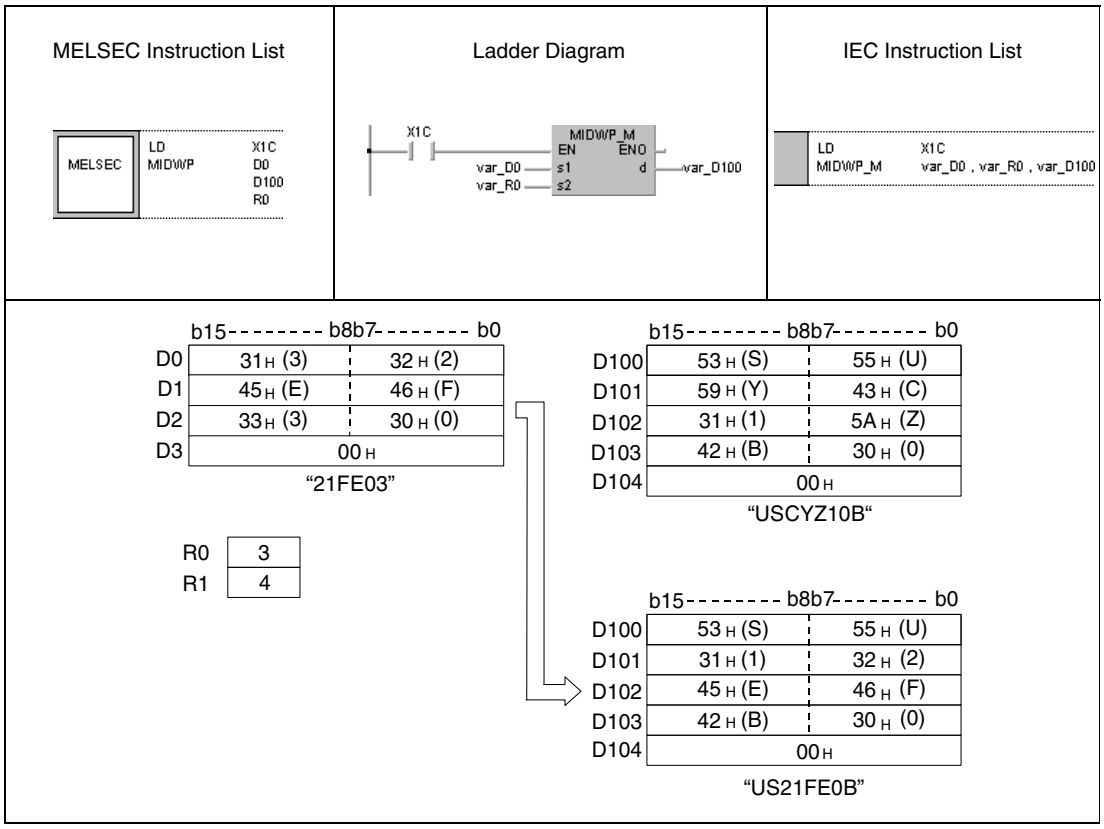
With leading edge from X0, the following program stores characters in D0 through D2 from a character string in D10 through D13. The number of characters to be stored is specified in R1 (var_R0 Array [2]). The starting position within the source string is specified in R0 (var_R0 Array [1]).



Program Example 2

MIDWP

With leading edge from X1C, the following program stores characters in D100 through D104 from the beginning of a character string in D0 through D3. The number of characters to be stored is specified in R1 (var_R0 Array [2]). The starting position where the characters are stored is specified by R0 (var_R0 Array [1]).



NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.11.17 INSTR, INSTRP

CPU

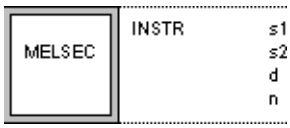
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

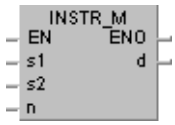
	Usable Devices									
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant		Other
	Bit	Word		Bit	Word			K, H (16#)	$	
s1	—	●	●	—	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	—	●	—
d	●	●	●	●	●	●	●	—	—	—
n	●	●	●	●	●	●	●	●	—	—

GX IEC Developer

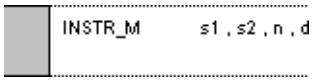
MELSEC Instruction List



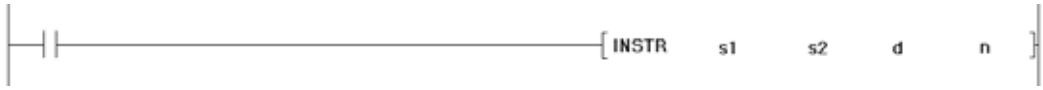
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s1	Character string to be searched for or first number of device storing the character string to be searched for	Character string
s2	Character string in which a search is performed or first number of device storing the character string data to be searched through	
d	Initial number of device storing the search result	BIN 16-bit
n	Initial position where data is searched	

Functions

Search for character strings

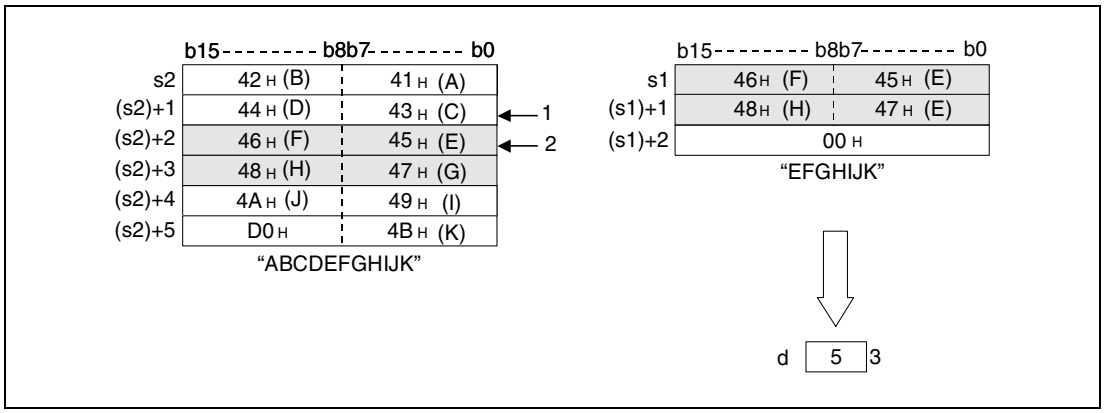
INSTR Search for character strings

The INSTR instruction searches the character string specified in s1 through (s1)+n within the character string data specified by s2 through (s2)+n.

The search begins with the character specified in n.

The first matching character is stored in d. The character is counted beginning from the left part of the character string (lower byte of s2).

– For n=3



[1] The search starts from the 3rd character

[2] First character of the searched character string

[3] Search result

If no matching character string is found, a zero is stored in d.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

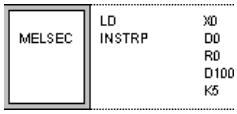
- The initial search position stored in n exceeds the range of (s2) through (s2)+n. (Error code 4100)
- 00H (NULL) does not exist within the corresponding device range after the device designated by s1, s2. (Error code 4100)
- The value of n is negative number or "0". (Error code 4100)

Program Example 1

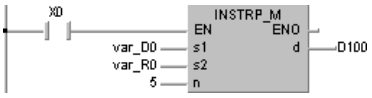
INSTRP

With leading edge from X0, the following program searches in R0 onwards beginning with the 5th character for the character string specified in D0 through D2. The result (0) is stored in D100.

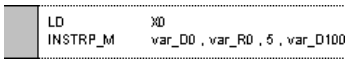
MELSEC Instruction List

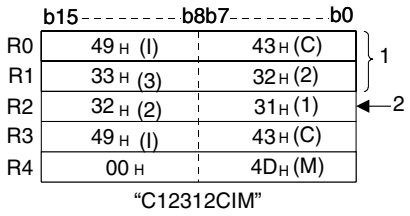


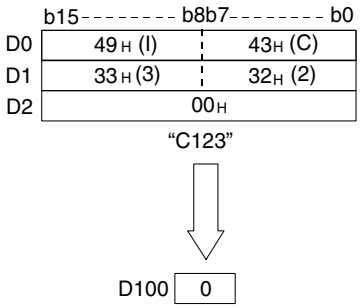
Ladder Diagram



IEC Instruction List







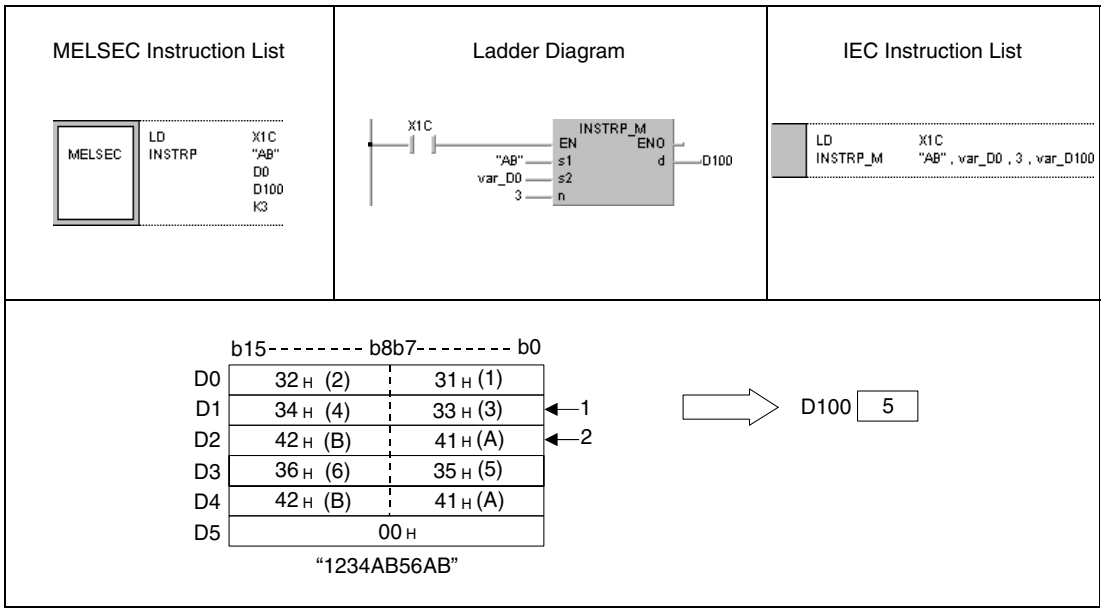
[1] This area is not searched through.

[2] The search begins with the 5th character.

Program Example 2

INSTRP

With leading edge from X0, the following program searches in D0 onwards beginning with the 3rd character for the character string "AB". The search result (5) is stored in D100.



[1] The search begins with the 3rd character.

[2] The searched character string begins at the 5th character.

NOTE

These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.11.18 STRINS, STRINSP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				● [1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

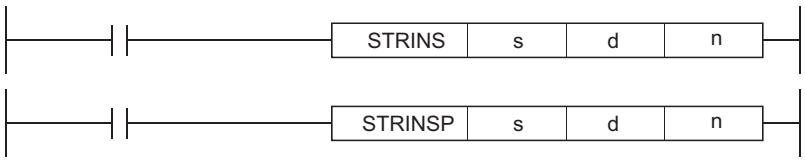
Devices

	Usable Devices									
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant		Other
	Bit	Word		Bit	Word			K, H (16#)	$	
s	—	●	●	—	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—	—
n	—	●	●	●	●	●	●	●	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s	First number of device storing the character string to be inserted	Character string
d	Initial number of device storing the insert character strings	Character string
n	Initial position where data is inserted. (Setting range: 1 <= n <= 16383)	BIN 16-bit

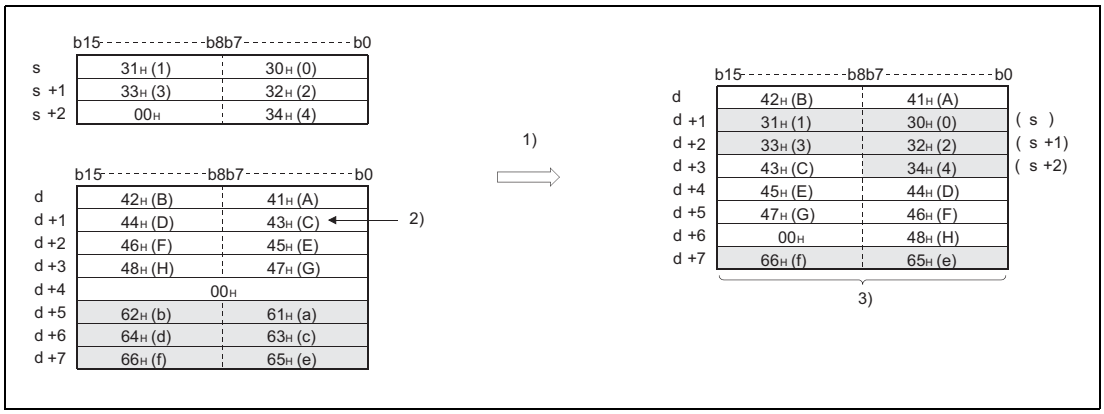
Functions

Insertion of character strings

STRINS Insertion of character strings

This instruction inserts the character string data specified by s to the nth device (insert position) from the initial character string data stored in the devices specified by d.

– For n=3



[1] Shifts the third character and up by the number of characters specified by s to the left and inserts the character string data specified by s

[2] Third character insertion position

[3] The character data stored after d+4 will be written over in accordance with the number of characters to be inserted.

This instruction stores the NULL code (00H) into the device (1 word) that positions in d after the last device where the character string data are stored, if the character string (s+d) value is even after the insertion.

This instruction stores the NULL code (00H) into the last device (high 8 bits) in d where the character string data are stored, if the character string (s+d) value is odd after the insertion.

This instruction links the device, where the character string data are stored, specified by s with the last device specified by d, if n is specified by the number of devices specified by d plus one.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

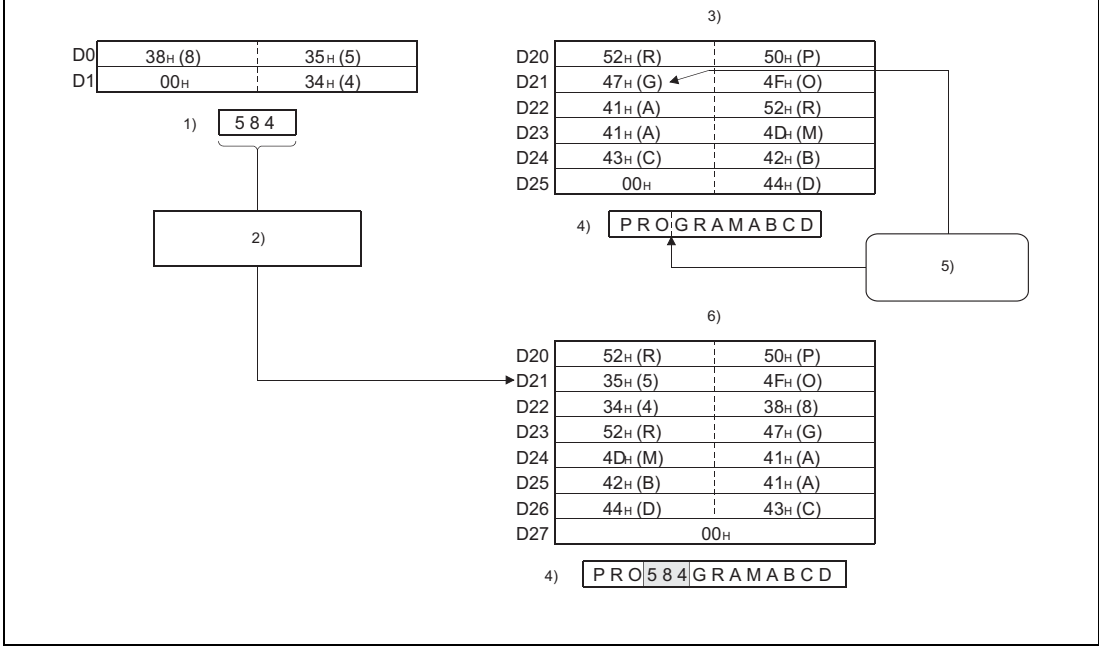
- The number of characters in the devices specified by s, d, or the devices specified by (s+d) after the insertion exceeds 16383 characters. (Error code 4100)
- The value specified by n is not within the specified range ($1 \leq n \leq 16383$). (Error code 4100)
- The value specified by n exceeds the number of the devices specified by d plus one. (Error code 4100)
- The devices, that store character strings, specified by s overlaps with even one of the devices specified by d. (Error code 4101)
- The range of the devices specified by (s+d) in which character strings data have been inserted exceeds the specified device range. (Error code 4101)
- The NULL code (00H) does not exist within the specified device range after the device specified by s or d. (Error code 4101)
- The range of the devices specified by (s+d) in which character strings data have been inserted overlaps with the range of the devices specified by s that store the character string data. (Error code 4101)

Program Example 1

STRINS

With leading edge from M0, the following program inserts the character string data stored in the devices D0 and up to the fourth device from the initial character string data stored in D20 and up.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



[1] D0 character string

[2] D20 character string inserted between"O"and"G"

[3] Before insertion

[4] D20 character string

[5] Fourth character from the left (Insert position)

[6] After insertion

7.11.19 STRDEL, STRDELP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

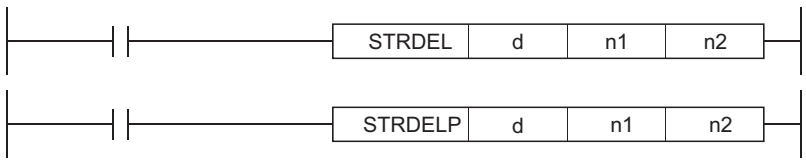
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H$	Other
	Bit	Word		Bit	Word				
d	—	●	●	—	—	—	—	—	—
n1	—	●	●	●	●	●	●	●	—
n2	—	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

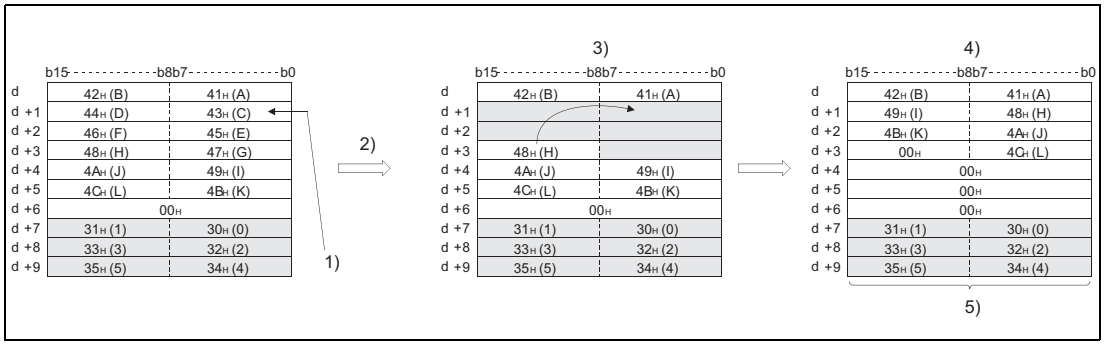
Set Data	Meaning	Data Type
d	First number of device storing the character strings to be deleted	Character string
n1	Deletion start position. (Setting range: 1 ≤ n1 ≤ 16383)	BIN 16-bit
n2	Number of characters to be deleted. (Setting range: 1 ≤ n2 ≤ 16384 – n1)	

Functions **Deletion of character strings**

STRDEL **Deletion of character strings**

This instruction deletes n2 character string data specified by d starting from the device (delete position) specified by n1.

– For n1 = 3 (device position of character string data to be deleted) and n2 = 5 (number of characters to be deleted)



[1] n1th character to be deleted

[2] Deletes n2 characters from the n1th device and up

[3] Shifts the n1+n2th characters and up, which are stored after the devices whose characters were deleted, by n2 characters to the left

[4] Stores the NULL code (00H) into the empty devices after shifting

[5] Characters of the devices other than the shifted devices do not change.

This instruction stores the NULL code (00H) into the device (1 word) that positions after the last device where the character string data are stored, if the character string specified by d is even, after the characters are deleted.

This instruction stores the NULL code (00H) into the last device (high 8 bits) where the character string data are stored, if the character string specified by d is odd, after the characters are deleted.

This instruction shifts the characters stored in the devices positioned after the deleted devices by n2 characters to the left - and then stores the NULL code (00H) into the empty device.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

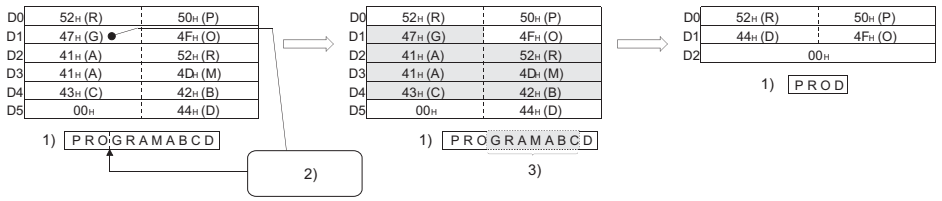
- The number of characters in the devices specified by d exceeds 16383. (Error code 4100)
- The value specified by n1 is not within the specified range ($1 \leq n1 \leq 16383$). (Error code 4100)
- The value specified by n1 exceeds the number of characters in the devices specified by d. (Error code 4100)
- The value specified by n2 exceeds the number of characters in the devices starting from n1th to the last devices position. (Error code 4100)
- The value specified by n2 is negative. (Error code 4100)

Program Example

STRDEL

With leading edge from M0, the following program deletes the seven characters starting with the fourth character in the character string data stored in the devices from D0 onward.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



[1] D0 character string

[2] Fourth character to be deleted

[3] Seven characters to be deleted

7.11.20 EMOD, EMODP

CPU

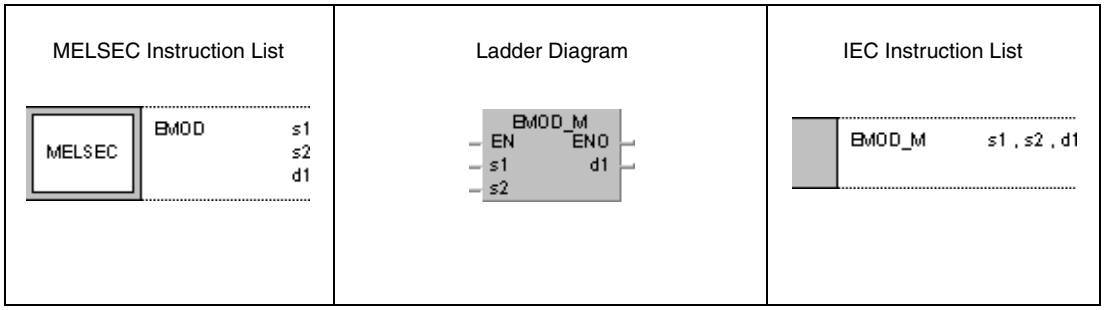
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

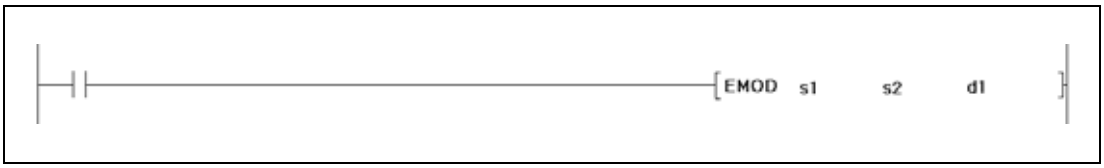
	Usable Devices									
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant		Other
	Bit	Word		Bit	Word			K, H (16#)	E	
s1	—	●	●	—	●	●	● [1)]	—	●	—
s2	●	●	●	●	●	●	●	●	—	—
d1	—	●	●	—	—	—	—	—	—	—

[1] Available only in multiple Universal model QCPU and LCPU

GX IEC Developer



GX Works2



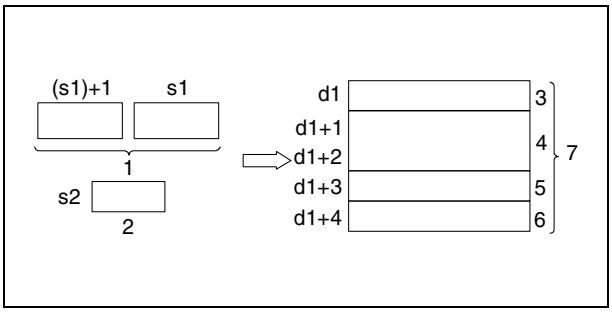
Variables

Set Data	Meaning	Data Type
s1	Floating point data (data type REAL) or first number of device storing the floating point data	REAL
s2	Number of digits the floating point is moved to the right or first number of device storing such data	BIN 16-bit
d1	First number of device storing the floating point number in BCD data format	

Functions **Conversion of floating point number into the BCD format**

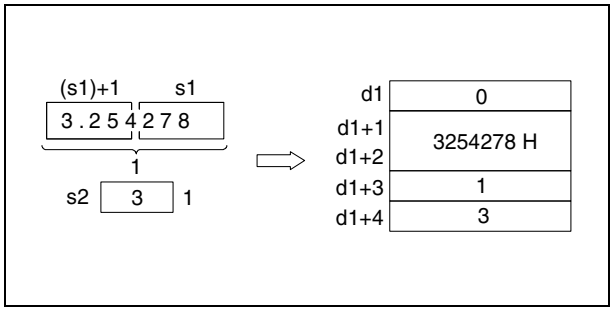
EMOD Conversion into the BCD format

The EMOD instruction calculates the BCD format from the floating point number (real number) in s1 and (s1)+1 considering the decimal point shift to the right specified in s2. The result is stored in d1 through (d1)+4.

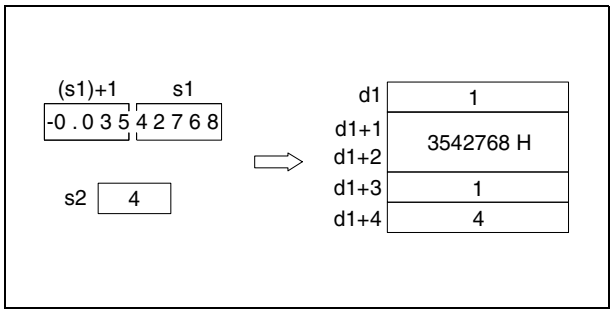


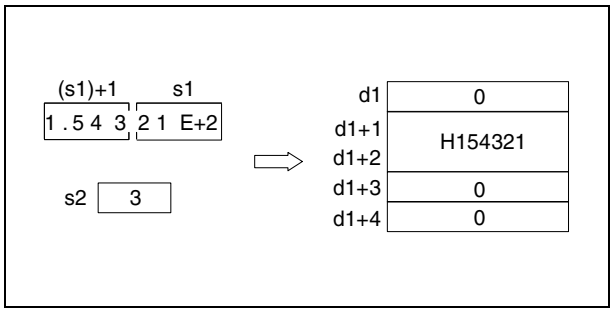
[1] Floating point data (data type REAL)
[2] Shift of the decimal point to the right
[3] Sign bit (0 = positive / 1 = negative)
[4] 7 BCD digits
[5] Exponent sign (0 = positive / 1 = negative)
[6] BCD exponent (Value range 0 to 38)
[7] Floating point number in BCD data format

The following diagrams show conversion examples.

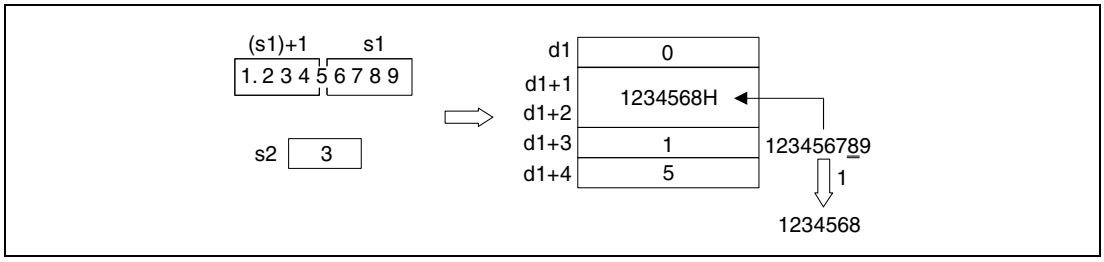


[1] Floating point data (data type REAL)





The floating point number in s1 and (s1)+1 is rounded up to 7 digits and stored in (d1)+1 and (d1)+2.



[1] Rounded up

Operation Errors

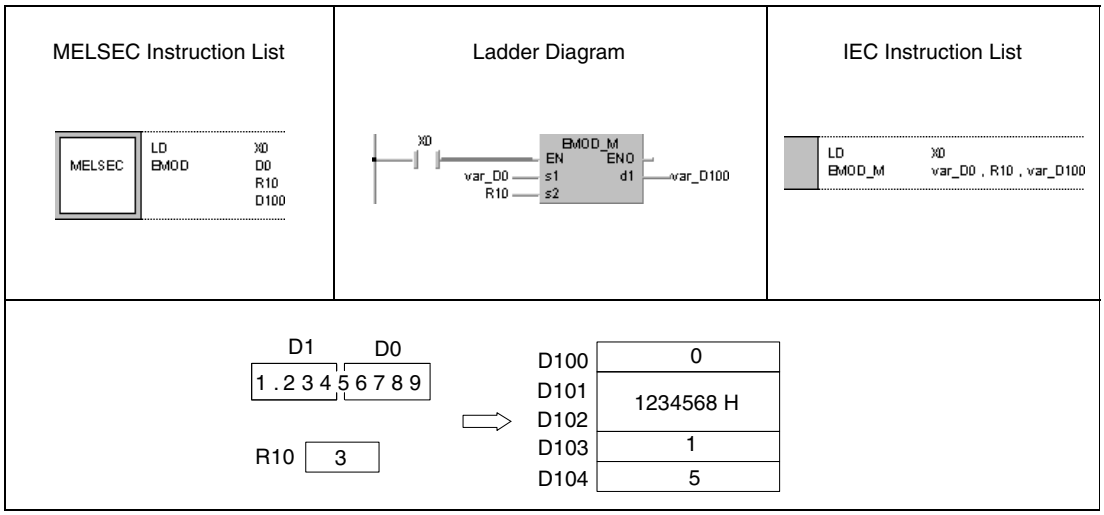
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of digits of the decimal point shift (s2) exceeds the range of 0 to 7.
  (Error code 4100)
- The value entered in d1 through (d1)+4 exceeds the relevant storage device area.
  (Error code 4101)
- The 32-bit floating point real number specified in s1 is not zero and not within the following range:
  $\pm 2^{-126} \leq (s1) < \pm 2^{128}$
  (Error code 4100)
- The device specified by d1 exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

EMOD

While X0 is set, the following program converts the floating point data (data type REAL) specified in D0 and D1 considering the decimal point shift specified in R10. The result is stored in D100 through D104.



NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.11.21 EREXP, EREXPP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

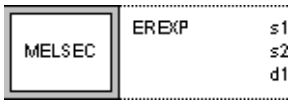
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	●	●	●	●	●	●	●	●	—
d1	—	●	●	—	●	●	● [1)]	—	—

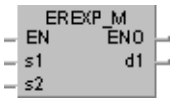
[1] Available only in multiple Universal model QCPU and LCPU

GX IEC Developer

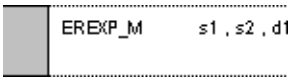
MELSEC Instruction List



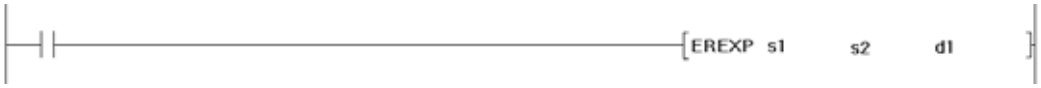
Ladder Diagram



IEC Instruction List



GX Works2



Variables

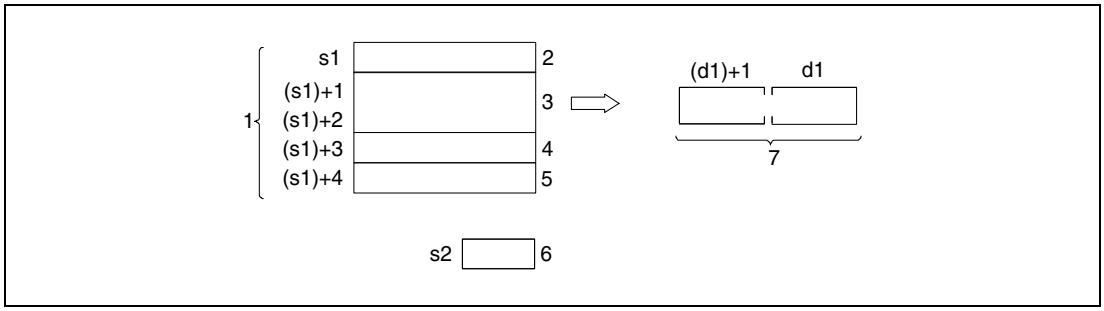
Set Data	Meaning	Data Type
s1	First number of device storing floating point data in BCD data format	BIN 16-bit
s2	Specification of decimal places or device storing such data	
d1	Device storing floating point data (data type REAL)	REAL

Functions

Conversion of floating point data into the decimal format

EREXP Conversion into the decimal format

The EREXP instruction calculates the decimal format of the floating point data (real number) from the floating point data in BCD format in s1 through (s1)+4, considering the decimal places specified in s2. The result is stored in d1 and (d1)+1.

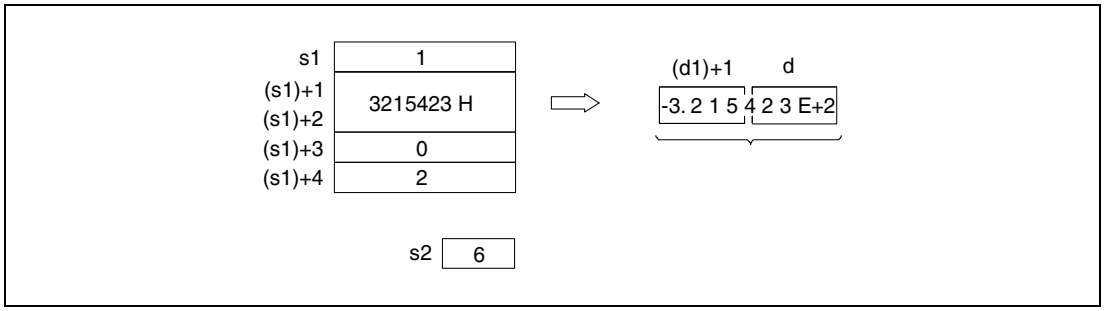


[1] Floating point data in BCD data format
[2] Sign bit (0 = positive / 1 = negative)
[3] 7 BCD digits
[4] Exponent sign (0 = positive / 1 = negative)
[5] BCD exponent (value range 0 to 38)
[6] Number of decimal places (value range 0 to 7)
[7] Floating point data (real number)

The sign in s1 and the sign of the exponent in (s1)+3 is set to 0 for a positive value. For a negative value the sign bit is 1.

The value of the BCD exponent (s1)+4 may range from 0 to 7.

The decimal places in s2 may range from 0 to 7.



Operation Errors

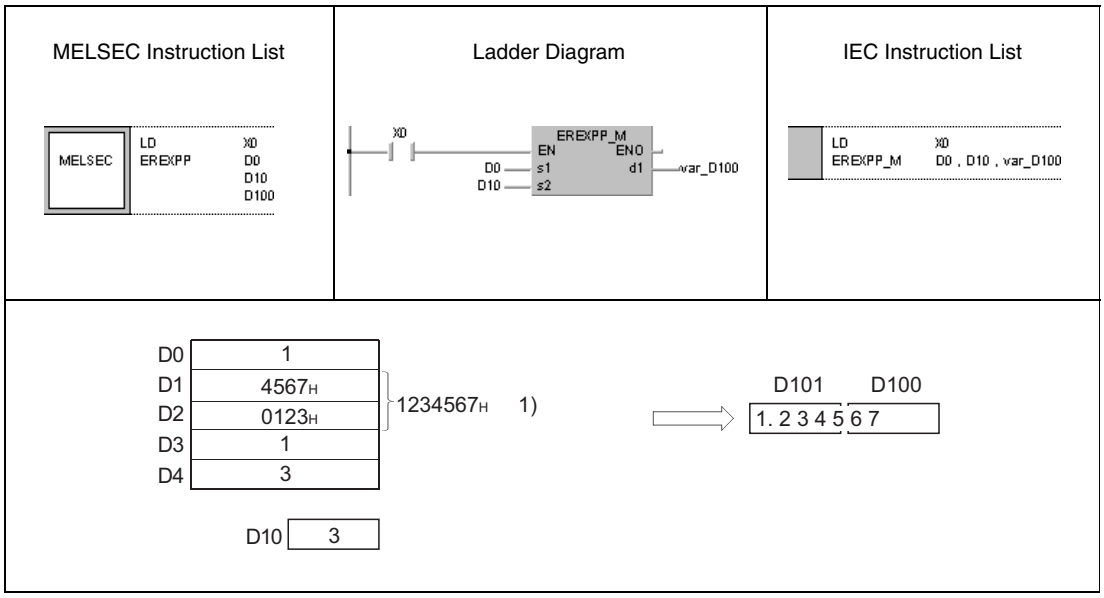
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The sign designation in s1 is not 0 or 1.
  (Error code 4100)
- The BCD data in (s1)+1 and (s1)+2 contains more than 8 digits.
  (Error code 4100)
- The exponent sign in (s1)+3 is not 0 or 1.
  (Error code 4100)
- The exponent data in (s1)+4 exceeds the range from 0 to 38.
  (Error code 4100)
- The number of decimal places in s2 exceeds the range of 0 to 7.
  (Error code 4101)
- The device specified by s1 exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

EREXPP

With leading edge from X0, the following program calculates the floating point value (real number) in decimal format from the floating point value in BCD format specified in D0 through D4 considering the decimal places specified in D10. The result is stored in D100 and D101.



[1] BCD 7 digits

7.12 Special functions

Function		MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Sine calculation	Single precision (32-bit floating-point number)	SIN	SIN_MD
			SIN_E_MD
		SINP	SIN_P_MD
			SIN_P_E_MD
	Double precision (64-bit floating-point number)	SIND	
		SINDP	
Cosine calculation	Single precision (32-bit floating-point number)	COS	COS_MD
			COS_E_MD
		COSP	COS_P_MD
			COS_P_E_MD
	Double precision (64-bit floating-point number)	COSD	
		COSDP	
Tangent calculation	Single precision (32-bit floating-point number)	TAN	TAN_MD
			TAN_E_MD
		TANP	TAN_P_MD
			TAN_P_E_MD
	Double precision (64-bit floating-point number)	TAND	
		TANDP	
Arcus sine calculation	Single precision (32-bit floating-point number)	ASIN	ASIN_MD
			ASIN_E_MD
		ASINP	ASIN_P_MD
			ASIN_P_E_MD
	Double precision (64-bit floating-point number)	ASIND	
		ASINDP	
Arcus cosine calculation	Single precision (32-bit floating-point number)	ACOS	ACOS_MD
			ACOS_E_MD
		ACOSP	ACOS_P_MD
			ACOS_P_E_MD
	Double precision (64-bit floating-point number)	ACOSD	
		ACOSDP	

Function		MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Arcus tangent calculation	Single precision (32-bit floating-point number)	ATAN	ATAN_MD
			ATAN_E_MD
		ATANP	ATAN_P_MD
			ATAN_P_E_MD
	Double precision (64-bit floating-point number)	ATAND	
		ATANDP	
Conversion from degrees into radian	Single precision (32-bit floating-point number)	RAD	RAD_MD
			RAD_E_MD
		RADP	RAD_P_MD
			RAD_P_E_MD
	Double precision (64-bit floating-point number)	RADD	
		RADDP	
Conversion from radian into degree	Single precision (32-bit floating-point number)	DEG	DEG_MD
			DEG_E_MD
		DEGP	DEG_P_MD
			DEG_P_E_MD
	Double precision (64-bit floating-point number)	DEGD	
		DEGDP	
Exponentiation	Single precision (32-bit floating-point number)	POW	
		POWP	
	Double precision (64-bit floating-point number)	POWD	
		POWDP	
Square root	Single precision (32-bit floating-point number)	SQR	SQR_MD
			SQR_E_MD
		SQRP	SQR_P_MD
			SQR_P_E_MD
	Double precision (64-bit floating-point number)	SQRD	
		SQRDP	

Function		MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Floating point value as exponent of e	Single precision (32-bit floating-point number)	EXP	EXP_MD
			EXP_E_MD
		EXPP	EXP_P_MD
			EXP_P_E_MD
	Double precision (64-bit floating-point number)	EXPD	
		EXPDP	
Logarithm (natural) calculation	Single precision (32-bit floating-point number)	LOG	LOG_MD
			LOG_E_MD
		LOGP	LOG_P_MD
			LOG_P_E_MD
	Double precision (64-bit floating-point number)	LOGD	
		LOGDP	
Common logarithm	Single precision (32-bit floating-point number)	LOG10	
		LOG10P	
	Double precision (64-bit floating-point number)	LOG10D	
		LOG10DP	
Randomize value		RND	RND_M
		RNDP	RNDP_M
Update random values		SRND	SRND_M
		SRNDP	SRNDP_M
Square root calculation from 4-digit BCD data		BSQR	BSQR_MD
			BSQR_K_MD
		BSQRP	BSQR_P_MD
			BSQR_K_P_MD
Square root calculation from 8-digit BCD data		BDSQR	BDSQR_MD
			BDSQR_K_MD
		BDSQRP	BDSQR_P_MD
			BDSQR_K_P_MD
Sine calculation from BCD data		BSIN	BSIN_MD
			BSIN_K_MD
		BSINP	BSIN_P_MD
			BSIN_K_P_MD
Cosine calculation from BCD data		BCOS	BCOS_MD
			BCOS_K_MD
		BCOSP	BCOS_P_MD
			BCOS_K_P_MD

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Tangent calculation from BCD data	BTAN	BTAN_MD
		BTAN_K_MD
	BTANP	BTAN_P_MD
		BTAN_K_P_MD
Arcus sine calculation from BCD data	BASIN	BASIN_MD
	BASINP	BASIN_P_MD
Arcus cosine calculation from BCD data	BACOS	BACOS_MD
	BACOSP	BACOS_P_MD
Arcus tangent calculation from BCD data	BATAN	BATAN_MD
	BATANP	BATAN_P_MD

NOTE *Within the IEC editors please use the IEC instructions.*

7.12.1 SIN, SINP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
● [1)]	●	●	●	●	●

[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

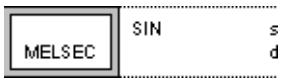
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	● [1)]	●	—
d	—	●	●	—	●	●	● [1)]	—	—

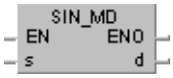
[1] Applicable for the Universal model QCPU, LCPU

GX IEC Developer

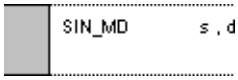
MELSEC Instruction List



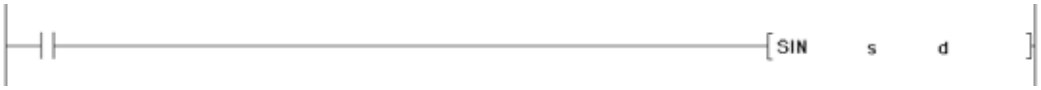
Ladder Diagram



IEC Instruction List



GX Works2



Variables

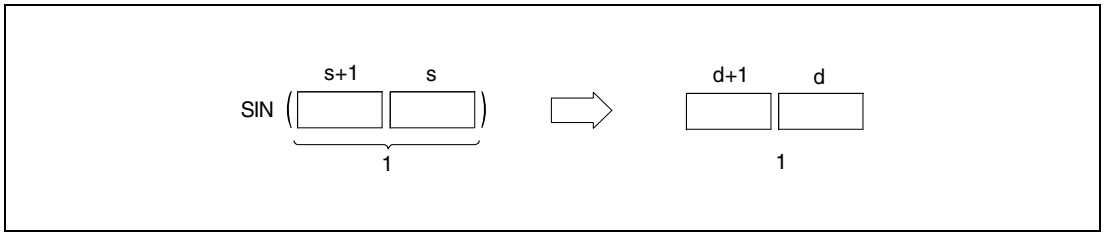
Set Data	Meaning	Data Type
s	Angle data for the SIN (sine) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions

Sine calculation from floating-point values (Single precision)

SIN Sine calculation

The SIN instruction calculates the sine value from angle data in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

The angle in s and s+1 must be specified in radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RAD and DEG instructions.

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is –0.
  (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (overflow occurs):
  (For the Universal model QCPU, LCPU)
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

SIN

The following program calculates the sine value from the 4-digit BCD angle specification in X20 through X2F. The result is stored as 32-bit floating-point value (real number) in D0 and D1.

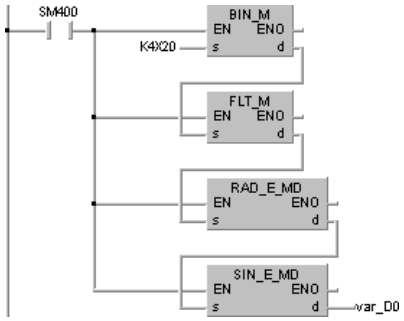
MELSEC Instruction List

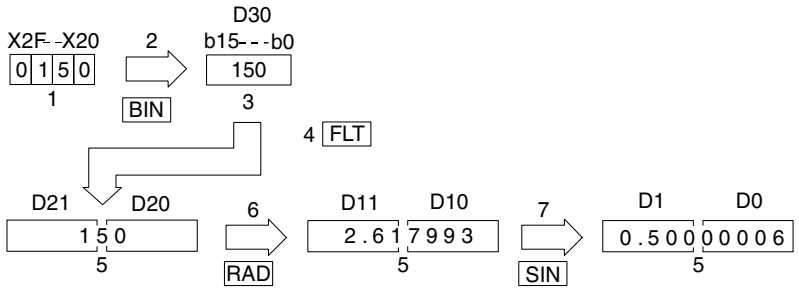
```
MELSEC
        LD      SM400
        BIN     K4X20
                D30
        FLT     D30
                D20
        RAD     D20
                D10
        SIN     D10
                D0
```

IEC Instruction List

```
LD          SM400
BIN_M       K4X20 , D30
FLT_M       D30 , var_D20
RAD_E_MD    var_D20 , var_D10
SIN_E_MD    var_D10 , var_D0
```

Ladder Diagram





[1] BCD value

[2] Conversion into the BIN format

[3] BIN value

[4] Conversion into the floating-point format

[5] 32-bit floating point value (real number)

[6] Conversion into the radian measure

[7] Calculation of the sine value

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.2 SIND, SINDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

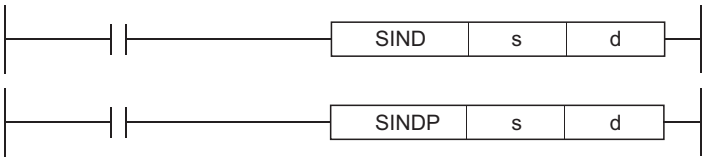
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



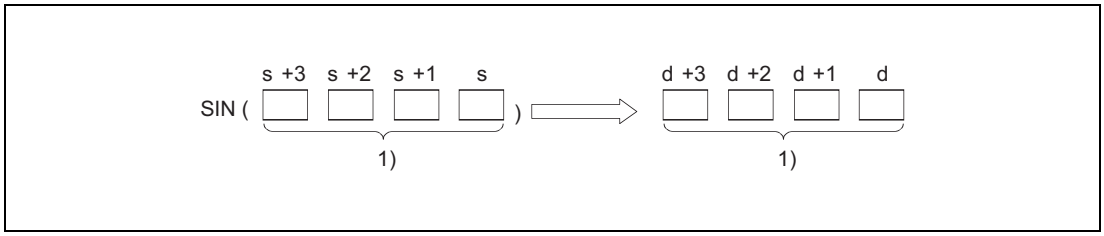
Variables

Set Data	Meaning	Data Type
s	Angle data for the SIND (sine) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions **Sine calculation from floating-point values (Double precision)**

SIND Sine calculation

The SIND instruction calculates the sine value from angle data specified by s. The result is stored into the device specified by d.



[1] 64-bit floating-point value (real number)

The angle in s must be specified in radian measure (degrees x π/180).
The conversion from degrees into radian is described in the sections on the RADD and DEGD instructions.

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (Operation results in an overflow):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

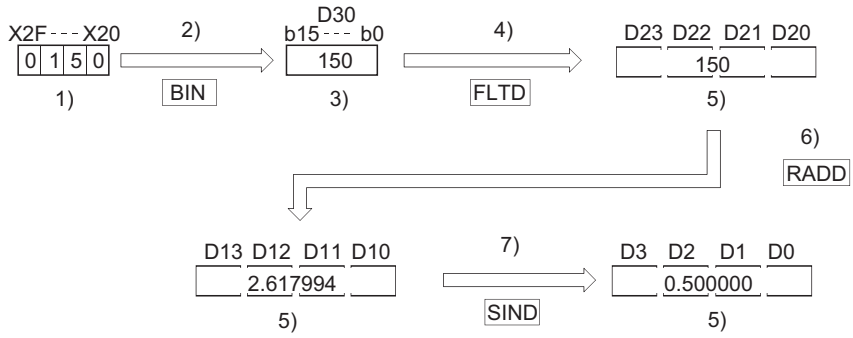
SIND

The following program calculates the sine value from the 4-digit BCD angle specification in X20 through X2F. The result is stored as 64-bit floating-point value (real number) in D0 to D3.

MELSEC Instruction List

IEC Instruction List

Ladder Diagram



[1] BCD value
[2] Conversion into the BIN format
[3] BIN value
[4] Conversion into the floating-point format
[5] 64-bit floating-point value (real number)
[6] Conversion into the radian measure
[7] Calculation of the sine value

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.3 COS, COSP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

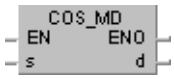
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Applicable for the Universal model QCPU, LCPU

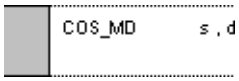
GX IEC Developer

MELSEC Instruction List

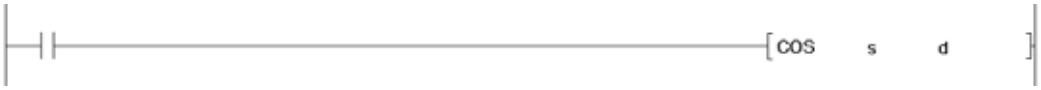
Ladder Diagram



IEC Instruction List



GX Works2



Variables

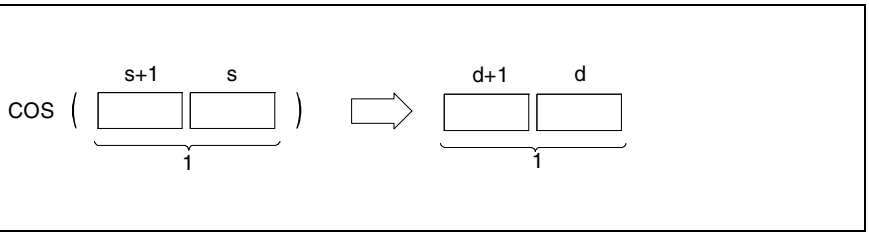
Set Data	Meaning	Data Type
s	Angle data for the COS (cosine) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions

Cosine calculation from floating-point values (Single precision)

COS Cosine calculation

The COS instruction calculates the cosine value from angle data in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

The angle in s and s+1 must be specified in radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RAD and DEG instructions.

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is –0.
  (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (overflow occurs):
  (For the Universal model QCPU, LCPU)
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

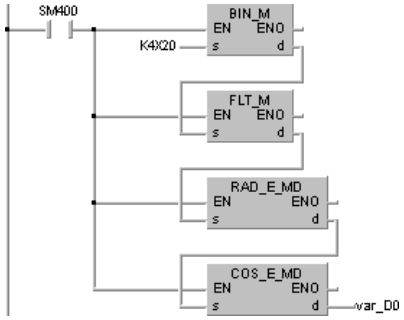
COS

The following program calculates the cosine value from the 4-digit BCD angle specification in X20 through X2F. The result is stored as 32-bit floating-point value (real number) in D0 and D1.

MELSEC Instruction List

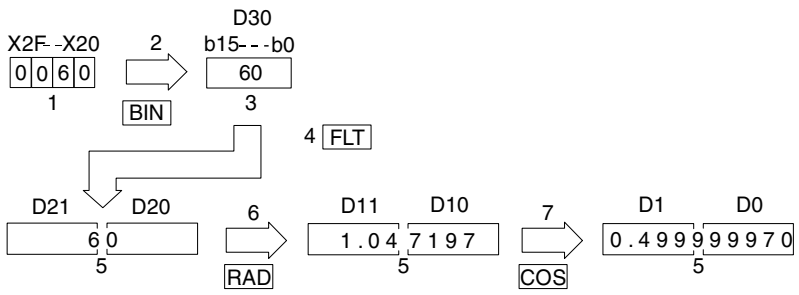
```
MELSEC
        LD      SM400
        BIN     K4X20
                D30
        FLT     D30
                D20
        RAD     D20
                D10
        COS     D10
                D0
```

Ladder Diagram



IEC Instruction List

```
        LD      SM400
        BIN_M   K4X20 , D30
        FLT_M   D30 , var_D20
        RAD_E_MD  var_D20 , var_D10
        COS_E_MD  var_D10 , var_D0
```



[1] BCD value

[2] Conversion into the BIN format

[3] Binary value

[4] Conversion into the floating-point format

[5] 32-bit floating point value (real number)

[6] Conversion into the radian measure

[7] Calculation of the cosine value

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.4 COSD, COSDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

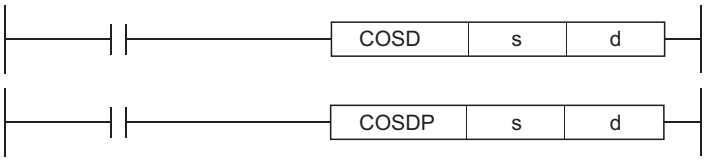
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



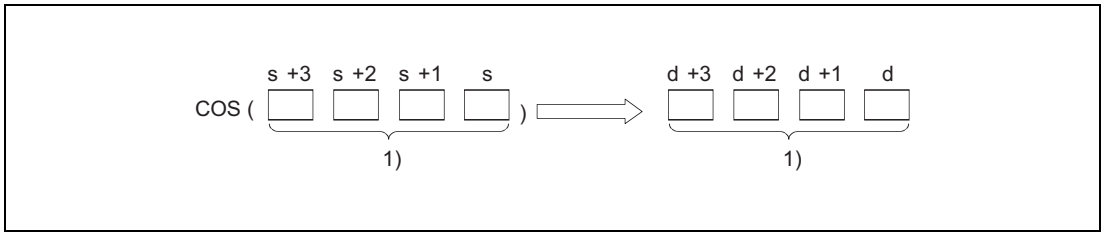
Variables

Set Data	Meaning	Data Type
s	Angle data for the COSD (cosine) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions **Cosine calculation from floating-point values (Double precision)**

COSD Cosine calculation

The COSD instruction calculates the cosine value from angle data specified by s. The result is stored in the device specified by d.



[1] 64-bit floating-point value (real number)

The angle in s must be specified in radian measure (degrees x π/180).
The conversion from degrees into radian is described in the sections on the RADD and DEGD instructions.

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

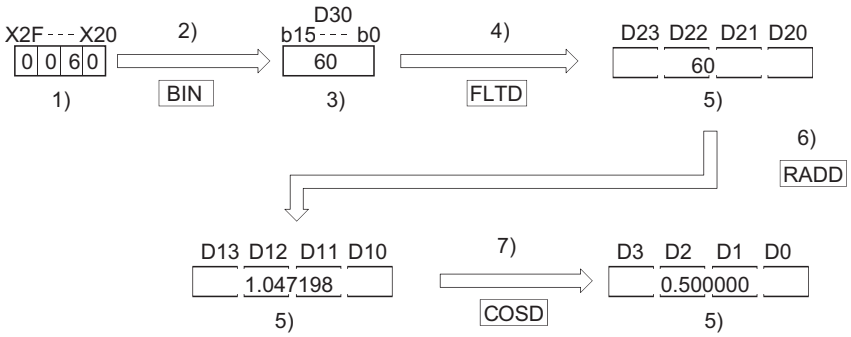
COSD

The following program calculates the cosine value from the 4-digit BCD angle specification in X20 through X2F. The result is stored as 64-bit floating-point value (real number) in D0 to D3.

MELSEC Instruction List

Ladder Diagram

IEC Instruction List



[1] BCD value
[2] Conversion into the BIN format
[3] Binary value
[4] Conversion into the floating-point format
[5] 64-bit floating-point value (real number)
[6] Conversion into the radian measure
[7] Calculation of the cosine value

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.5 TAN, TANP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

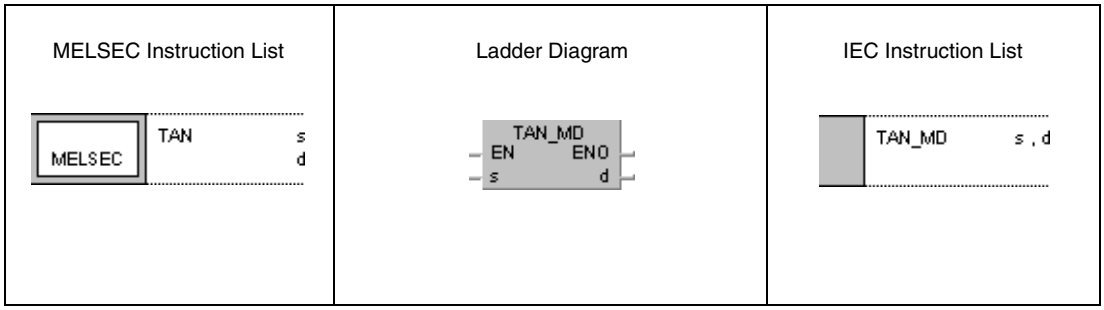
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

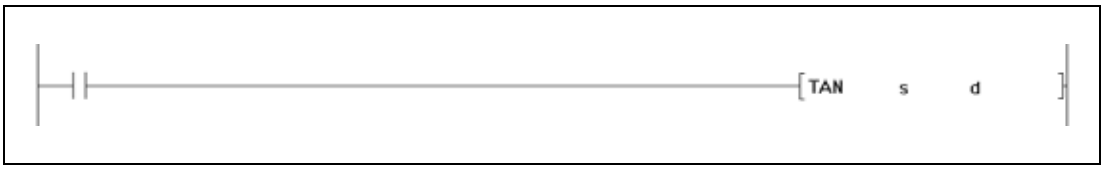
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Applicable for the Universal model QCPU, LCPU

GX IEC Developer



GX Works2



Variables

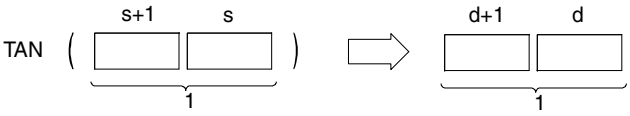
Set Data	Meaning	Data Type
s	Angle data for the TAN (tangent) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions

Tangent calculation from floating-point values (Single precision)

TAN Tangent calculation

The TAN instruction calculates the tangent value from angle data in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

The angle in s and s+1 must be specified in radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RAD and DEG instructions.

If the angle in s and s+1 retains the values π/2 rad or (3/2)xπ rad, an error message is returned from the radian measure calculation.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The operation result is not zero and not within the range from $\pm 2^{-126}$ to $\pm 2^{128}$.
  (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
- The value of the specified device is –0.
  (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (overflow occurs):
  (For the Universal model QCPU, LCPU)
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

TAN

The following program calculates the tangent value from the 4-digit BCD angle specification in X20 through X2F. The result is stored as 32-bit floating-point value (real number) in D0 and D1.

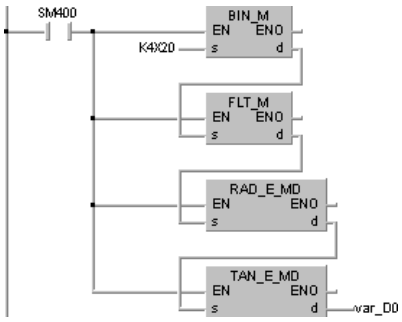
MELSEC Instruction List

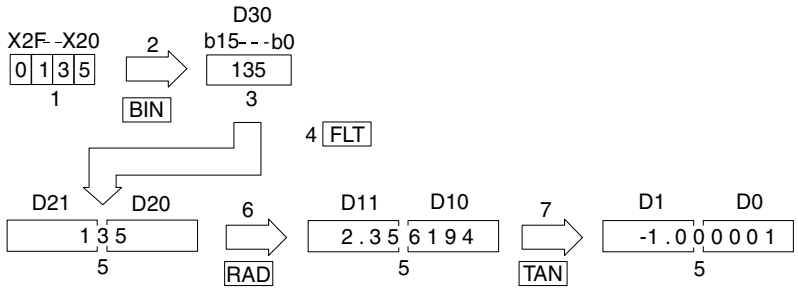
```
MELSEC
        LD      SM400
        BIN     K4X20
                D30
        FLT     D30
                D20
        RAD     D20
                D10
        TAN     D10
                D0
```

IEC Instruction List

```
LD      SM400
BIN_M   K4X20 , D30
FLT_M   D30 , var_D20
RAD_E_MD    var_D20 , var_D10
TAN_E_MD    var_D10 , var_D0
```

Ladder Diagram





[1] BCD value

[2] Conversion into the BIN format

[3] Binary value

[4] Conversion into the floating-point format

[5] 32-bit floating point value (real number)

[6] Conversion into the radian measure

[7] Calculation of the tangent value

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.6 TAND, TANDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

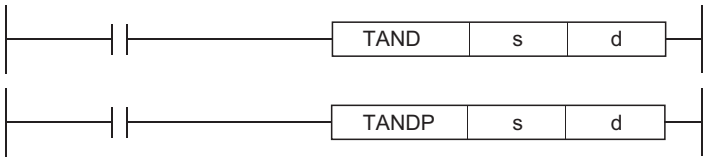
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



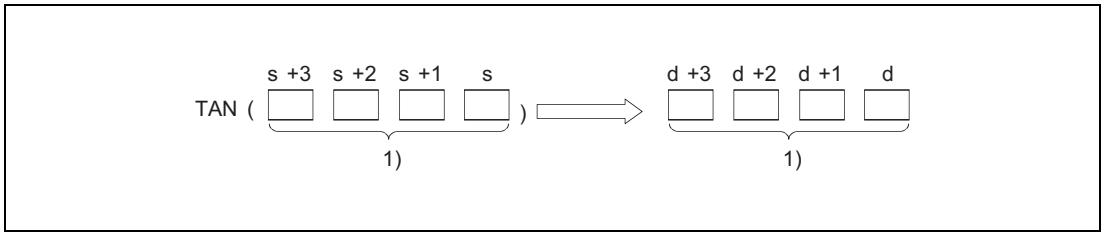
Variables

Set Data	Meaning	Data Type
s	Angle data for the TAND (tangent) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result.	

Functions **Tangent calculation from floating-point values (Double precision)**

TAND Tangent calculation

The TAND instruction calculates the tangent value from angle data specified by s. The result is stored in the device specified by d.



[1] 64-bit floating-point value (real number)

The angle in s must be specified in radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RADD and DEGD instructions.

If the angle in s retains the values π/2 rad or (3/2)xπ rad, an error message is returned from the radian measure calculation.

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Errors In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

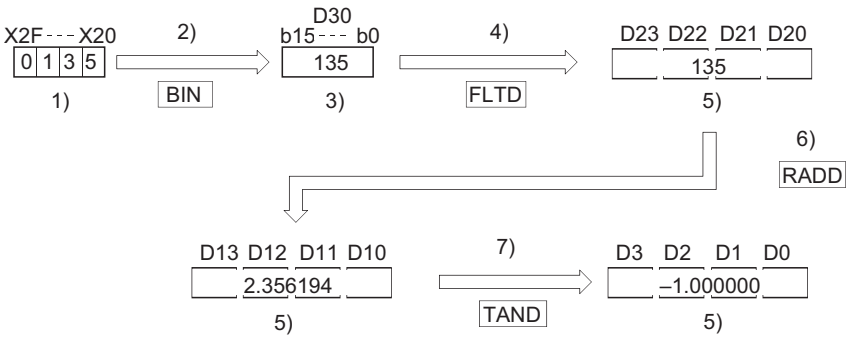
TAND

The following program calculates the tangent value from the 4-digit BCD angle specification in X20 through X2F. The result is stored as 64-bit floating-point value (real number) in D0 to D3.

MELSEC Instruction List

IEC Instruction List

Ladder Diagram



[1] BCD value
[2] Conversion into the BIN format
[3] Binary value
[4] Conversion into the floating-point format
[5] 64-bit floating-point value (real number)
[6] Conversion into the radian measure
[7] Calculation of the tangent value

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.7 ASIN, ASINP

CPU

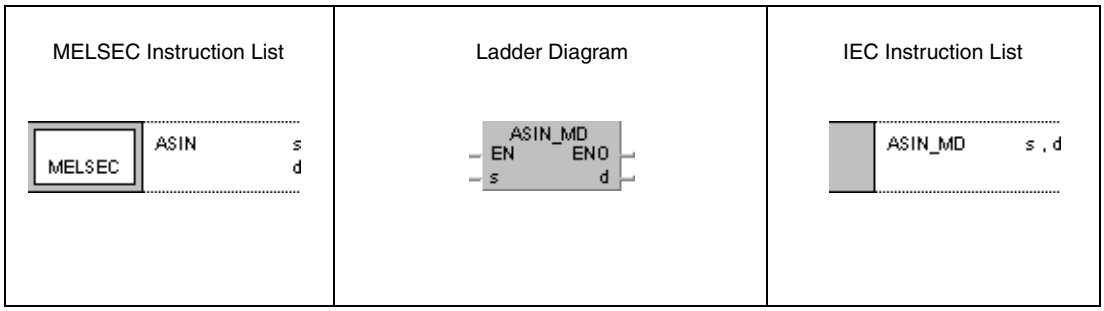
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

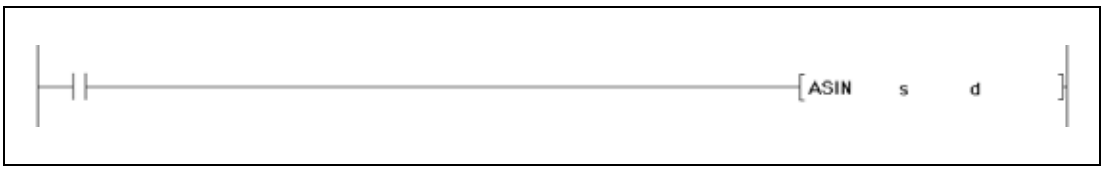
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Applicable for the Universal model QCPU, LCPU

GX IEC Developer



GX Works2



Variables

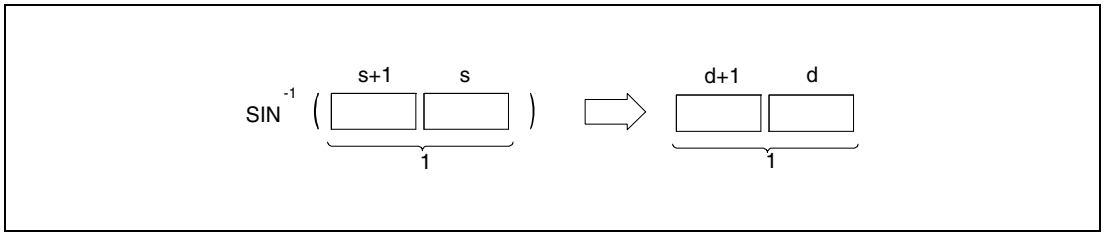
Set Data	Meaning	Data Type
s	SIN value for the ASIN (arcus sine) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions

Arcus sine calculation of floating-point values (Single precision)

ASIN Arcus sine calculation

The ASIN instruction calculates the angle from the sine value in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

The sine value in s and s+1 may range within the value range of -1 to 1.

The angle (operation result) at d is stored in radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RAD and DEG instructions.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value in s and s+1 exceeds the value range of -1 to 1. (Error code 4100)
- The contents of the specified device is not zero and not within the range from $\pm 2^{-126}$ to $\pm 2^{128}$. (For the Universal model QCPU, LCPU) (Error code 4100)
- The value of the specified device is –0.
  (For the High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (Operation results in an overflow)
  (For the Universal model QCPU, LCPU)
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

ASIN

The following program calculates the arcus sine value from the 32-bit floating-point data (real number) in D0 and D1. The resulting angle in radian measure is output at Y40 through Y4F as 4-digit BCD value.

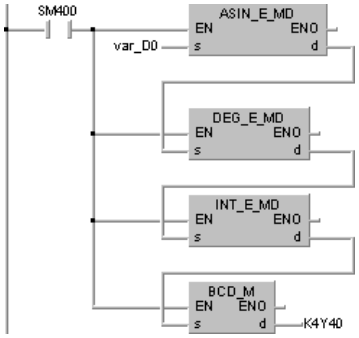
MELSEC Instruction List

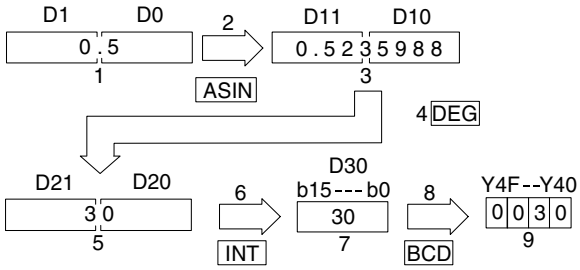
```
MELSEC   LD     SM400
         ASIN   D0
                D10
         DEG    D10
                D20
         INT    D20
                D30
         BCD    D30
                K4Y40
```

IEC Instruction List

```
LD          SM400
ASIN_E_MD   var_D0 , var_D10
DEG_E_MD    var_D10 , var_D20
INT_E_MD    var_D20 , D30
BCD_M       D30 , K4Y40
```

Ladder Diagram





[1] 32-bit floating point value (real number)
[2] Arcus sine calculation
[3] 32-bit floating point value (real number)
[4] Conversion of the angle measures
[5] 32-bit floating point value (real number)
[6] Conversion into the BIN format
[7] Binary value
[8] Conversion into the BCD format
[9] BCD value

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.8 ASIND, ASINDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

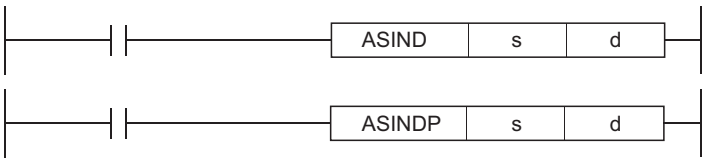
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



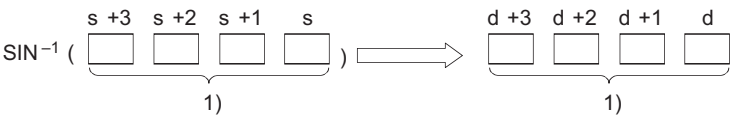
Variables

Set Data	Meaning	Data Type
s	SIN value for the ASIND (arcus sine) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions **Arcus sine calculation of floating-point values (Double precision)**

ASIND Arcus sine calculation

The ASIND instruction calculates the angle from the sine value specified by s. The result is stored in the device specified by d.



[1] 64-bit floating-point value (real number)

The sine value in s may range within the value range of -1 to 1.

The angle (operation result) in d is stored in radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RADD and DEGD instructions.

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The value specified by s is within the double-precision floating-point range and outside the range of -1.0 to 1.0. (Error code 4100)
- The result exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

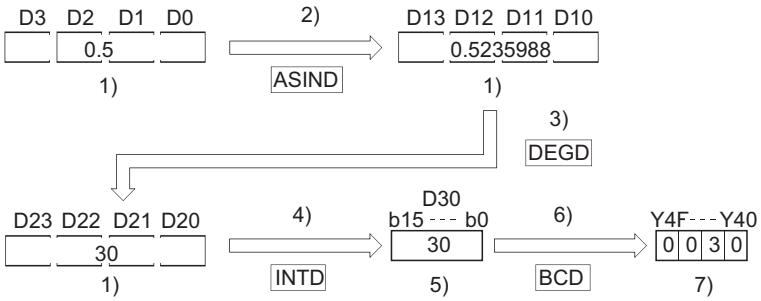
ASIND

The following program calculates the arcus sine value from the 64-bit floating-point data (real number) in D0 to D3. The resulting angle in radian measure is output at Y40 through Y4F as 4-digit BCD value.

MELSEC Instruction List

IEC Instruction List

Ladder Diagram



[1] 64-bit floating-point value (real number)
[2] Arcus sine calculation
[3] Conversion of the angle measures
[4] Conversion into the BIN format
[5] Binary value
[6] Conversion into the BCD format
[7] BCD value

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.9 ACOS, ACOSP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

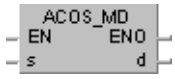
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Applicable for the Universal model QCPU, LCPU

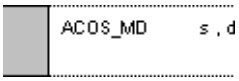
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

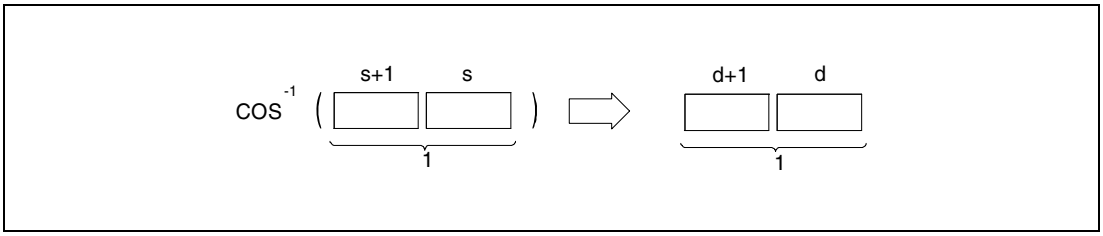
Set Data	Meaning	Data Type
s	COS value for the ACOS (arcus cosine) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions

Arcus cosine calculation of floating-point values (Single precision)

ACOS Arcus cosine calculation

The ACOS instruction calculates the angle from the cosine value in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

The cosine value in s and s+1 may range within the value range of -1 to 1.

The angle (operation result) at d is stored in radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RAD and DEG instructions.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

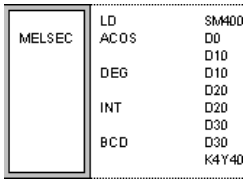
- The value in s and s+1 exceeds the value range of -1 to 1. (Error code 4100)
- The contents of the specified device or the operation result is not zero and not within the range from $\pm 2^{-126}$ to $\pm 2^{128}$.
  (For the Universal model QCPU, LCPU)
  (Error code 4100)
- The value of the specified device is –0.
  (For the Basic Model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (operation results in an overflow)
  (For the Universal model QCPU, LCPU)
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

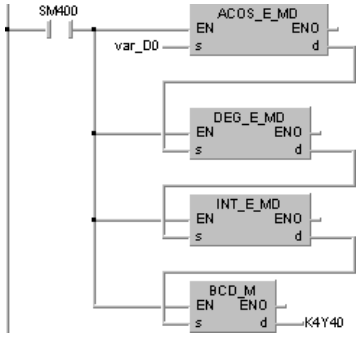
ACOS

The following program calculates the arcus cosine value from the 32-bit floating-point data (real number) in D0 and D1. The resulting angle in radian measure is output at Y40 through Y4F as 4-digit BCD value.

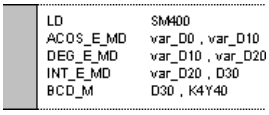
MELSEC Instruction List

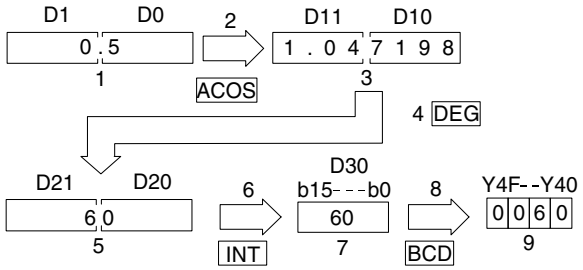


Ladder Diagram



IEC Instruction List





[1] 32-bit floating point value (real number)
[2] Arcus cosine calculation
[3] 32-bit floating point value (real number)
[4] Conversion of the angle measures
[5] 32-bit floating point value (real number)
[6] Conversion into the BIN format
[7] Binary value
[8] Conversion into the BCD format
[9] BCD value

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.10 ACOSD, ACOSDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

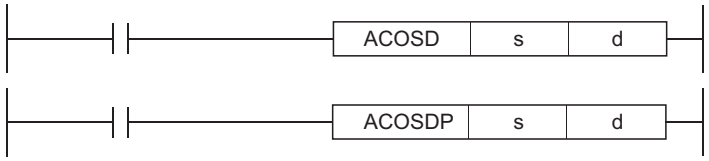
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

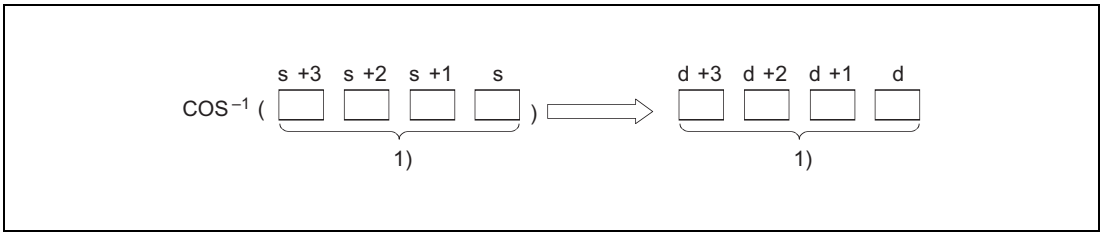
Set Data	Meaning	Data Type
s	COS value for the ACOSD (arcus cosine) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions

Arcus cosine calculation of floating-point values (Double precision)

ACOSD Arcus cosine calculation

The ACOSD instruction calculates the angle from the cosine value specified by s. The result is stored in the devices specified by d.



[1] 64-bit floating-point value (real number)

The cosine value in s may range within the value range of -1 to 1.

The angle (operation result) at d is stored in radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RADD and DEGD instructions.

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The value specified by s is within the double-precision floating-point range and outside the range of -1.0 to 1.0.
  (Error code 4100)
- The result exceeds the following range (Operation results in an overflow):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

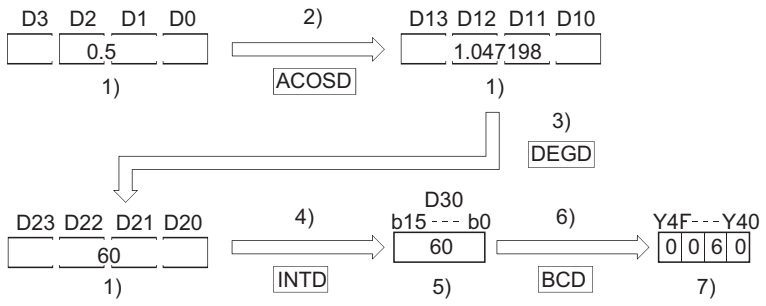
ACOSD

The following program calculates the arcus cosine value from the 64-bit floating-point data (real number) in D0 to D3. The resulting angle in radian measure is output at Y40 through Y4F as 4-digit BCD value.

MELSEC Instruction List

IEC Instruction List

Ladder Diagram



[1] 64-bit floating-point value (real number)
[2] Arcus cosine calculation
[3] Conversion of the angle measures
[4] Conversion into the BIN format
[5] Binary value
[6] Conversion into the BCD format
[7] BCD value

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.11 ATAN, ATANP

CPU

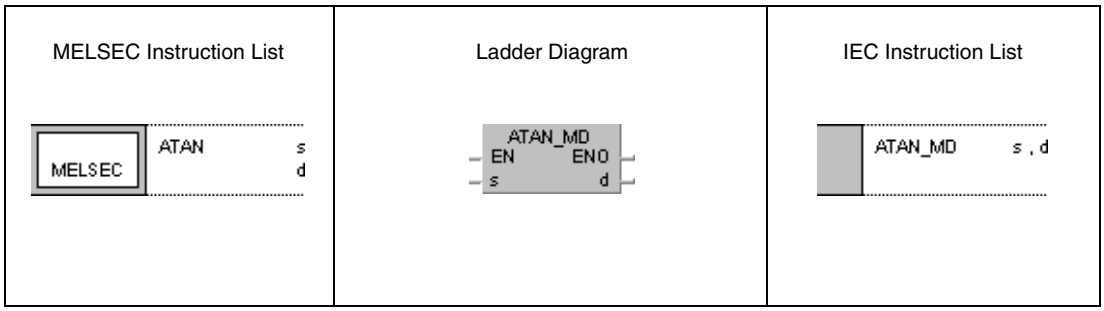
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

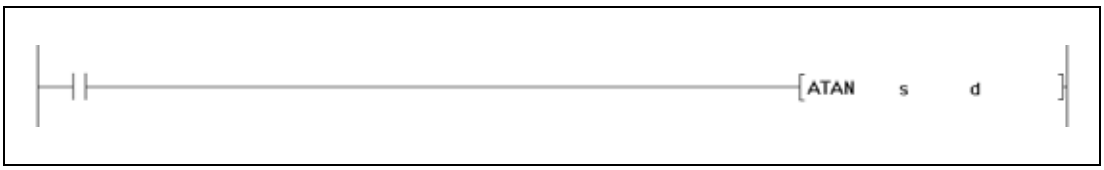
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Applicable for the Universal model QCPU, LCPU

GX IEC Developer



GX Works2



Variables

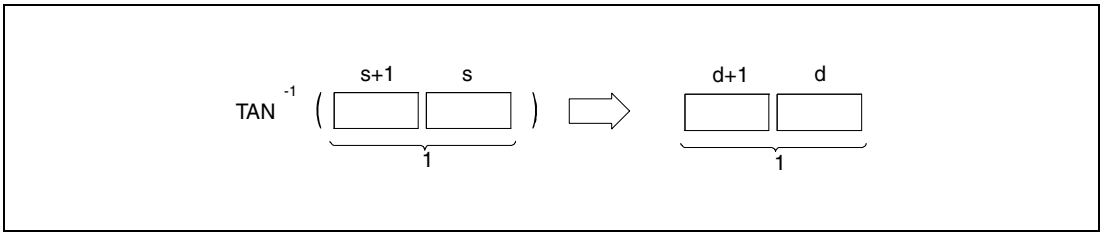
Set Data	Meaning	Data Type
s	TAN value for the ATAN (arcus tangent) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions

Arcus tangent calculation of floating-point values (Single precision)

ATAN Arcus tangent calculation

The ATAN instruction calculates the angle from the tangent value in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

The angle (operation result) at d is stored in radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RAD and DEG instructions.

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The contents of the specified device or the operation result is not zero and not within the range from $\pm 2^{-126}$ to $\pm 2^{128}$.
  (For the Universal model QCPU, LCPU)
  (Error code 4100)
- The value of the specified device is –0.
  (For the High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (Operation results in an overflow)
  (For the Universal model QCPU, LCPU)
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

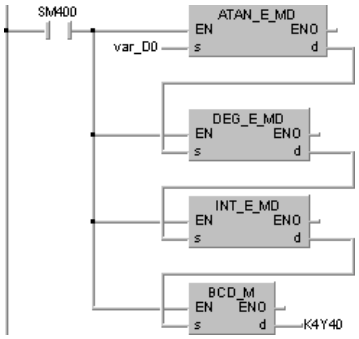
ATAN

The following program calculates the arcus tangent value from the 32-bit floating-point data (real number) in D0 and D1. The resulting angle in radian measure is output at Y40 through Y4F as 4-digit BCD value.

MELSEC Instruction List

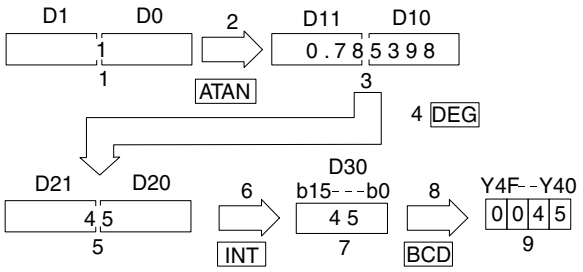
```
MELSEC   LD     SM400
         ATAN   D0
                D10
         DEG    D10
                D20
         INT    D20
                D30
         BCD    D30
                K4Y40
```

Ladder Diagram



IEC Instruction List

```
LD          SM400
ATAN_E_MD   var_D0 , var_D10
DEG_E_MD    var_D10 , var_D20
INT_E_MD    var_D20 , D30
BCD_M       D30 , K4Y40
```



[1] 32-bit floating point value (real number)
[2] Arcus tangent calculation
[3] 32-bit floating point value (real number)
[4] Conversion of the angle measures
[5] 32-bit floating point value (real number)
[6] Conversion into the BIN format
[7] Binary value
[8] Conversion into the BCD format
[9] BCD value

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.12 ATAND, ATANDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

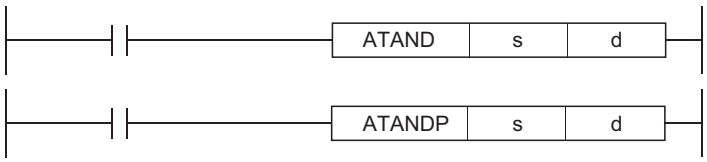
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

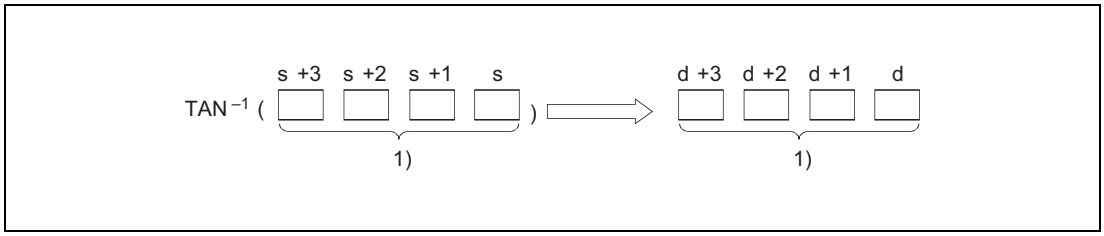
Set Data	Meaning	Data Type
s	TAN value for the ATAND (arcus tangent) instruction or first number of the devices storing such data	Real number
d	First number of device storing the operation result	

Functions

Arcus tangent calculation of floating-point values (Double precision)

ATAND Arcus tangent calculation

The ATAND instruction calculates the angle from the tangent value specified by s. The result is stored in the device specified by d.



[1] 64-bit floating-point value (real number)

The angle (operation result) at d is stored radian measure (degrees x π/180). The conversion from degrees into radian is described in the sections on the RADD and DEGD instructions.

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

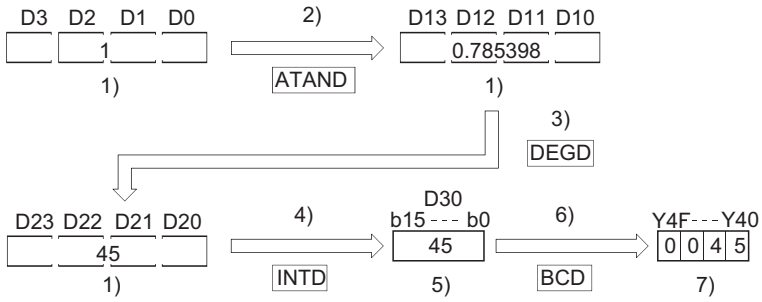
ATAND

The following program calculates the arcus tangent value from the 64-bit floating-point data (real number) in D0 to D3. The resulting angle in radian measure is output at Y40 through Y4F as 4-digit BCD value.

MELSEC Instruction List

IEC Instruction List

Ladder Diagram



[1] 64-bit floating-point value (real number)
[2] Arcus tangent calculation
[3] Conversion of the angle measures
[4] Conversion into the BIN format
[5] Binary value
[6] Conversion into the BCD format
[7] BCD value

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.13 RAD, RADP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

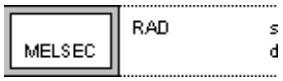
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

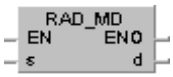
[1] Applicable for the Universal model QCPU, LCPU

GX IEC Developer

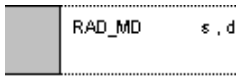
MELSEC Instruction List



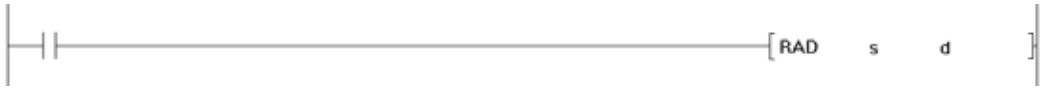
Ladder Diagram



IEC Instruction List



GX Works2



Variables

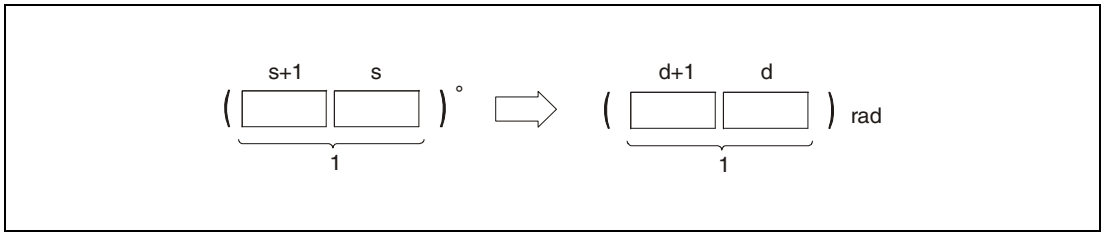
Set Data	Meaning	Data Type
s	Angle to be converted to radian units or first number of the devices storing such data	Real number
d	First number of device storing conversion result	

Functions

Conversion from degrees into radian as floating-point value (Single precision)

RAD Conversion from degrees into radian

The RAD instruction calculates the radian value (rad) from the degree value (°) in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

The conversion from degrees into radiant applies to the following equation:

Radian value = degree value x π / 180

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The contents of the specified device or the operation result is not zero and not within the range from $\pm 2^{-126}$ to $\pm 2^{128}$.
  (For the Universal model QCPU, LCPU)
  (Error code 4100)
- The value of the specified device is –0.
  (For the Basic Model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (operation results in an overflow):
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (For the Universal model QCPU, LCPU)
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

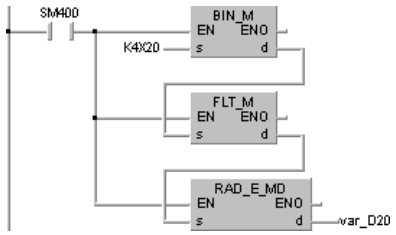
RAD

The following program calculates the radian value from the degree value of the 4-digit BCD value in X20 through X2F. The result is stored in D20 and D21 as 32-bit floating-point value.

MELSEC Instruction List

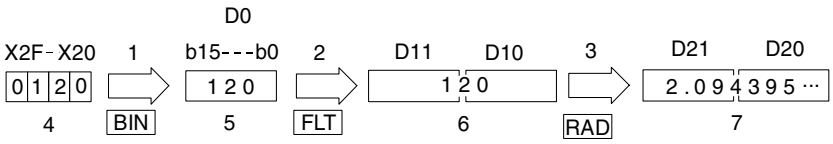
```
MELSEC   LD    SM400
         BIN   K4X20
               D0
         FLT   D0
               D10
         RAD   D10
               D20
```

Ladder Diagram



IEC Instruction List

```
LD       SM400
BIN_M    K4X20 , D0
FLT_M    D0 , var_D10
RAD_E_MD var_D10 , var_D20
```



[1] Conversion into the BIN format

[2] Conversion into the floating-point format

[3] Conversion into radian measure

[4] BCD value

[5] Binary value

[6] 32-bit floating point value (real number)

[7] 32-bit floating point value (real number)

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.14 RADD, RADDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

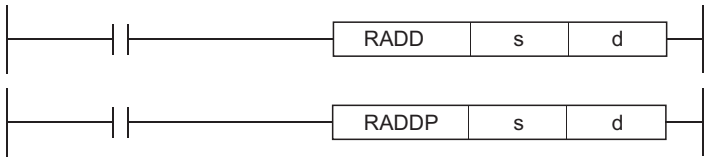
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



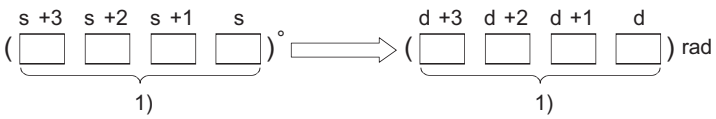
Variables

Set Data	Meaning	Data Type
s	Angle to be converted to radian units or first number of the devices storing such data	Real number
d	First number of device storing conversion result	

Functions **Conversion from degrees into radian as floating-point value (Double precision)**

RADD Conversion from degrees into radian

The RADD instruction calculates the radian value (rad) from the degree value (°) specified by s. The result is stored in the device specified by d.



[1] 64-bit floating-point value (real number)

The conversion from degrees into radiant applies to the following equation:

Radian value = degree value x π / 180

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

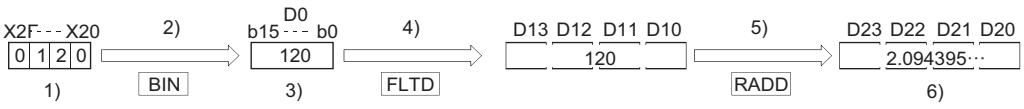
- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (Operation results in an overflow):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

RADD

The following program calculates the radian value from the degree value of the 4-digit BCD value in X20 through X2F. The result is stored in D20 to D23 as 64-bit floating-point value.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



[1] BCD value
[2] Conversion into the BIN format
[3] Binary value
[4] Conversion into the floating-point format
[5] Conversion into radian measure
[6] 64-bit floating-point value (real number)

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.15 DEG, DEGP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

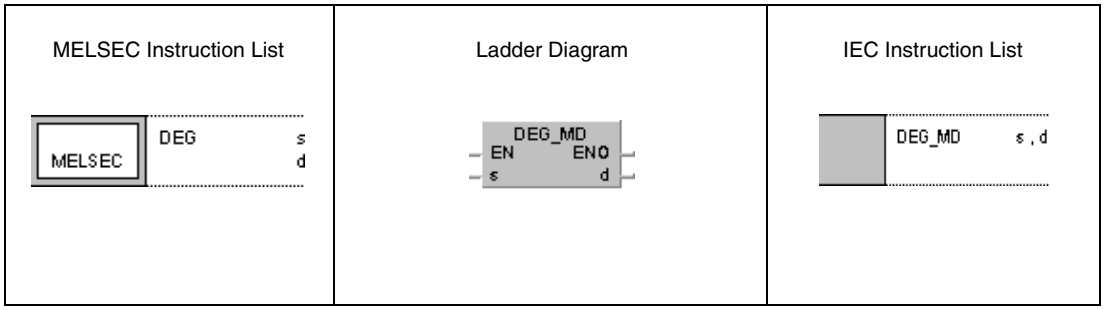
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

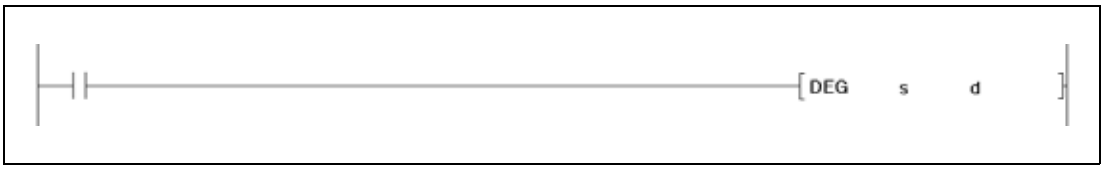
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Applicable for the Universal model QCPU, LCPU

GX IEC Developer



GX Works2



Variables

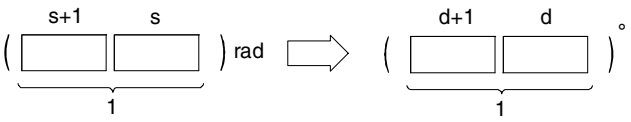
Set Data	Meaning	Data Type
s	Radian angle to be converted to degrees or first number of the devices storing such data	Real number
d	First number of device storing conversion result	

Functions

Conversion from radian in floating-point format into degrees (Single precision)

DEG Conversion from radian into degrees

The DEG instruction calculates the degree value (°) from the radian value (rad) in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

The conversion from radian into degrees applies to the following equation:

Degree value = *radian value x 180 / π*

Operation Error

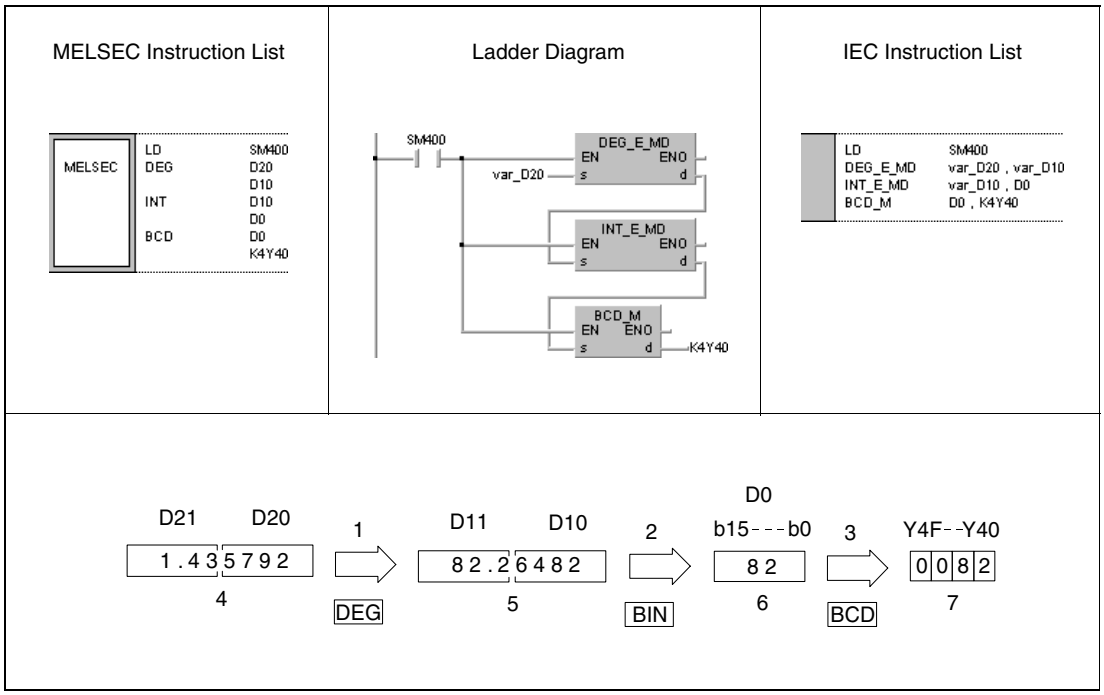
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value of the specified device is –0.
  (For the Basic Model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (overflow occurs):
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (For the Universal model QCPU, LCPU)
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

DEG

The following program calculates the degree value from the radian value stored in D20 and D21 as 32-bit floating-point value. The result is stored in Y40 to Y4F as BCD value.



[1] Conversion into degrees
[2] Conversion into the BIN format
[3] Conversion into the BCD format
[4] 32-bit floating point value (real number)
[5] 32-bit floating point value (real number)
[6] Binary value
[7] BCD value

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.16 DEGD, DEGDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

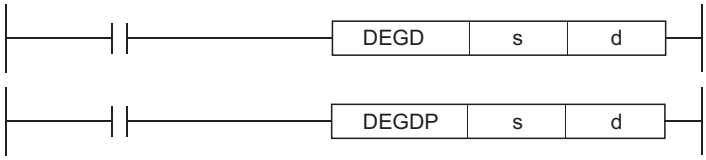
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



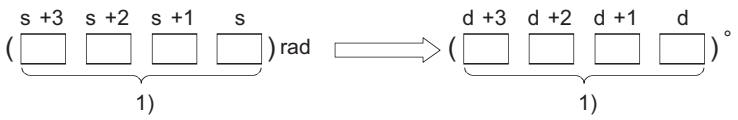
Variables

Set Data	Meaning	Data Type
s	Radian angle to be converted to degrees or first number of the devices storing such data	Real number
d	First number of device storing conversion result	

Functions **Conversion from radian in floating-point format into degrees (Double precision)**

DEGD Conversion from radian into degrees

The DEGD instruction calculates the degree value (°) from the radian value (rad) specified by s. The result is stored in the device specified by d.



[1] 64-bit floating-point value (real number)

The conversion from radian into degrees applies to the following equation:

Degree value = *radian value x 180 / π*

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

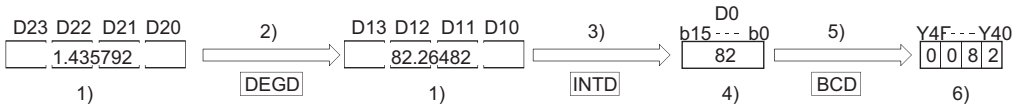
- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

DEGD

The following program calculates the degree value from the radian value stored in D20 to D23 as 64-bit floating point value. The result is stored in Y40 to Y4F as BCD value.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



[1] 64-bit floating-point value (real number)
[2] Conversion to angle
[3] Conversion into the BIN format
[4] Binary value
[5] Conversion into the BCD format
[6] BCD value

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.17 POW, POWP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

Devices

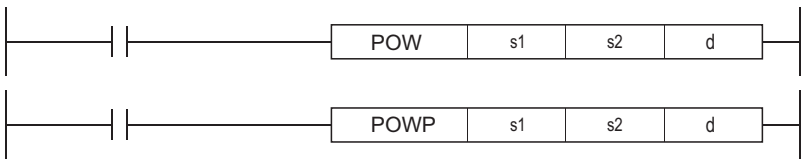
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	●	●	●	●[1)]	—
s2	—	●	●	—	●	●	●	●[1)]	—
d	—	●	●	—	●	●	●	—	—

[1] Available only for real number

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

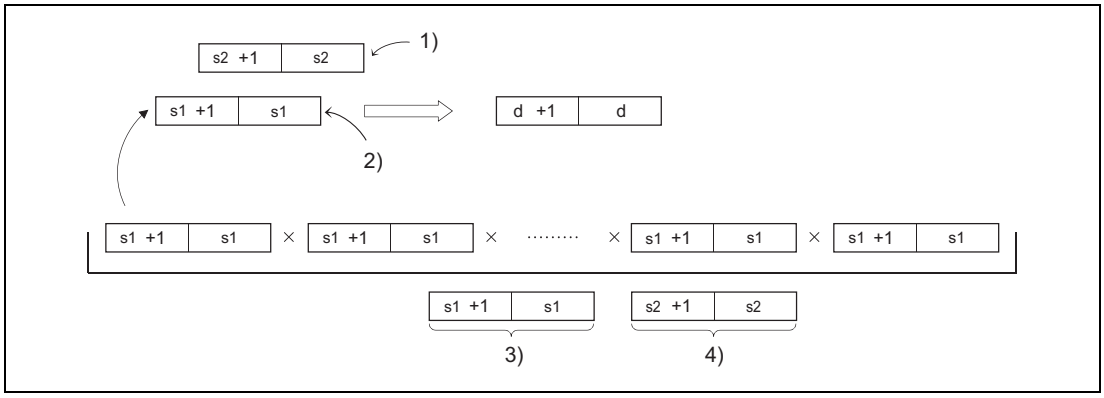
Set Data	Meaning	Data Type
s1	Exponentiation recipient data or first number of the devices storing such data	Real number
s2	Exponentiation data or first number of the devices storing such data	
d	First number of device storing operation result	

Functions

Exponentiation operation on floating-point data (Single precision)

POW Exponentiation operation

The POW instruction raises the 32-bit floating-point data type real number specified by s1 to the number nth power specified by s2, and then stores the operation result into the device specified by d.



[1] Exponentiation data

[2] Exponentiation recipient data

[3] 32-bit floating-point value (real number)

[4] 32-bit floating-point value (real number)

The instruction raises 3) to the power of 4).

The following shows the values to be specified by and stored into s1 or s2:
$0, \pm 2^{-126} \leq$ (Set values (Storage values)) $< \pm 2^{128}$.

If the value resulted from the operation is –0 or an underflow occurs, the result will be processed as 0.

Operation Error

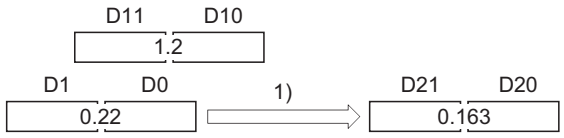
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The values specified by s1 or s2 are not zero and not within the following range:
  $\pm 2^{-126} \leq$ (Specified value (storage values)) $< \pm 2^{128}$
  (Error code 4140)
- The value of s1 or s2 is –0.
  (Error code 4140)
- The result exceeds the following range:
  $-2^{126} \leq$ (Operation result) $\leq 2^{126}$
  (Error code 4141)

Program Example

POW

The following program raises the 32-bit floating-point data type real number data specified by D0 and D1 to the data specified by (D10 and D11)th power, when X10 is turned on. The operation result is stored into D20 and D21.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

[1] Exponentiation operation

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.18 POWD, POWDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

Devices

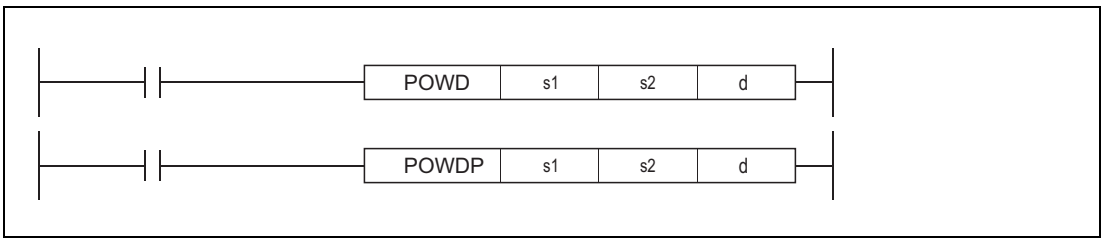
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	●	●	—	●[1)]	—
s2	—	●	●	—	●	●	—	●[1)]	—
d	—	●	●	—	●	●	—	—	—

[1] Available only for real number

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



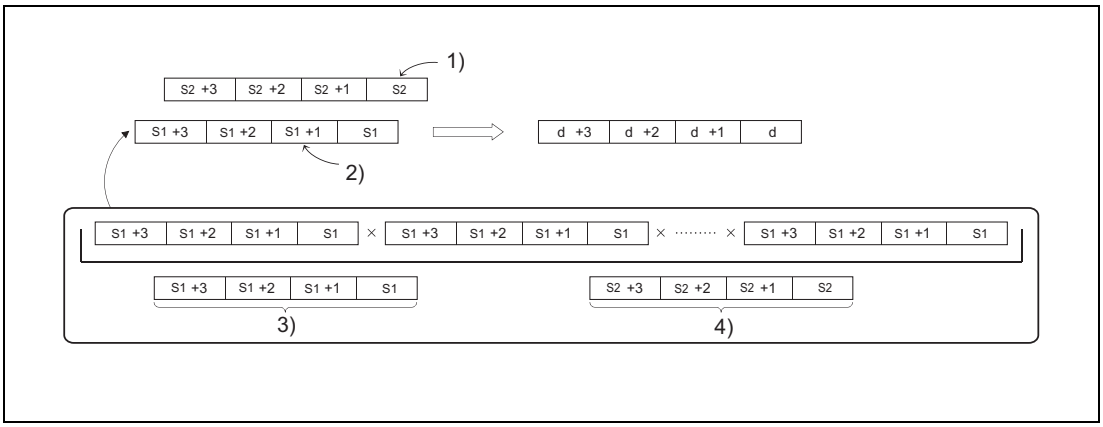
Variables

Set Data	Meaning	Data Type
s1	Exponentiation recipient data or first number of the devices storing such data	Real number
s2	Exponentiation data or first number of the devices storing such data	
d	First number of device storing operation result	

Functions **Exponentiation operation on floating-point data (Double precision)**

POWD Exponentiation operation

The POW instruction raises the 64-bit floating-point data type real number specified by s1 to the number nth power specified by s2, and then stores the operation result into the device specified by d.



[1] Exponentiation data

[2] Exponentiation recipient data

[3] 64-bit floating-point value (real number)

[4] 64-bit floating-point value (real number)

The instruction raises the real number 3) to the power of 4).

The following shows the values to be specified by and stored into s1 or s2:
0, $\pm 2^{-1022} \leq$ (Set values (storage values)) $< \pm 2^{1024}$.

If the value resulted from the operation is –0 or an underflow occurs, the result will be processed as 0.

Operation Error

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

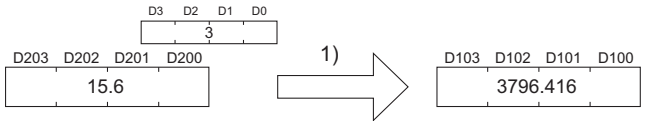
- The values specified by s1 or s2 are not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Specified value (storage values)) $< \pm 2^{1024}$
  (Error code 4140)
- The value of s1 or s2 is –0.
  (Error code 4140)
- The result exceeds the following range:
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

POWD

The following program raises the 64-bit floating-point data type real number data specified by D200 to D203 to the number nth specified by (D0 to D3) power, when X10 is turned on. The operation result is stored into D100 to D103.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



[5] Exponentiation operation

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.19 SQR, SQRP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

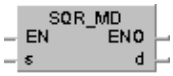
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Applicable for the Universal model QCPU, LCPU

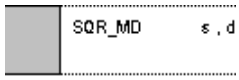
GX IEC Developer

MELSEC Instruction List

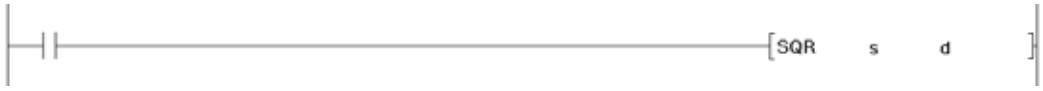
Ladder Diagram



IEC Instruction List



GX Works2



Variables

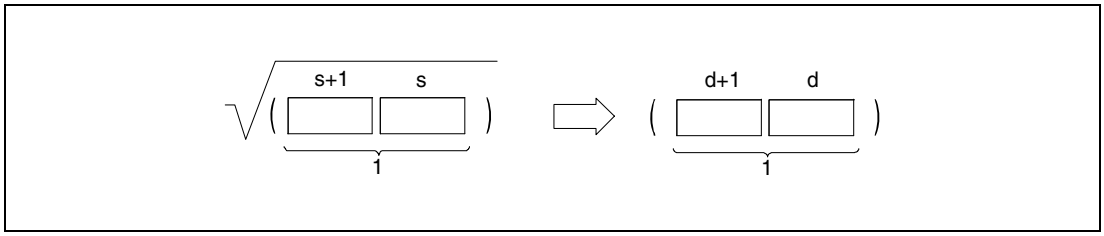
Set Data	Meaning	Data Type
s	Data of which the square root is obtained or first number of the devices storing such data	Real number
d	First number of device storing the calculation result	

Functions

Square root calculation of floating-point values (Single precision)

SQR Square root calculation

The SQR instruction calculates the square root of the 32-bit floating-point value in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

Only positive values may be stored in s and s+1.
(Negative values cannot be processed).

Operation Errors

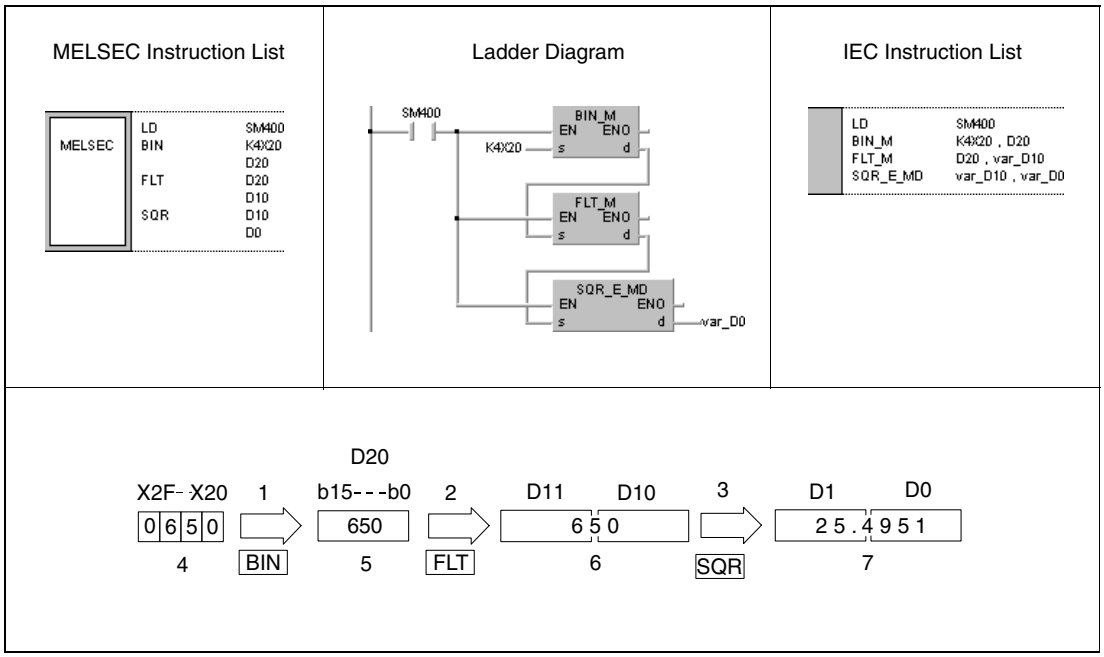
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value entered in s is negative. (Error code 4100)
- The contents of the specified device or the operation result is not zero and not within the range from $\pm 2^{-126}$ to $\pm 2^{128}$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)
- The value of the specified device is –0.
  (For the Basic Model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (Operation results in an overflow):
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (For the Universal model QCPU, LCPU)
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)
-

Program Example

SQR

The following program calculates the square root of the 4-digit BCD value in X20 through X2F. The result is stored in D0 and D1.



[1] Conversion into the BIN format

[2] Conversion into the floating-point format

[3] Square root calculation

[4] BCD value

[5] Binary value

[6] 32-bit floating point value (real number)

[7] 32-bit floating point value (real number)

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.20 SQRD, SQRDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

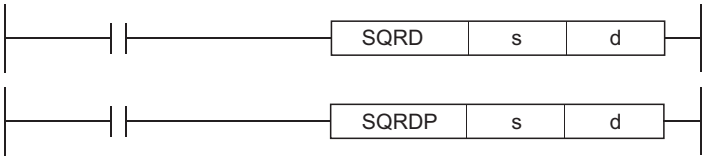
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



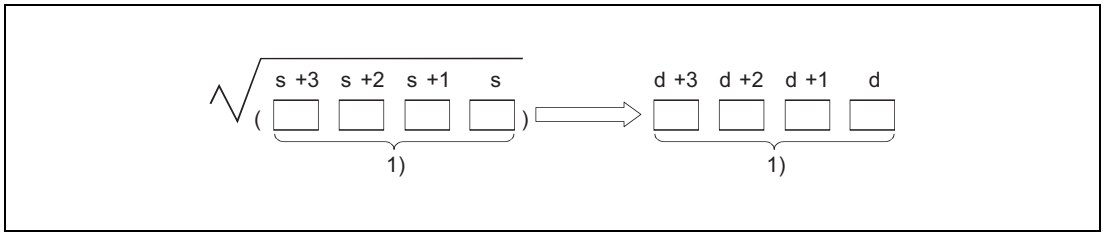
Variables

Set Data	Meaning	Data Type
s	Data of which the square root is obtained or first number of the devices storing such data	Real number
d	First number of device storing the calculation result	

Functions **Square root calculation of floating-point values (Double precision)**

SQRD Square root calculation

The SQRD instruction calculates the square root of the 64-bit floating-point value specified by s. The result is stored in the device specified in d.



[1] 64-bit floating-point value (real number)

Only positive values may be stored in s.
(Negative values cannot be processed).

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

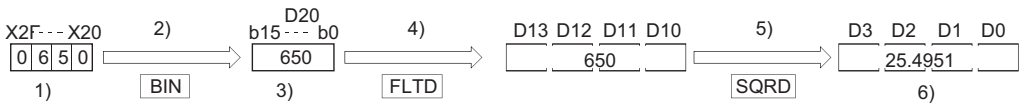
- The value entered in s is negative. (Error code 4100)
- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

SQRD

While X0 is set, the following program calculates the square root of the 4-digit BCD value in X20 through X2F. The result is stored in D0 to D3.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



[1] BCD value
[2] Conversion into the BIN format
[3] Binary value
[4] Conversion into the floating-point format
[5] Square root calculation
[6] 64-bit floating-point value (real number)

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.21 EXP, EXPP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

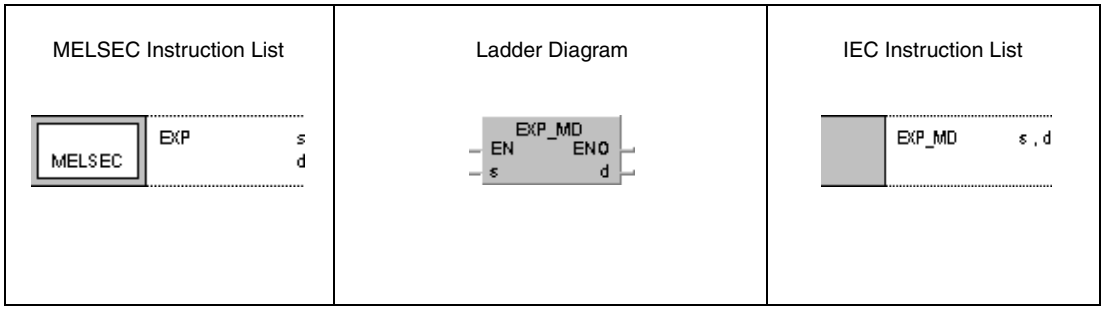
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

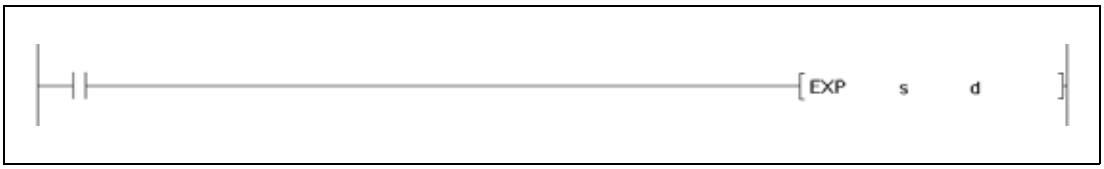
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

[1] Applicable for the Universal model QCPU, LCPU

GX IEC Developer



GX Works2



Variables

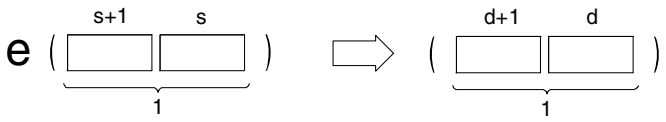
Set Data	Meaning	Data Type
s	Value for the EXP instruction or first number of device storing such data	Real number
d	First number of device storing the operation result	

Functions

Floating point values as exponent of the base e (Single precision)

EXP Exponent of e

The EXP instruction calculates the corresponding exponent to the base e from the 32-bit floating-point value in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

The calculation is based on the Euler´s constant: "e = 2.718281828".

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The operation result is outside the range shown below:
  $\pm 2^{-126} \leq$ (Operation result) $\leq \pm 2^{128}$ for the High Performance model QCPU and
  $\pm 2^{-126} \leq$ (Operation result) $< \pm 2^{128}$ for the Basic Model QCPU, Process CPU, Redundant CPU
  (Error code 4100)
- The value of the specified device is –0.
  (For the Basic Model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if –0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (overflow occurs):
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (For the Universal model QCPU, LCPU)
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

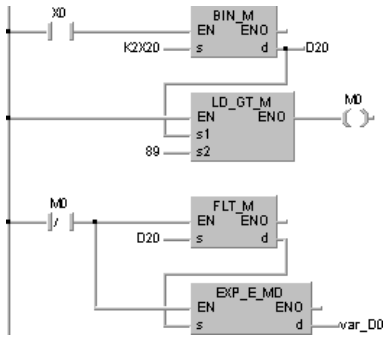
EXP

The following program calculates the result of the exponential function to the base e with the 2-digit BCD value at X20 through X27. The result is stored in D0 and D1 in 32-bit floating-point format.

MELSEC Instruction List

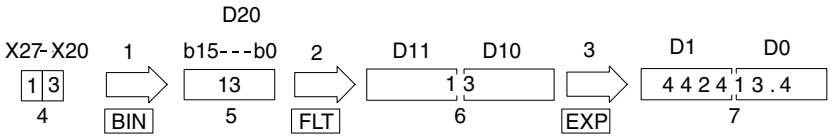
```
MELSEC
LD    X0
BIN   K2X20
      D20
LD>   D20
      K89
OUT   M0
LDI   M0
FLT   D20
      D10
EXP   D10
      D0
```

Ladder Diagram



IEC Instruction List

```
LD      X0
BIN_M   K2X20 , D20
LD_GT_M D20 , 89
ST      M0
LDN     M0
FLT_M   D20 , var_D10
EXP_E_MD var_D10 , var_D0
```



[1] Conversion into the BIN format

[2] Conversion into the floating-point format

[3] Exponential calculation

[4] BCD value

[5] Binary value

[6] 32-bit floating point value (real number)

[7] 32-bit floating point value (real number)

NOTES

The operation result will be under 2^{129} if the BCD value of X20 to X27 is less than 89, from the calculation ln 2^{129} = 89.4.
Because setting a value of over 89 will return an operation error, M0 is turned ON in this example if a value of over 89 has been set to avoid the error.

Conversion from natural logarithm to common logarithm:
In the CPU module, calculation is made using a natural logarithm.

To obtain a common logarithm value, enter in s a common logarithm value divided by 0.43429: $10^x = e^{x/0.43429}$.

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.22 EXPD, EXPDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

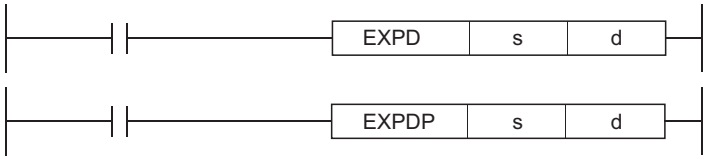
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



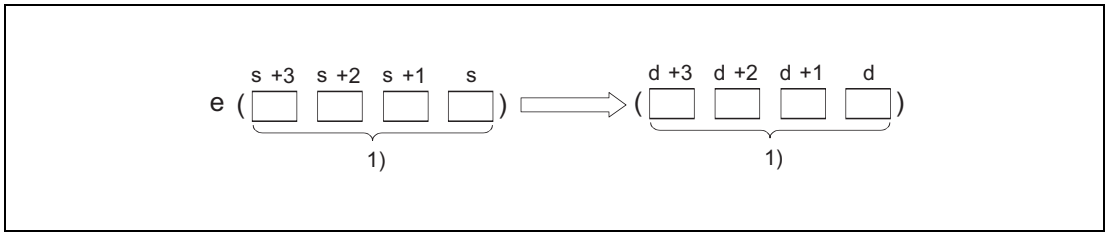
Variables

Set Data	Meaning	Data Type
s	Value for the EXPD instruction or first number of device storing such data	Real number
d	First number of device storing the operation result	

Functions **Floating point values as exponent of the base e (Double precision)**

EXPD Exponent of e

The EXPD instruction calculates the corresponding exponent to the base e from the 64-bit floating-point value specified by s. The result is stored in the device specified by d.



[1] 64-bit floating-point value (real number)

The calculation is based on the Euler´s constant: "e = 2.718281828".

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

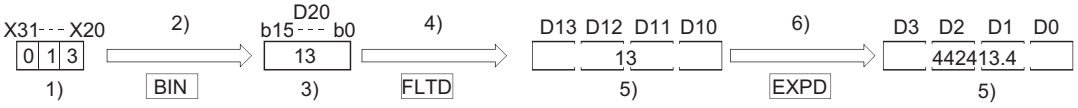
- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

EXPD

The following program calculates the result of the exponential function to the base e with the 2-digit BCD value at X20 through X31. The result is stored in D0 to D3 in 64-bit floating-point format.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



[1] BCD value
[2] Conversion into the BIN format
[3] Binary value
[4] Conversion into the floating-point format
[5] 64-bit floating-point value (real number)
[6] Exponential calculation

NOTES

The operation result will be under 2^{1024} if the BCD value of X20 to X31 is less than 709, from the calculation ln 2^{1024} = 709.7832.
Because setting a value of over 709 will return an operation error, M0 is turned ON in this example if a value of over 709 has been set to avoid the error.

Conversion from natural logarithm to common logarithm:
In the CPU module, calculation is made using a natural logarithm.

To obtain a common logarithm value, enter in s a common logarithm value divided by 0.43429: $10^x = e^{x/0.43429}$.

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.23 LOG, LOGP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	●[1)]	●	—
d	—	●	●	—	●	●	●[1)]	—	—

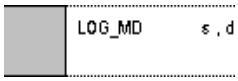
[1] Applicable for the Universal model QCPU, LCPU

GX IEC Developer

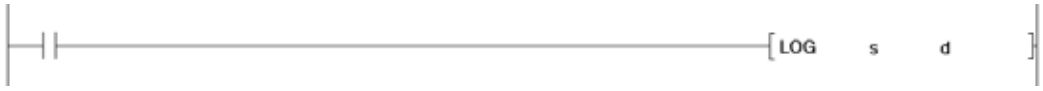
MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2



Variables

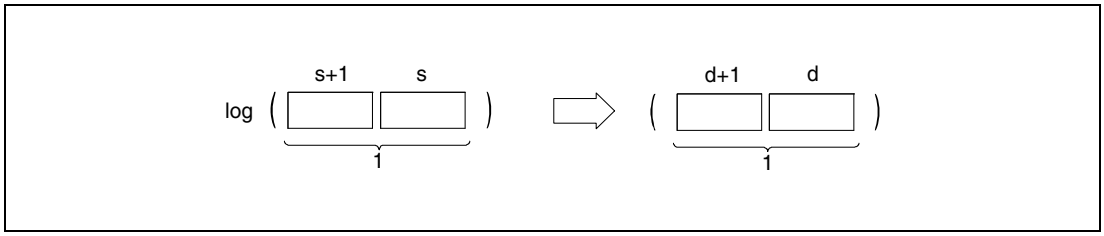
Set Data	Meaning	Data Type
s	Value for the LOG instruction or first number of device storing such data	Real number
d	First number of device storing the operation result	

Functions

Logarithm (ln) calculation from floating-point values (Single precision)

LOG Logarithm (ln) calculation

The LOG instruction calculates the natural logarithm from the 32-bit floating-point number in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

Only positive values can be specified in s and s+1. Negative values cannot be calculated.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value specified in s is negative. (Error code 4100)
- The value specified in s is 0. (Error code 4100)
- The contents of the specified device or the operation result are not zero and not within the following range:
  $\pm 2^{-126}$ <= (Contents of device or operation result) < $\pm 2^{128}$
  (For the Universal model QCPU, LCPU)
  (Error code 4100)
- The value of the specified device is –0.
  (For the Basic Model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
  (Error code 4100)
  There are CPU modules that will not result in an operation error if -0 is specified.
  For details, refer to section 3.5.1.
- The result exceeds the following range (operation results in an overflow):
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (For the Universal model QCPU, LCPU)
  (Error code 4141)
- The value of the specified device is –0, unnormalized number, nonnumeric, or $\pm\infty$.
  (For the Universal model QCPU, LCPU)
  (Error code 4140)

Program Example

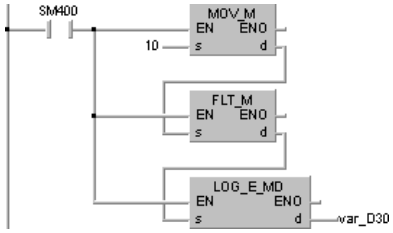
LOG

The following program calculates the natural logarithm from the value 10. The result is stored in D30 and D31.

MELSEC Instruction List

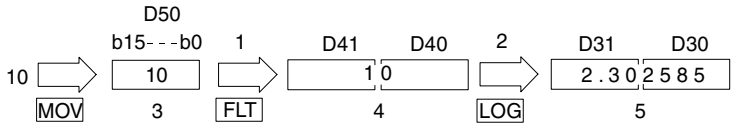
```
MELSEC   LD    SM400
         MOV   K10
               D50
         FLT   D50
               D40
         LOG   D40
               D30
```

Ladder Diagram



IEC Instruction List

```
LD       SM400
MOV_M    10 , D50
FLT_M    D50 , var_D40
LOG_E_MD var_D40 , var_D30
```



[1] Conversion into the floating-point format

[2] Logarithm calculation

[3] Binary value

[4] 32-bit floating point value (real number)

[5] 32-bit floating point value (real number)

NOTES

The LOG instruction calculates the natural logarithm (base e). The following formula converts the natural logarithm to normal logarithm (base 10):

$log_{10} X = 0.43429 \times log_e X$

Universal model QCPU and LCPU can also calculate the normal logarithm (base 10) (refer to section 7.12.25 "LOG10, LOG10P" and section 7.12.26 "LOG10D, LOG10DP").

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.24 LOGD, LOGDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

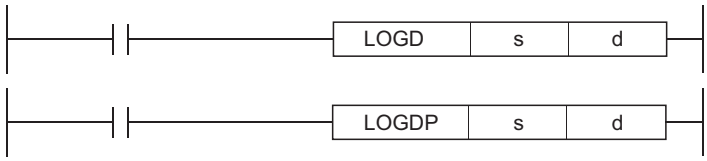
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

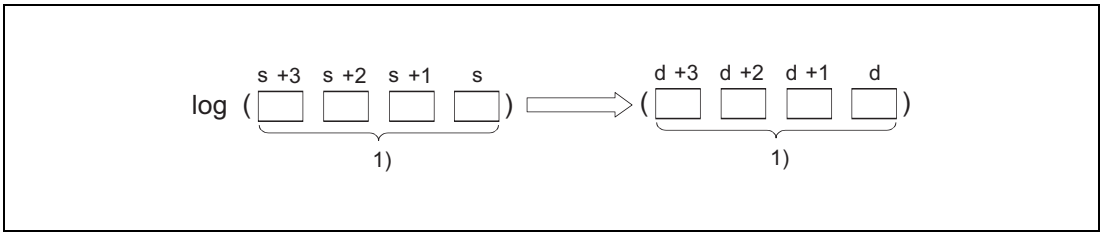
Set Data	Meaning	Data Type
s	Value for the LOGD instruction or first number of device storing such data	Real number
d	First number of device storing the operation result	

Functions

Logarithm (ln) calculation from floating-point values (Double precision)

LOGD Logarithm (ln) calculation

The LOGD instruction calculates the natural logarithm from the 64-bit floating-point number specified by s taking (e) as a base. The result is stored in the device specified by d.



[1] 64-bit floating-point value (real number)

Only positive values can be specified by s. Negative values cannot be calculated.

When the operation results in -0 or an underflow, the result is processed as 0.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

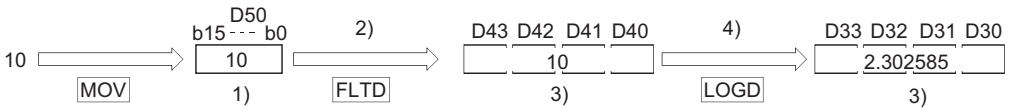
- The value specified in s is negative. (Error code 4100)
- The value specified in s is 0. (Error code 4100)
- The value of the specified device is not zero ro not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

LOGD

The following program calculates the natural logarithm from the value 10 set by D50. The result is stored in D30 to D33.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



[1] Binary value

[2] Conversion into the floating-point format

[3] 64-bit floating-point value (real number)

[4] Logarithm calculation

NOTES

The LOGD instruction calculates the natural logarithm (base e). The following formula converts the natural logarithm to normal logarithm (base 10):

$\log_{10} X = 0.43429 \times \log_e X$

Universal model QCPU and LCPU can also calculate the normal logarithm (base 10) (refer to section 7.12.25 "LOG10, LOG10P" and section 7.12.26 "LOG10D, LOG10DP").

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.25 LOG10, LOG10P

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

Devices

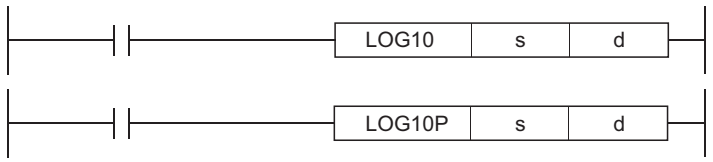
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant E	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	●	●	—	●[1)]	—
d	—	●	●	—	●	●	—	—	—

[1] Available only for real number

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

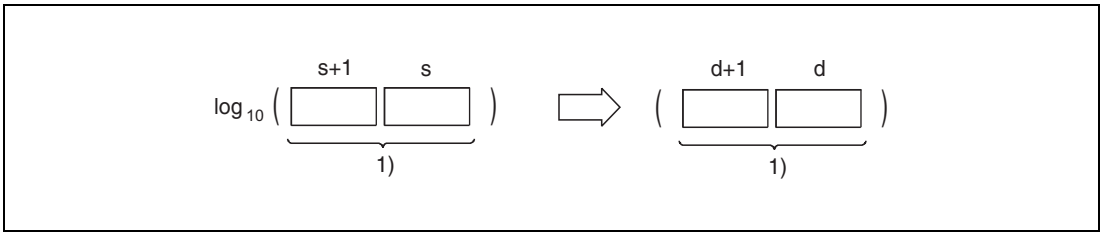
Set Data	Meaning	Data Type
s	Value for the LOG10 instruction or first number of device storing such data	Real number
d	First number of device storing the operation result	

Functions

Common logarithm calculation from floating-point values (Single precision)

LOG10 Common logarithm calculation

The LOG10 instruction calculates the common logarithm (base 10) from the 32-bit floating-point number in s and s+1. The result is stored in d and d+1.



[1] 32-bit floating point value (real number)

Only positive values can be specified in s and s+1. Negative values cannot be calculated.

If the value resulted from the operation is –0 or an underflow occurs, the result will be processed as 0.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

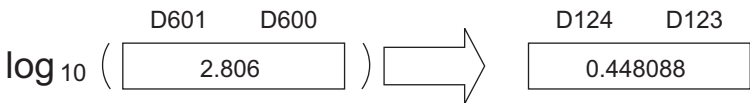
- The value specified in s is negative. (Error code 4100)
- The value specified in s is 0. (Error code 4100)
- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-126} \leq$ (Value of specified device) $< \pm 2^{128}$
  (Error code 4140)
- The value specified in s is –0.
  (Error code 4140)
- The result exceeds the following range (overflow occurs):
  $-2^{128} \leq$ (Operation result) $\leq 2^{128}$
  (Error code 4141)

Program Example

LOG10

The following program obtains the value for common logarithm of the 32-bit floating-point data type real number specified by D600 and D601, when M0 is turned on. The result is stored into D123 and D124.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.26 LOG10D, LOG10DP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

Devices

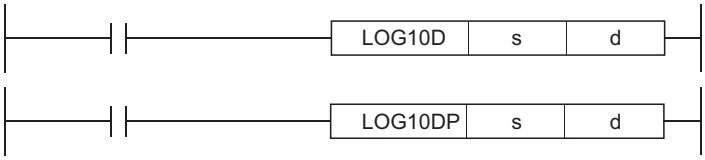
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●[1)]	—
d	—	●	●	—	—	—	—	—	—

[1] Available only for real number

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



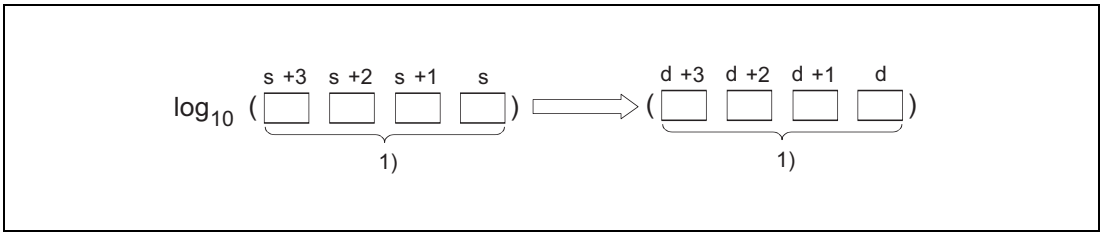
Variables

Set Data	Meaning	Data Type
s	Value for the LOG10D instruction or first number of device storing such data	Real number
d	First number of device storing the operation result	

Functions **Common logarithm calculation from floating-point values (Double precision)**

LOG10D Common logarithm calculation

The LOG10D instruction calculates the common logarithm (base 10) from the 64-bit floating-point number specified by s. The result is stored in the device specified by d.



[1] 64-bit floating-point value (real number)

Only positive values can be specified in s. Negative values cannot be calculated.

If the value resulted from the operation is –0 or an underflow occurs, the result will be processed as 0.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

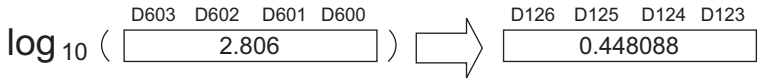
- The value specified in s is negative. (Error code 4100)
- The value specified in s is 0. (Error code 4100)
- The value of the specified device is not zero and not within the following range:
  $\pm 2^{-1022} \leq$ (Value of specified device) $< \pm 2^{1024}$
  (Error code 4140)
- The value of the specified device is –0.
  (Error code 4140)
- The result exceeds the following range (overflow occurs):
  $-2^{1024} \leq$ (Operation result) $\leq 2^{1024}$
  (Error code 4141)

Program Example

LOG10D

The following program obtains the value for common logarithm of the 64-bit floating-point data type real number specified by D600 to D603, when M0 is turned on. The result is stored into D123 to D126.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List



NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.27 RND, RNDP, SRND, SRNDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

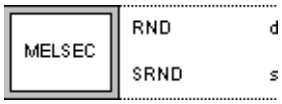
[1] Basic model QCPU: The upper five digits of the serial No. are "04122" or higher.

Devices

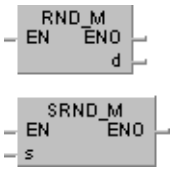
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H, 16#	Other
	Bit	Word		Bit	Word				
	for RND, RNDP								
d	●							—	—
	for SRND, SRNDP								
s	●							●	—

GX IEC Developer

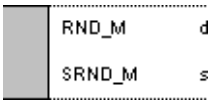
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type
d	First number of device storing the randomized value	BIN 16-bit
s	Random value series or first number of device storing such data	

Functions **Randomizing values and series update**

RND Randomizing values

The RND instruction generates a random value ranging from 0 to 32767 and stores it in d.

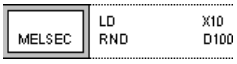
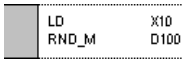
SRND Updating series of random values

The SRND instruction updates the series of random values stored in s.

Program Example 1

RND

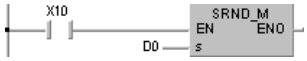
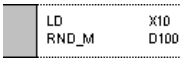
While X10 is set, the following program stores the generated random value in D100.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

Program Example 2

SRND

While X10 is set, the following program updates the series of random values in D0.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

7.12.28 BSQR, BSQRP, BDSQR, BDSQRP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

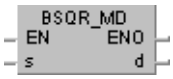
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H, (16#)	Other
	Bit	Word		Bit	Word				
s	●							●	—
d	●							—	—

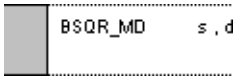
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

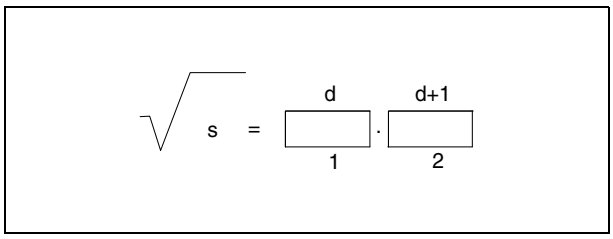
Set Data	Meaning	Data Type
s	Data for the square root calculation or number of device storing such data	BCD 4-/ 8-digit
d	First number of device storing the calculation result	BCD 4-digit

Functions

Square root calculation from 4-digit or 8-digit BCD data

BSQR Square root calculation from 4-digit BCD data

The BSQR instruction calculates the square root of s and stores the result in d and d+1.



[1] Integer part
[2] Decimal places

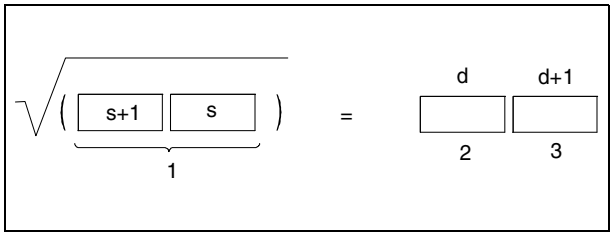
The data in s must be a BCD value with at maximum 4 digits. The value range from 0 to 9999 must not be exceeded.

The calculation result stored in d and d+1 must not exceed the value range from 0 to 9999.

The result is calculated with a 5-digit accuracy and rounded to a 4-digit value.

BDSQR Square root calculation from 8-digit BCD data

The BDSQR instruction calculates the square root of s and s+1 and stores the result in d and d+1.



[1]Two-word data
[2]Integer part
[3]Decimal places

The data in s and s+1 must be a BCD value with at maximum 8 digits. The value range from 0 to 99999999 must not be exceeded.

The calculation result stored in d and d+1 must not exceed the value range from 0 to 9999.

The result is calculated with a 5-digit accuracy and rounded up to a 4-digit value.

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data stored in s (s+1) is no BCD data.
  (Error code 4100)

Program Example 1

BSQR

The following program calculates the square root of the BCD value 1325 and outputs the integer part of the result as 4-digit BCD value at Y50 through Y5F. The decimal places are output as 4-digit BCD value at Y40 through Y4F.

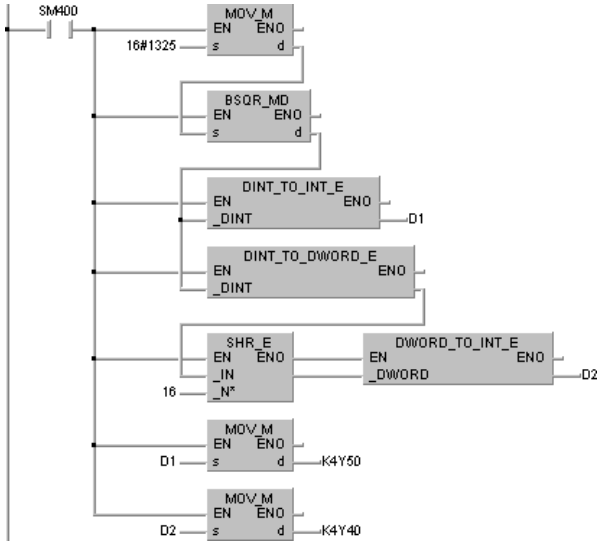
MELSEC Instruction List

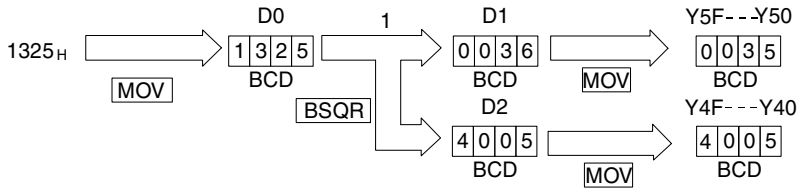
```
MELSEC   LD      SM400
         MOV     H1235
                 D0
         BSQR    D0
                 D1
         MOV     D1
                 K4Y50
         MOV     D2
                 K4Y40
```

IEC Instruction List

```
LD              SM400
MOV_M           H1325 , D0
BSQR_MD         D0 , var_D1
DINT_TO_INT_E   var_D1 , D1
DINT_TO_DWORD_E var_D1 , var_temp
SHR_E           var_temp , K16 , var_temp
DWORD_TO_INT_E  var_temp , D2
LD              SM400
MOV_M           D1 , K4Y50
MOV_M           D2 , K4Y40
```

Ladder Diagram





[1] Square root calculation

Program Example 2

BDSQR

The following program calculates the square root of the BCD value 74625813 and outputs the integer part of the result as 4-digit BCD value at Y50 through Y5F. The decimal places are output as 4-digit BCD value at Y40 through Y4F.

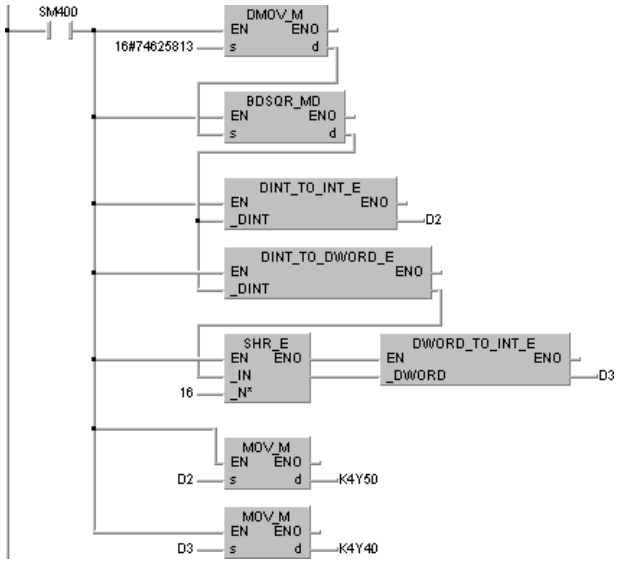
MELSEC Instruction List

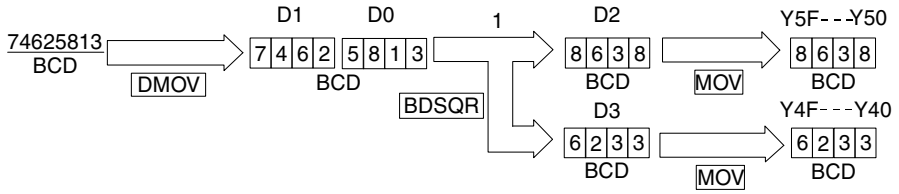
```
MELSEC  LD      SM400
        DMOV    H74625813
                D0
        BDSQR   D0
                D2
        MOV     D2
                K4Y50
        MOV     D3
                K4Y40
```

IEC Instruction List

```
LD                  SM400
DMOV_M              H74625813 , var_D0
BDSQR_MD            var_D0 , var_D2
DINT_TO_INT_E       var_D2 , D2
DINT_TO_DWORD_E     var_D2 , var_temp
SHR_E               var_temp , K16 , var_temp
DWORD_TO_INT_E      var_temp , D3
ST                  D3
LD                  SM400
MOV_M               D2 , K4Y50
MOV_M               D3 , K4Y40
```

Ladder Diagram





[1] Square root calculation

NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). They would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.29 BSIN, BSINP

CPU

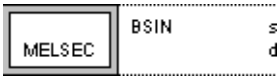
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

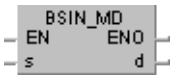
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H, (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

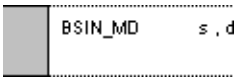
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

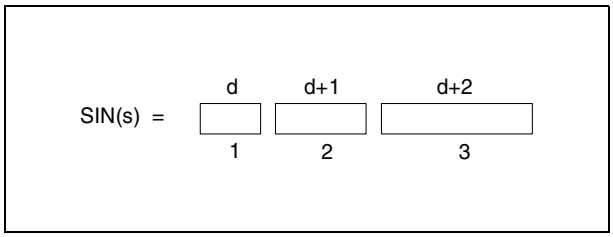
Set Data	Meaning	Data Type
s	Data of which the BSIN (sine) value is obtained or the number of the device storing such data	4-digit BCD value
d	First number of device storing the calculation result	

Functions

Sine calculation from BCD data

BSIN Sine calculation

The BSIN instruction calculates the sine value from the angle data in s. The sign character of the result is stored in d. The value of the result is stored in d+1 and d+2.



[1] Sign bit
[2] Integer part
[3] Decimal places

The value s must be a BCD value ranging from 0° to 360°.

The sign of the result in d is 0 for a positive value and 1 for a negative value.

The result in d+1 and d+2 may range from -1.000 to 1.000 in BCD format.

The calculation result will be rounded from the 5th digit on.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data specified in s is no BCD data. (Error code 4100)
- The data specified in s exceeds the value range from 0° to 360°. (Error code 4100)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

BSIN

The following program calculates the sine value of the 3-digit BCD value at X20 through X2B. If the value at X20 through X2B exceeds 360 (degrees), it will be corrected to suit the required value range of 0° to 360°.

The sign is output at Y60. The integer part is output at Y50 through Y53 as 1-digit BCD value.

The decimal places are output at Y40 through Y4F as 4-digit BCD value.

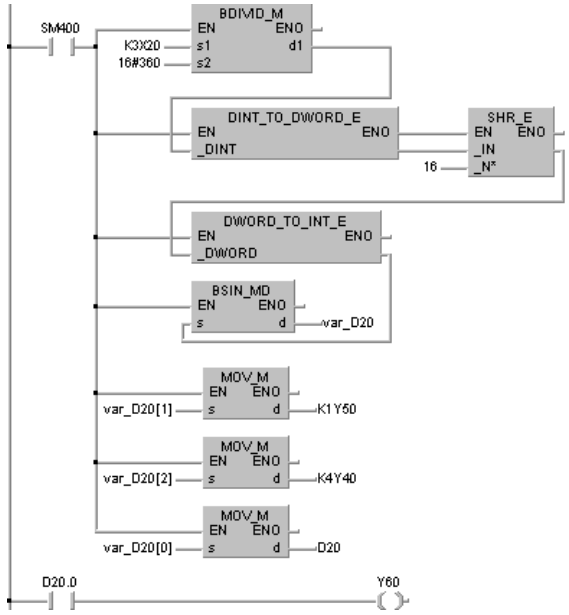
MELSEC Instruction List

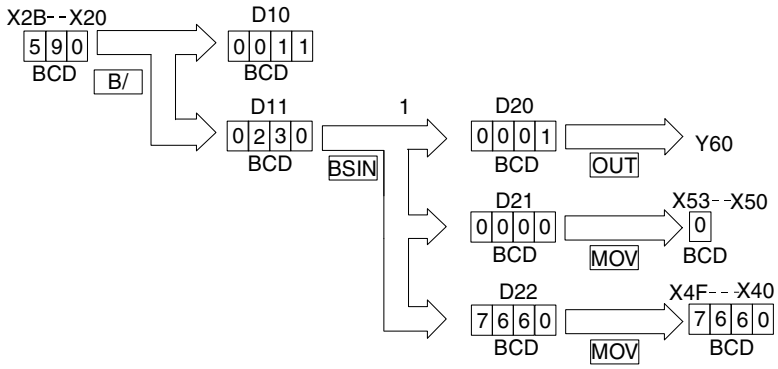
```
MELSEC  LD      SM400
        B/      K3X20
                H360
                D10
        BSIN    D11
                D20
        MOV     D21
                K1X50
        MOV     D22
                K4X40
        LD      D20.0
        OUT     Y60
```

IEC Instruction List

```
LD              SM400
BDIMD_M         K3X20 , H360 , var_D10
DINT_TO_DWORD_E var_D10 , var_Merker
SHR_E           var_Merker , K16 , var_Merker
DWORD_TO_INT_E  var_Merker , D11
LD              SM400
BSIN_MD         D11 , var_D20
MOV_M           var_D20[1] , K1Y50
MOV_M           var_D20[2] , K4Y40
MOV_M           var_D20[0] , D20
LD              D20.0
ST              Y60
```

Ladder Diagram





[1] Sine calculation

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.30 BCOS, BCOSP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

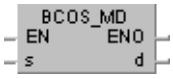
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H, (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—

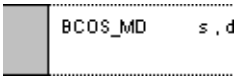
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

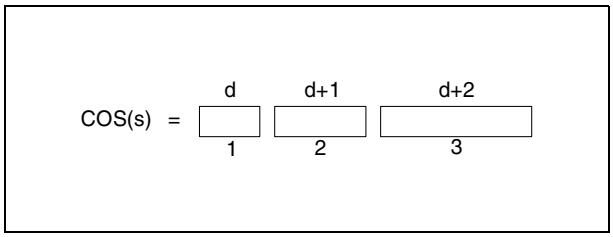
Variables

Set Data	Meaning	Data Type
s	Data of which the BCOS (cosine) value is obtained or the number of the device storing such data	4-digit BCD value
d	First number of device storing the calculation result	

Functions **Cosine calculation from BCD data**

BCOS Cosine calculation

The BCOS instruction calculates the cosine value from the angle data in s. The sign character of the result is stored in d. The value of the result is stored in d+1 and d+2.



[1] Sign bit
[2] Integer part
[3] Decimal places

The value s must be a BCD value ranging from 0° to 360°.

The sign of the result in d is 0 for a positive value and 1 for a negative value.

The result in d+1 and d+2 may range from -1.000 to 1.000 in BCD format.

The calculation result will be rounded from the 5th digit on.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data specified in s is no BCD data.
  (Error code 4100)
- The data specified in s exceeds the value range from 0° to 360°.
  (Error code 4100)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

BCOS

The following program calculates the cosine value of the 3-digit BCD value at X20 through X2B. If the value at X20 through X2B exceeds 360 (degrees), it will be corrected to suit the required value range of 0° to 360°.

The sign is output at Y60. The integer part is output at Y50 through Y53 as 1-digit BCD value.

The decimal places are output at Y40 through Y4F as 4-digit BCD value.

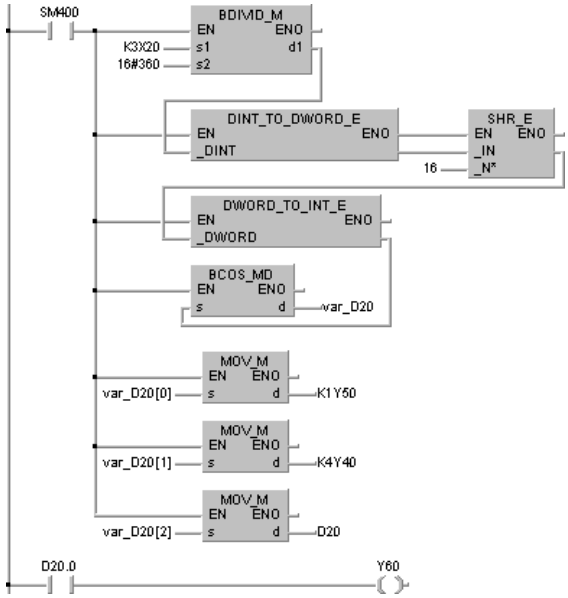
MELSEC Instruction List

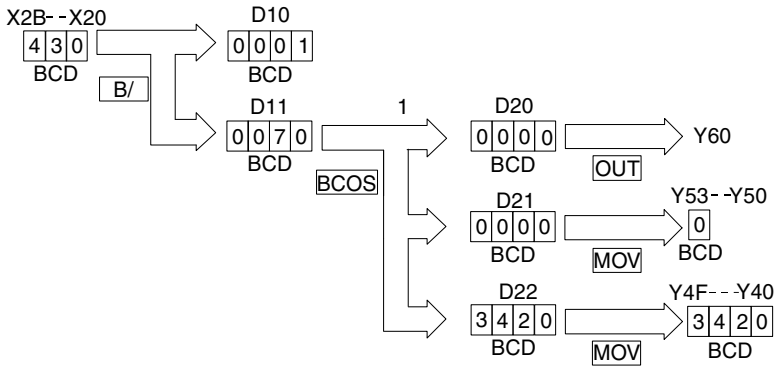
```
MELSEC   LD      SM400
         B/      K3X20
                 H360
                 D10
         BCOS    D11
                 D20
         MOV     D21
                 K1X50
         MOV     D22
                 K4X40
         LD      D20.0
         OUT     Y60
```

IEC Instruction List

```
LD              SM400
BDIVD_M         K3X20 , H360 , var_D10
DINT_TO_DWORD_E var_D10 , var_Merker
SHR_E           var_Merker , K16 , var_Merker
DWORD_TO_INT_E  var_Merker , D11
LD              SM400
BCOS_MD         D11 , var_D20
MOV_M           var_D20[0] , K1Y50
MOV_M           var_D20[1] , K4Y40
MOV_M           var_D20[2] , D20
LD              D20.0
ST              Y60
```

Ladder Diagram





[1] Cosine calculation

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.12.31 BTAN, BTANP

CPU

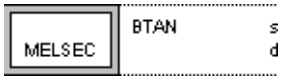
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

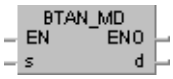
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H, (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

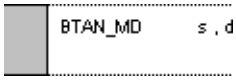
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

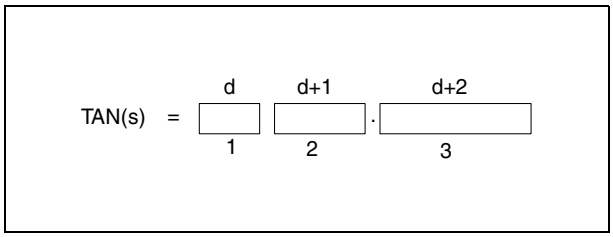
Set Data	Meaning	Data Type
s	Data of which the BTAN (tangent) value is obtained or the number of the device storing such data	4-digit BCD value
d	First number of device storing the calculation result	

Functions

Tangent calculation from BCD data

BTAN Tangent calculation

The BTAN instruction calculates the tangent value from the angle data in s. The sign character of the result is stored in d. The value of the result is stored in d+1 and d+2.



[1] Sign bit
[2] Integer part
[3] Decimal places

The value s must be a BCD value ranging from 0° to 360°.

The sign of the result in d is 0 for a positive value and 1 for a negative value.

The result in d+1 and d+2 may range from -57.2901 to 57.2902 in BCD format.

The calculation result will be rounded from the 5th digit on.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data specified in s is no BCD data.
  (Error code 4100)
- The data specified in s exceeds the value range from 0° to 360°.
  (Error code 4100)
- The value in s is 90° or 270°.
  (Error code 4100)
- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

BTAN

The following program calculates the tangent value of the 3-digit BCD value at X20 through X2B. If the value at X20 through X2B exceeds 360 (degrees), it will be corrected to suit the required value range of 0° to 360°.

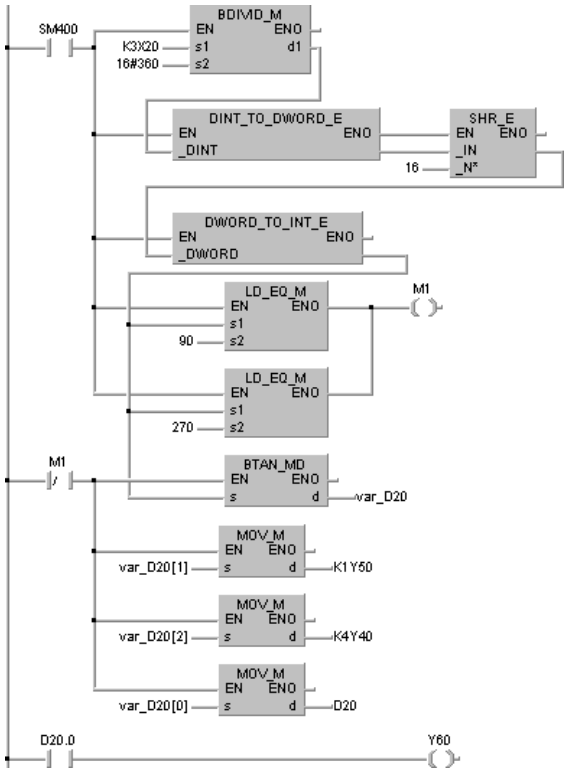
The sign is output at Y60. The integer part is output at Y50 through Y53 as 1-digit BCD value.

The decimal places are output at Y40 through Y4F as 4-digit BCD value.

MELSEC Instruction List

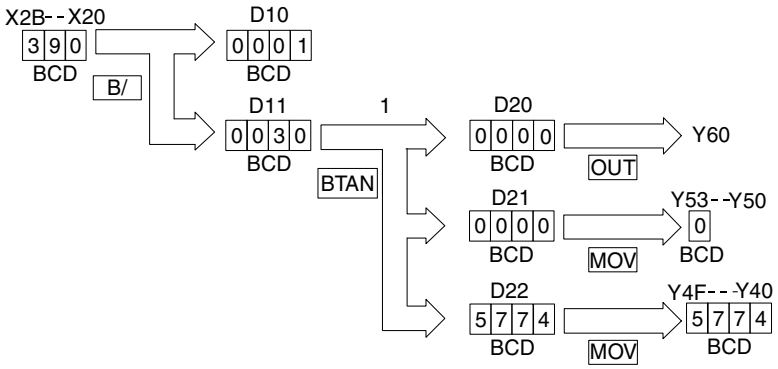
```
MELSEC
        LD      SM400
        B/      K3X20
                H360
                D10
        LD=     D11
                K90
        OR=     D11
                K270
        OUT     M1
        LDI     M1
        BTAN    D11
                D20
        MOV     D21
                K1X50
        MOV     D22
                K4X40
        LD      D20.0
        OUT     Y60
```

Ladder Diagram



IEC Instruction List

```
LD              SM400
BDIVD_M         K3X20 , H360 , var_D10
DINT_TO_DWORD_E var_D10 , var_Merker
SHR_E           var_Merker , K16 , var_Merker
DWORD_TO_INT_E  var_Merker , D11
LD              TRUE
LD_EQ_M         D11 , K90
OR_EQ_M         D11 , K270
ST              M1
LDN             M1
BTAN_MD         D11 , var_D20
MOV_M           var_D20[1] , K1Y50
MOV_M           var_D20[2] , K4Y40
MOV_M           var_D20[0] , D20
LD              D20.0
ST              Y60
```



[1] Tangent calculation

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.32 BASIN, BASINP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

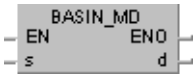
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H, (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	●	●	●	●	●	●	●	—	—

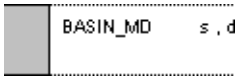
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

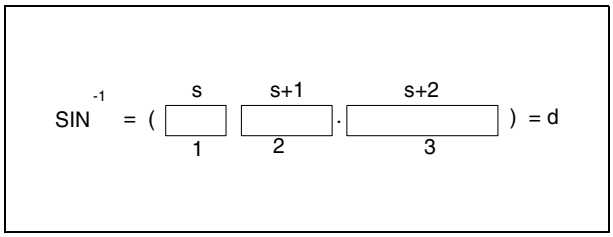
Variables

Set Data	Meaning	Data Type
s	Number of device storing the sine value for the BASIN instruction (arcus sine)	4-digit BCD value
d	First number of device storing the calculation result	

Functions **Arcus sine calculation from BCD data**

BASIN Arcus sine calculation

The BASIN instruction calculates the angle data from the sine value in s, s+1, and s+2. The result is stored in d.



[1] Sign bit
[2] Integer part
[3] Decimal places

The sign of the result in s is 0 for a positive value and 1 for a negative value.

The integer part prior to the decimal point and the decimal places must be BCD values ranging from 0 to 1.0000.

The value or the result in d must be a BCD value ranging from 0° to 90° or from 270° to 360°.

The calculation result will be rounded from the 5th digit on.

Operation Errors In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data specified in s through s+2 is no BCD data.
  (Error code 4100)
- The data specified in s through s+2 exceeds the value range from -1.0000 to 1.0000.
  (Error code 4100)
- The device specified by s exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU.)
  (Error code 4101)

Program Example

BASIN

The following program calculates the arcus sine value from the sign bit at X0 (positive when X0 is OFF, and negative when X0 is ON), the 1-digit BCD integer part at X30 through X33, and the decimal places of the 4-digit BCD value at X20 through X2F. The resulting angle value is output in 4-digit BCD format at Y40 through Y4F.

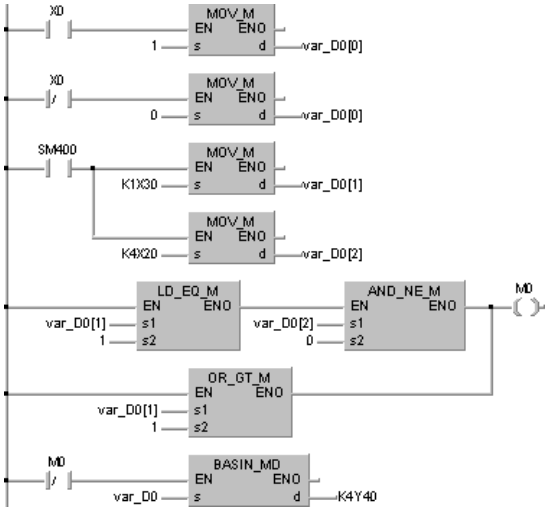
MELSEC Instruction List

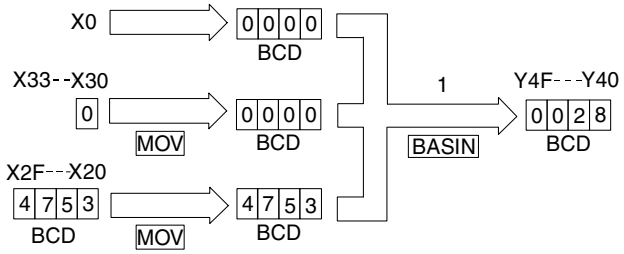
```
MELSEC  LD      X0
        MOV     K1
                D0
        LDI     X0
        MOV     K0
                D0
        LD      SM400
        MOV     K1X30
                D1
        MOV     K4X20
                D2
        LD=     D1
                K1
        AND<>   D2
                K0
        OR>     D1
                K1
        OUT     M0
        LDI     M0
        BASIN   D0
                K4Y40
```

IEC Instruction List

```
LD          X0
MOV_M       1 , var_D0[0]
LDN         X0
MOV_M       0 , var_D0[0]
LD          SM400
MOV_M       K1X30 , var_D0[1]
MOV_M       K4X20 , var_D0[2]
LD_EQ_M     var_D0[1] , 1
AND_NE_M    var_D0[2] , 0
OR_GT_M     var_D0[1] , 1
ST          M0
LDN         M0
BASIN_MD    var_D0 , K4Y40
```

Ladder Diagram





[1] Arcus sine calculation

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.33 BACOS, BACOSP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

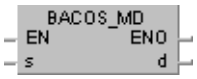
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H, (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	●	●	●	●	●	●	●	—	—

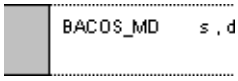
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

Variables

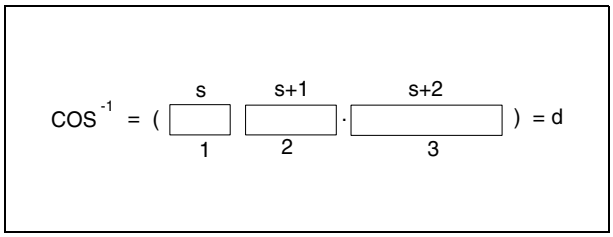
Set Data	Meaning	Data Type
s	Number of device storing the cosine value for the BACOS instruction (arcus cosine)	4-digit BCD value
d	First number of device storing the calculation result	

Functions

Arcus cosine calculation from BCD data

BACOS Arcus cosine calculation

The BACOS instruction calculates the angle data from the cosine value in s, s+1, and s+2. The result is stored in d.



[1] Sign bit
[2] Integer part
[3] Decimal places

The sign of the result in s is 0 for a positive value and 1 for a negative value.

The integer part prior to the decimal point and the decimal places must be BCD values ranging from 0 to 1.0000.

The value or the result in d must be a BCD value ranging from 0° to 180.

The calculation result will be rounded from the 5th digit on.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data specified in s through s+2 is no BCD data.
  (Error code 4100)
- The data specified in s through s+2 exceeds the value range from -1.000 to 1.000.
  (Error code 4100)
- The device specified by s exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

BACOS

The following program calculates the arcus cosine value from the sign bit at X0 (positive when X0 is OFF and negative when X0 is ON), the 1-digit BCD integer part at X30 through X33, and the decimal places of the 4-digit BCD value at X20 through X2F. The resulting angle value is output in 4-digit BCD format at Y40 through Y4F.

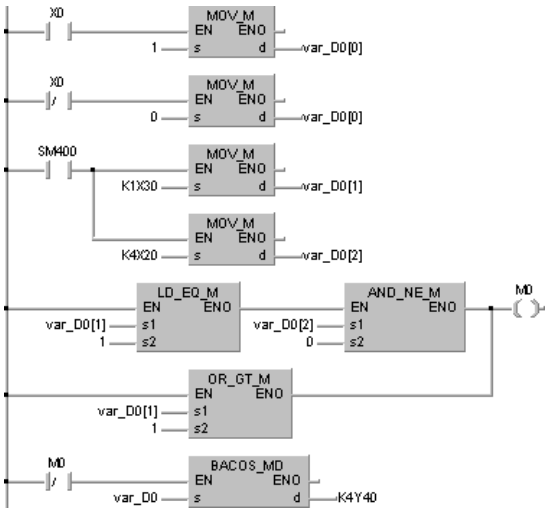
MELSEC Instruction List

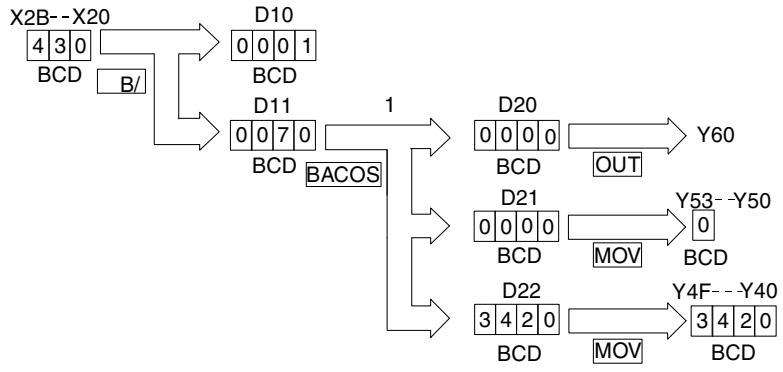
```
MELSEC  LD      X0
        MOV     K1
                D0
        LDI     X0
        MOV     K0
                D0
        LD      SM400
        MOV     K1X30
                D1
        MOV     K4X20
                D2
        LD=     D1
                K1
        AND<>   D2
                K0
        OR>     D1
                K1
        OUT     M0
        LDI     M0
        BACOS   D0
                K4Y40
```

IEC Instruction List

```
LD          X0
MOV_M       1 , var_D0[0]
LDN         X0
MOV_M       0 , var_D0[0]
LD          SM400
MOV_M       K1X30 , var_D0[1]
MOV_M       K4X20 , var_D0[2]
LD_EQ_M     var_D0[1] , 1
AND_NE_M    var_D0[2] , 0
OR_GT_M     var_D0[1] , 1
ST          M0
LDN         M0
BACOS_MD    var_D0 , K4Y40
```

Ladder Diagram





[1] Arcus cosine calculation

NOTE *This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.12.34 BATAN, BATANP

CPU

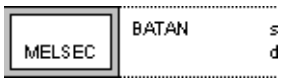
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d	●	●	●	●	●	●	●	—	—

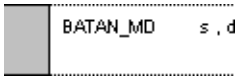
GX IEC Developer

MELSEC Instruction List



Ladder Diagram

IEC Instruction List



GX Works2

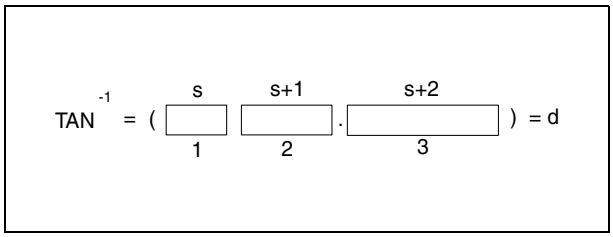
Variables

Set Data	Meaning	Data Type
s	Number of device storing the tangent value for the BATAN instruction (arcus tangent)	4-digit BCD value
d	First number of device storing the calculation result	

Functions **Arcus tangent calculation from BCD data**

BATAN Arcus tangent calculation

The BATAN calculates the angle data from the tangent value in s, s+1, and s+2. The result is stored in d.



[1] Sign bit
[2] Integer part
[3] Decimal places

The sign bit of the result in s is 0 for a positive value and 1 for a negative value.

The integer part prior to the decimal point and the decimal places must be BCD values ranging from 0 to 9999.9999.

The value of the result in d must be a BCD value ranging from 0° to 90° or 270° or from 270° and 360°.

The calculation result will be rounded from the 5th digit on.

Operation Errors In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

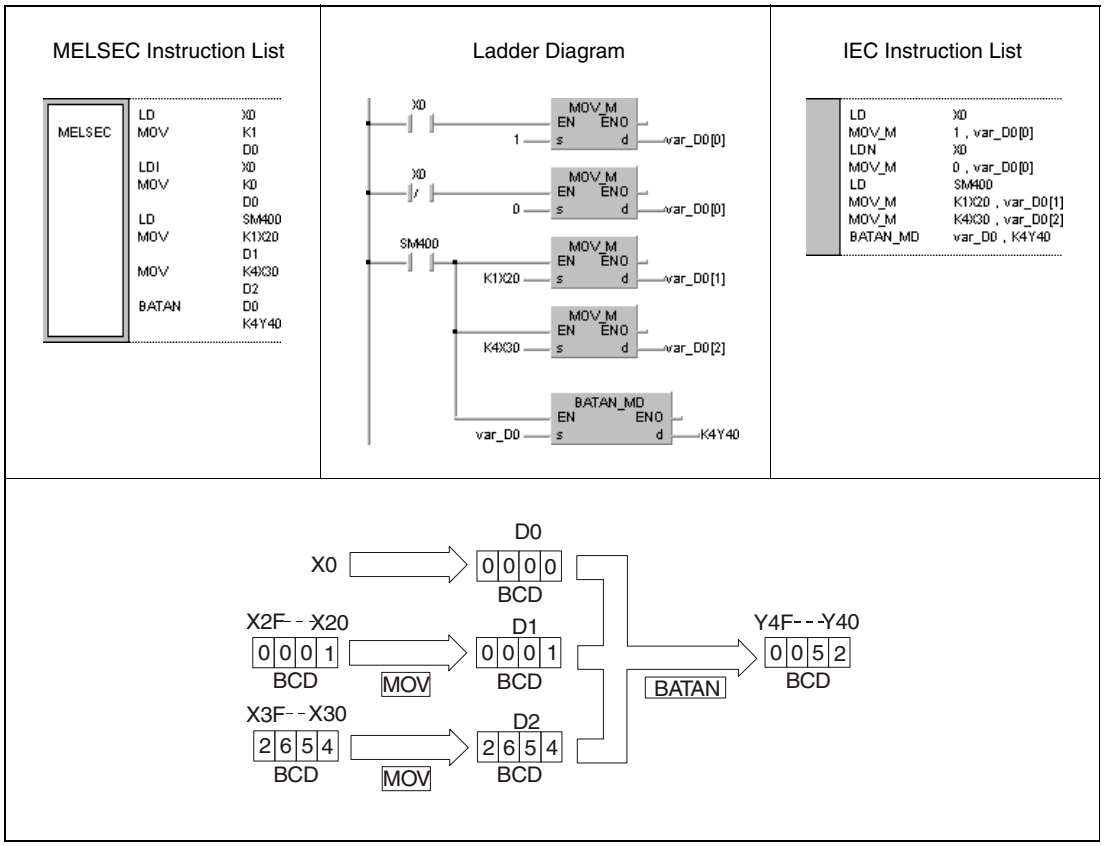
- The data specified in s through s+2 is no BCD data. (Error code 4100)
- The device specified by s exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

BATAN

The following program calculates the arcus tangent value from the sign bit at X0 (positive when X0 is OFF and negative when X0 is ON), the 4-digit BCD integer part at X20 through X2F, and the decimal places of the 4-digit BCD value at X30 through X3F.

The resulting angle value is output in 4-digit BCD format at Y40 through Y4F.



[1] Arcus tangent calculation

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

7.13 Data control instructions

The data control instructions include input and output devices. The 16-bit and 32-bit data of the input devices are output to the output devices via parameters controlling the upper and lower limits, the dead band, the zone or after execution of a scaling operation.

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Upper and lower limit controls for BIN 16-/32-bit data	LIMIT	LIMIT_MD
	LIMITP	LIMIT_P_MD
	DLIMIT	DLIMIT_MD
	DLIMITP	DLIMIT_P_MD
Dead band controls for BIN 16-/32-bit data	BAND	BAND_MD
	BANDP	BAND_P_MD
	DBAND	DBAND_MD
	DBANDP	DBAND_P_MD
Zone control for BIN 16-/32-bit data	ZONE	ZONE_MD
	ZONEP	ZONE_P_MD
	DZONE	DZONE_MD
	DZONEP	DZONE_P_MD
Scaling (Point-by point coordinate data)	SCL	
	SCLP	
	DSCL	
	DSCLP	
Scaling (X or Y coordinate data)	SCL2	
	SCL2P	
	DSCL2	
	DSCL2P	

NOTE *Within the IEC editors please use the IEC instructions.*

7.13.1 LIMIT, LIMITP, DLIMIT, DLIMITP

CPU

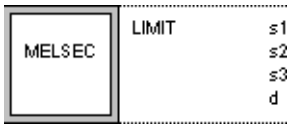
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

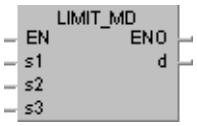
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
s3	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer

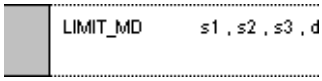
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

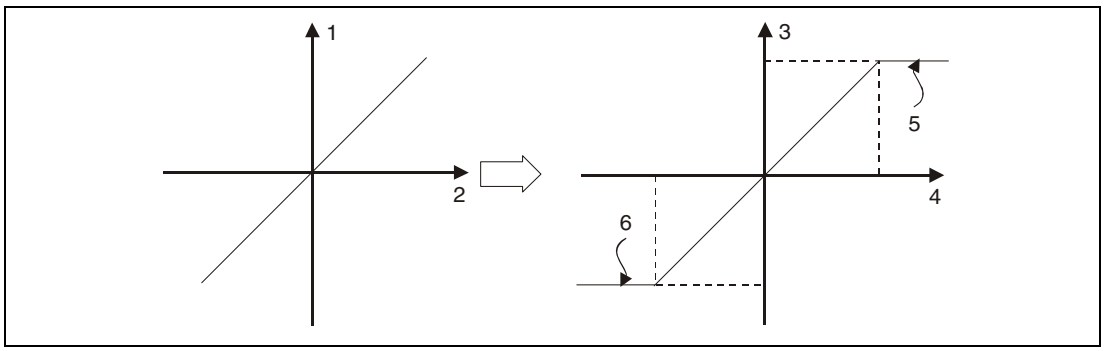
Set Data	Meaning	Data Type
s1	Lower limit value (minimum output threshold value)	BIN 16/32-bit
s2	Upper limit value (maximum output threshold value)	
s3	Input value to be limited	
d	First number of device storing limited output value	

Functions **Limitation of output values for BIN 16-bit and BIN 32-bit data**

LIMIT Limitation instruction for BIN 16-bit data

The LIMIT instruction controls whether data in the device specified by s3 ranges within the lower limits specified by s1 and the upper limits specified by s2. Depending on the control operation result the values are stored as follows in the device specified by d:

- If the data value in s3 is less than the lower limit value in s1, the lower limit value is stored in d.
- If the data value in s3 is greater than the upper limit value in s2, the upper limit value is stored in d.
- If the data value in s3 ranges within the lower and the upper limit value, the data value is stored in d.



[1] Output value
[2] Input value
[3] Output value (d)
[4] Input value (s3)
[5] Upper limit value (s2)
[6] Lower limit value (s1)

The values specified by s1, s2, and s3 have to range within -32768 and 32767.

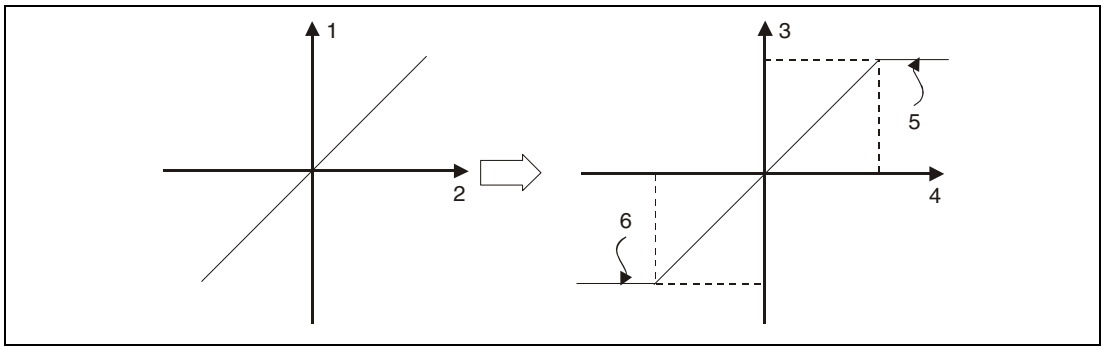
If only the upper limit value is to be checked, the lower limit value in s1 has to be set to -32768.

If only the lower limit value is to be checked, the upper limit value in s2 has to be set to 32767.

DLIMIT Limitation instruction for BIN 32-bit data

The DLIMIT instruction controls whether data in the devices specified by s3 and (s3)+1 range within the lower limits specified by s1 and (s1)+1 and the upper limits specified by s2 and (s2)+1. Depending on the control operation result the values are stored as follows in the device specified by d and d+1:

- If the data value in s3 and (s3)+1 is less than the lower limit value in s1and (s1)+1, the lower limit value is stored in d and d+1.
- If the data value in s3 and (s3)+1 is greater than the upper limit value in s2 and (s2)+1, the upper limit value is stored in d and d+1.
- If the data value in s3 and (s3)+1 ranges within the lower and the upper limit value, the data value is stored in d and d+1.



[1] Output value
[2] Input value
[3] Output value (d+1, d)
[4] Input value ((s3)+1, s3)
[5] Upper limit value ((s2)+1, s2)
[6] Lower limit value ((s1)+1, s1)

The values specified by s1 and (s1)+1, s2 and (s2)+1, and s3 and (s3)+1 have to range within -2147483648 and 2147483647.

If only the upper limit value is to be checked, the lower limit value in s1 and (s1)+1 has to be set to -2147483648.

If only the lower limit value is to be checked, the upper limit value in s2 and (s2)+1 has to be set to 2147483647.

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value in s1 ((s1)+1) is greater than that in s2 ((s2)+1).
  (Error code 4100)

Program Example 1

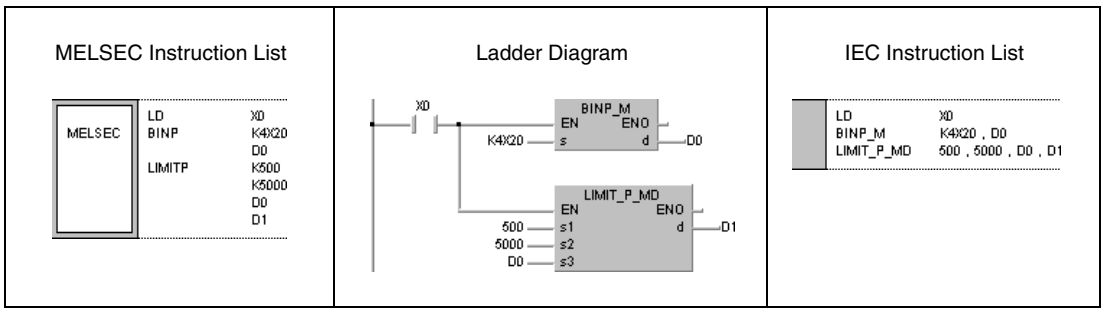
LIMITP

With leading edge from X0, the following program controls whether BCD data at X20 through X2F ranges between the lower limit of 500 and the upper limit of 5000. The result of the control operation is stored in D1.

– If the value in D0 is greater than 5000, the value 5000 is stored in D1.

– If the value in D0 is less than 500, the value 500 is stored in D1.

– If the value ranges within 500 and 5000, the data value is stored in D1.



Program Example 2

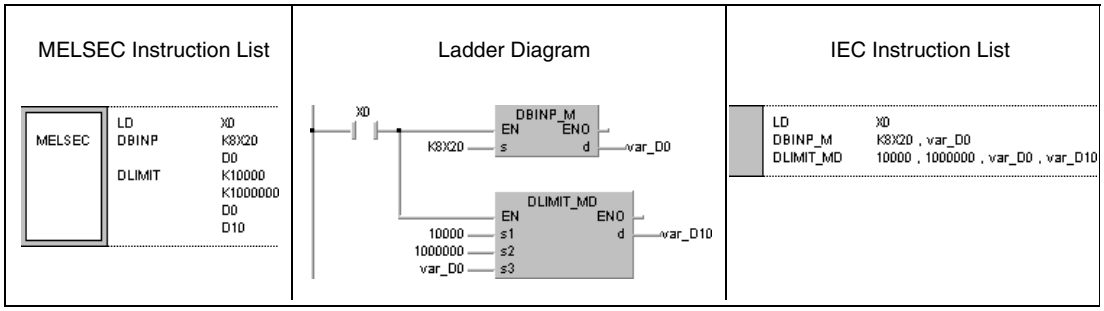
DLIMIT

With leading edge from X0, the following program controls whether BCD data at X20 through X3F ranges within the lower limit of 10000 and the upper limit of 1000000. The result of the control operation is stored in D10 and D11.

– If the value in D0 and D1 is greater than 1000000, the value 1000000 is stored in D10 and D11.

– If the value in D0 and D1 is less than 10000, the value 10000 is stored in D10 and D11.

– If the value ranges within 10000 and 1000000, the data value is stored in D10 and D11.



7.13.2 BAND, BANDP, DBAND, DBANDP

CPU

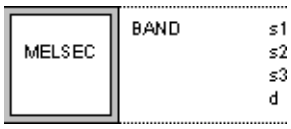
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

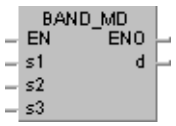
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
s3	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer

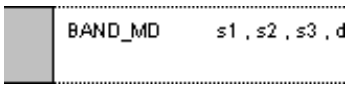
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

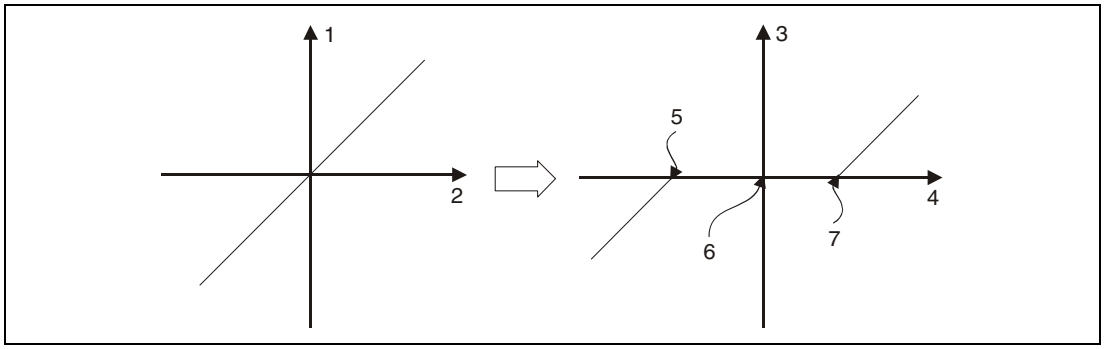
Set Data	Meaning	Data Type
s1	Lower limit value of dead band (output value = 0)	BIN 16/32-bit
s2	Upper limit value of dead band (output value = 0)	
s3	Input value to be controlled via dead band control	
d	First number of device storing subtraction result of input value minus limit value	

Functions **BIN 16-bit and 32-bit dead band control**

BAND Dead band control of BIN 16-bit data

The BAND instruction subtracts a lower and an upper limit value from a BIN 16-bit value in a device specified by s3. The lower limit value is specified by s1; the upper limit value is specified by s2. The result is stored depending on the input value in the device specified by d as follows:

- If the data value in s3 is less than the lower limit value in s1, the result of the subtraction s3-s1 is stored in the device specified by d.
- If the data value in s3 is greater than the upper limit value in s2, the result of the subtraction s3-s2 is stored in the device specified by d.
- If the data value in s3 ranges within the limit values, the value 0 is stored in the device specified by d.



[1] Output value
[2] Input value
[3] Output value (d)
[4] Input value (s3)
[5] Lower limit value (s1)
[6] Output value = 0
[7] Upper limit value (s2)

The values in s1, s2, and s3 have to range within -32768 and 32767.

If the subtraction result leaves the relevant device range of -32768 and 32767 the output value is controlled as follows:

- If the value -32768 is fallen below, the remaining subtraction is proceeded beginning from 32767. For example, if s3 stores the value -32768 and the value 10 in s1 is subtracted, the result is

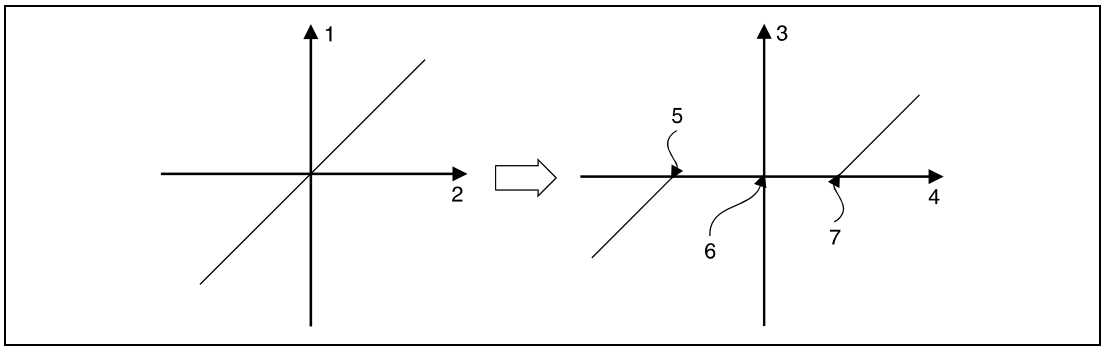
$$-32768 - 10 = 8000_H - A_H = 7FF6_H = 32758.$$

- If the value 32767 is exceeded, the remaining subtraction is proceeded beginning from -32768.

DBAND Dead band control of BIN 32-bit data

The DBAND instruction subtracts a lower and an upper limit value from a BIN 32-bit value in a device specified by s3 and (s3)+1. The lower limit value is specified by s1 and (s1)+1; the upper limit value is specified by s2 and (s2)+1. The result is stored depending on the input value in the device specified by d and d+1 as follows:

- If the data value in s3 and (s3)+1 is less than the lower limit value in s1 and (s1)+1, the result of the subtraction s3, (s3)+1 - s1, (s1)+1 is stored in the device specified by d and d+1.
- If the data value in s3 and (s3)+1 is greater than the upper limit value in s2 and (s2)+1, the result of the subtraction s3, (s3)+1 - s2, (s2)+1 is stored in the device specified by d and d+1.
- If the data value in s3 and (s3)+1 ranges within the limit values, the value 0 is stored in the device specified by d and d+1.



[1] Output value
[2] Input value
[3] Output value (d+1, d)
[4] Input value ((s3)+1, s3)
[5] Lower limit value ((s1)+1, s1)
[6] Output value = 0
[7] Upper limit value ((s2)+1, s2)

The values in s1 and (s1)+1, s2 and (s2)+1, and s3 and (s3)+1 have to range within -2147483648 and 2147483647.

If the subtraction result leaves the relevant device range of -2147483648 and 2147483647 the output value is controlled as follows:

- If the value -2147483648 is fallen below, the remaining subtraction is proceeded beginning from 2147483647. For example, if s3 and (s3)+1 store the value -2147483648 and the value 1000 in s1 is subtracted, the result is

$$-2147483648 - 1000 = 80000000_H - 3E8_H = 7FFFFC18_H = 2147482648.$$

- If the value 2147483647 is exceeded, the remaining subtraction is proceeded beginning from -2147483648.

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The value in s1 ((s1)+1) is greater than that in s2 ((s2)+1).
  (Error code 4100)

Program Example 1

BANDP

With leading edge from X0, the following program subtracts the lower limit value -1000 and the upper limit value 1000 from the BCD data at X20 through X2F. The result is stored in D1.

– If the value in D0 is greater than 1000, the value D0 - 1000 is stored in D1.

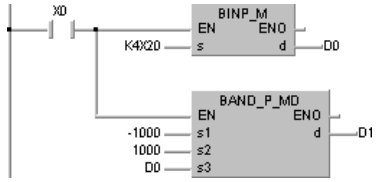
– If the value in D0 is less than -1000, the value D0 - (-1000) is stored in D1.

– If the value in D0 ranges within -1000 and 1000, the value 0 is stored in D1.

MELSEC Instruction List

```
MELSEC  LD      X0
        BINP    K4X20
                D0
        BANDP   K-1000
                K1000
                D0
                D1
```

Ladder Diagram



IEC Instruction List

```
LD          X0
BINP_M      K4X20 , D0
BAND_P_MD   -1000 , 1000 , D0 , D1
```

Program Example 2

DBANDP

With leading edge from X0, the following program subtracts the lower limit value -10000 and the upper limit value 10000 from the BCD data at X20 through X3F. The result is stored in D10 and D11.

– If the value in D0 and D1 is greater than 10000, the value D0, D1 - 1000 is stored in D10 and D11.

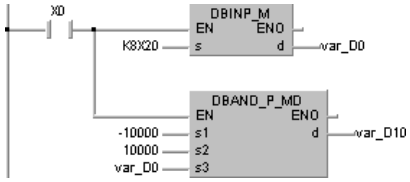
– If the value in D0 and D1 is less than -10000, the value D0, D1 - (-10000) is stored in D10 and D11.

– If the value in D0 and D1 ranges within -10000 and 1000, the value 0 is stored in D10 and D11.

MELSEC Instruction List

```
MELSEC  LD      X0
        DBINP   K8X20
                D0
        DBANDP  K-10000
                K10000
                D0
                D10
```

Ladder Diagram



IEC Instruction List

```
LD          X0
DBINP_M     K8X20 , var_D0
DBAND_P_MD  -10000 , 10000 , var_D0 , var_D10
```

7.13.3 ZONE, ZONEP, DZONE, DZONEP

CPU

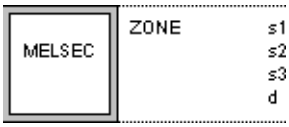
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

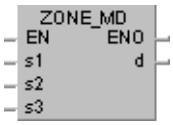
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	●	●	●	●	●	●	●	●	—
s2	●	●	●	●	●	●	●	●	—
s3	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

GX IEC Developer

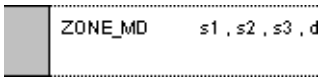
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

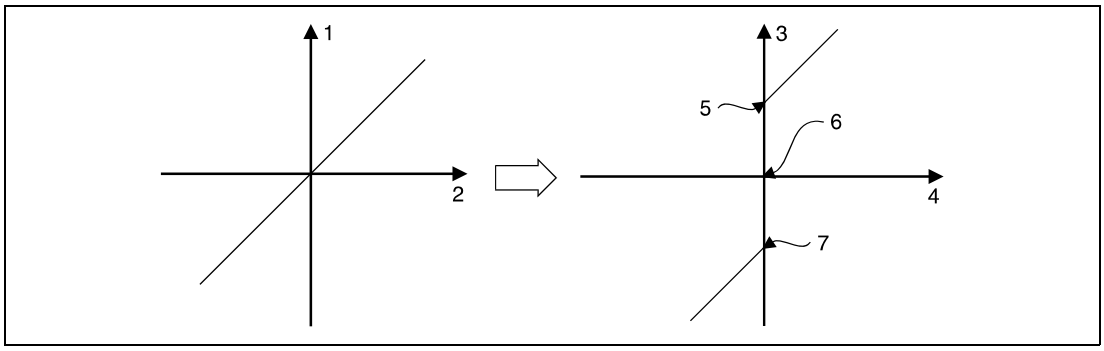
Set Data	Meaning	Data Type
s1	Negative zone control value to be added to the input value	BIN 16/32-Bit
s2	Positive zone control value to be added to the input value	
s3	Input value to be controlled via zone control	
d	First number of device storing total of input value and zone control value	

Functions **BIN 16-bit and 32-bit zone control**

ZONE Zone control of BIN 16-bit data

The ZONE instruction adds a negative and a positive control value to a BIN 16-bit value in a device specified by s3. The negative control value is stored in s1; the positive control value is stored in s2. The result is stored depending on the input value in the device specified by d as follows:

- If the data value in s3 is less than 0, the result of the addition s3 + s1 is stored in the device specified by d.
- If the data value in s3 is greater than 0, the result of the addition s3 + s2 is stored in the device specified by d.
- If the data value in s3 is equal to 0, the value 0 is stored in the device specified by d.



[1] Output value
[2] Input value
[3] Output value (d)
[4] Input value (s3)
[5] Upper (positive) zone control value (s2)
[6] Input value = 0
[7] Lower (negative) zone control value (s1)

The values in s1, s2, and s3 have to range within -32768 and 32767.

If the addition result leaves the relevant device range of -32768 and 32767, the output value is controlled as follows:

- If the value -32768 is fallen below, the remaining addition is proceeded beginning from 32767. For example, if s3 stores the value -32768 and the value -100 in s1 is added, the result is

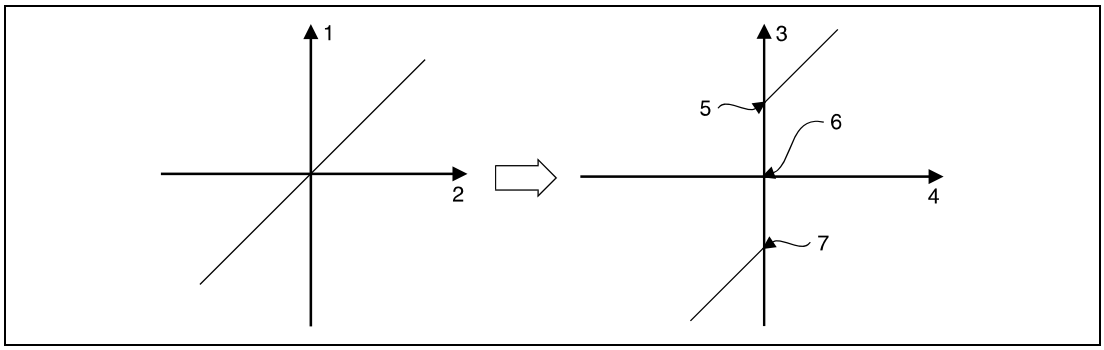
$$-32768 + (-100) = 8000_H + FF9C_H = 7F9C_H = 32668.$$

- If the value 32767 is exceeded, the remaining addition is proceeded beginning from -32768.

DZONE Zone control of BIN 32-bit data

The DZONE instruction adds a negative and a positive control value to a BIN 32-bit value in a device specified by s3 and (s3)+1. The negative control value is stored in s1 and (s1)+1; the positive control value is stored in s2 and (s2)+1. The result is stored depending on the input value in the device specified by d and d+1 as follows:

- If the data value in s3 and (s3)+1 is less than 0, the result of the addition s3, (s3)+1 + s1, (s1)+1 is stored in the device specified by d and d+1.
- If the data value in s3 and (s3)+1 is greater than 0, the result of the addition s3, (s3)+1 + s2, (s2)+1 is stored in the device specified by d+1.
- If the data value in s3 and (s3)+1 is equal to 0, the value 0 is stored in the device specified by d and d+1.



[1] Output value
[2] Input value
[3] Output value (d+1, d)
[4] Input value ((s3)+1, s3)
[5] Upper (positive) zone control value ((s2)+1, s2)
[6] Input value = 0
[7] Lower (negative) zone control value ((s1)+1, s1)

The values in s1 and (s1)+1, s2 and (s2)+1, and s3 and (s3)+1 have to range within -2147483648 and 2147483647.

If the addition result leaves the relevant device range of -2147483648 and 2147483647 the output value is controlled as follows:

- If the value -2147483648 is fallen below, the remaining addition is proceeded beginning from 2147483647. For example, if s3 and (s3)+1 store the value -2147483648 and the value -1000 in s1 is added, the result is

  $-2147483648 + (-1000) = 80000000_H + FFFFFC18_H = 7FFFFC18_H = 2147482648.$

- If the value 2147483647 is exceeded, the remaining addition is proceeded beginning from -2147483648.

Program Example 1

ZONEP

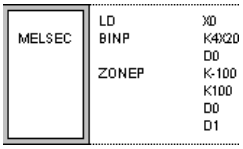
With leading edge from X0, the following program adds the negative zone control value -100 and the positive zone control value 100 to BCD data at X20 through X2F. The result is stored in D1.

– If the value in D0 is greater than 0, the value D0 + 100 is stored in D1.

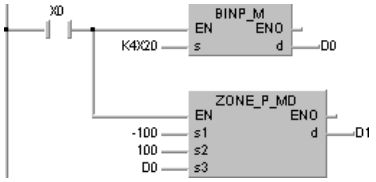
– If the value in D0 is less than 0, the value D0 + (-100) is stored in D1.

– If the value D0 is equal to 0, the value 0 is stored in D1.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD          X0
BINP_M      K4X20 , D0
ZONE_P_MD   -100 , 100 , D0 , D1
```

Program Example 2

DZONEP

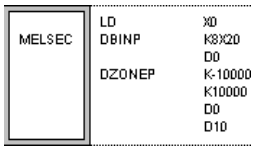
With leading edge from X0, the following program adds the negative zone control value -10000 and the positive zone control value 10000 to BCD data at X20 through X3F. The result is stored in D10 and D11.

– If the value in D0 and D1 is greater than 0, the value D0, D1 + 10000 is stored in D10 and D11.

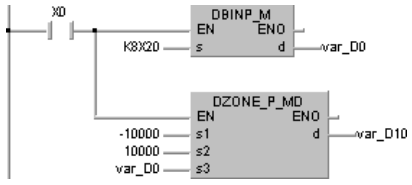
– If the value in D0 and D1 is less than 0, the value D0, D1 + (-10000) is stored in D10 and D11.

– If the value D0 and D1 is equal to 0, the value 0 is stored in D10 and D11.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

```
LD          X0
DBINP_M     K8X20 , var_D0
DZONE_P_MD  -10000 , 10000 , var_D0 , var_D10
```

7.13.4 SCL, SCLP, DSCL, DSCLP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

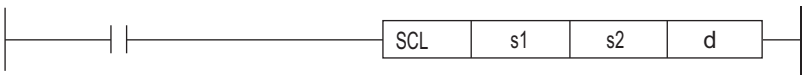
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
s1	—	●	●	●	●	●	●	●	—
s2	—	●	●	—	—	—	—	—	—
d	—	●	●	●	●	●	●	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s1	Input values for scaling or first number of the device where input values are stored	BIN 16/32-Bit
s2	First number of the devices where scaling conversion data are stored	
d	First number of of the devices where output values depending on scaling are stored	

Functions **Scaling (Point-by-point coordinate data)**

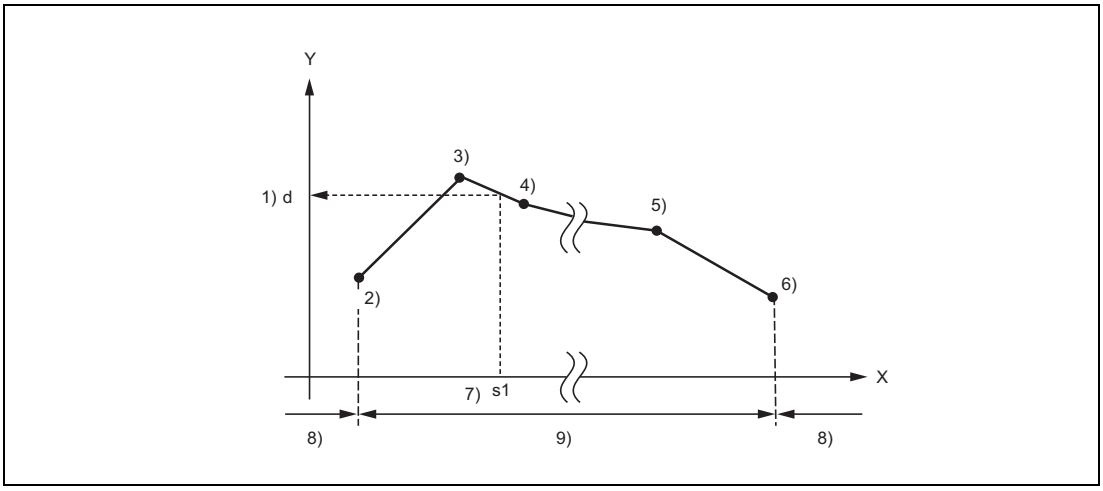
SCL Scaling of BIN 16-bit data

This instruction executes scaling for the scaling conversion data (16-bit data units) specified by s2 with the input value specified by s1, and then stores the operation result into the devices specified by d.

The scaling conversion is executed based on the scaling conversion data stored in the device specified by s2 and up (see following table).

Setting Item		Device Assignment
Number of coordinate points		s2
Point 1	X coordinate	(s2)+1
	Y coordinate	(s2)+2
Point 2	X coordinate	(s2)+3
	Y coordinate	(s2)+4
...		
Point n [1)]	X coordinate	(s2)+(2n–1)
	Y coordinate	(s2)+2n

[1] n indicates the number of coordinates specified by s2.



[1] Output value (d)
[2] Point 1
[3] Point 2
[4] Point 3
[5] Point n–1
[6] Point n
[7] Input value (s1)
[8] Operation error
[9] Operable range

If the value does not result in an integer, this instruction rounds the value to the whole number.

Set the X coordinate of the scaling conversion data in ascending order.

Set the input value s1 within the range of the scaling conversion data (within the range of s2 devices).

If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.

Specify the number of coordinate points of scaling conversion data from 1 to 32767.

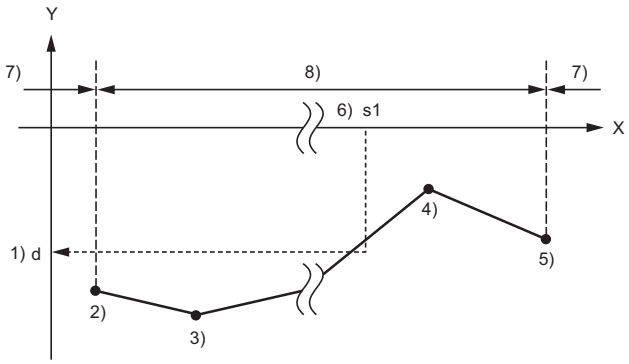
DSCL Scaling of BIN 32-bit data

This instruction executes scaling for the scaling conversion data (32-bit data units) specified by s2 with the input value specified by s1, and then stores the operation result into the devices specified by d.

The scaling conversion is executed based on the scaling conversion data stored in the device specified by s2 and up (see following table).

Setting Item		Device Assignment
Number of coordinate points		(s2)+1, (s2)
Point 1	X coordinate	(s2)+3, (s2)+2
	Y coordinate	(s2)+5, (s2)+4
Point 2	X coordinate	(s2)+7, (s2)+6
	Y coordinate	(s2)+9, (s2)+8
...		
Point n [1)]	X coordinate	(s2)+(4n–1), (s2)+(4n–2)
	Y coordinate	(s2)+(4n+1), (s2)+4n

[1] n indicates the number of coordinates specified by s2.



[1] Output value (d)
[2] Point 1
[3] Point 2
[4] Point n–1
[5] Point n
[6] Input value (s1)
[7] Operation error
[8] Operable range

If the value does not result in an integer, this instruction rounds the value to the whole number.

Set the X coordinate of the scaling conversion data in ascending order.

Set the input value s1 within the range of the scaling conversion data (within the range of s2 and ((s2)+1) devices).

If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.

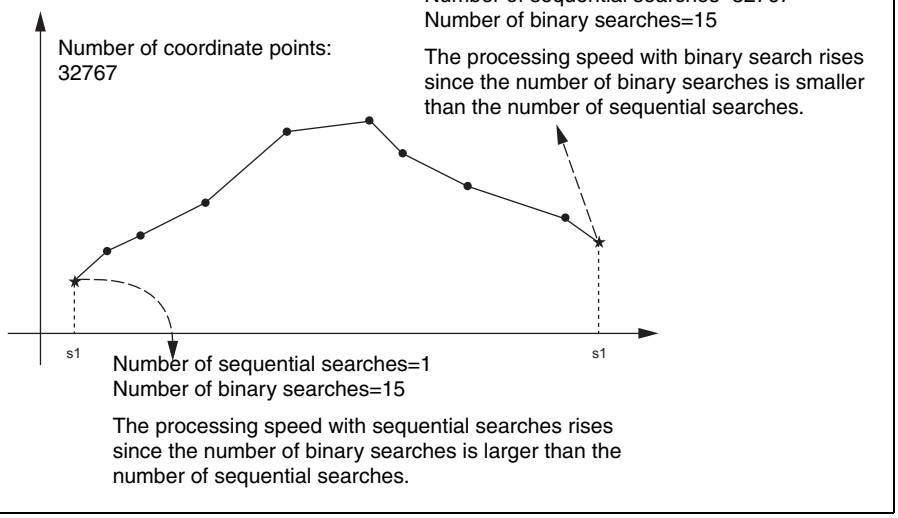
Specify the number of coordinate points of scaling conversion data from 1 to 32767.

NOTE *There are two searching methods that depend on whether SM750 is on or off.*

SM750	Searching method	Range of number of searches
OFF (0)	Sequential search	1 <= Number of times <= 32767
ON (1)	Binary search	1 <= Number of times <= 15

When the scaling conversion data are set in ascending order, the searching methods change from one to the other depending on the SM750 status. Therefore, the processing speed also changes. The number of searches determines the processing speed. Fewer number of serches make the processing run faster.

- *If the data processing speed with the sequential search rises:*
  If the number of coordinates is highest and the input value s1 is within the coordinate range from 1 to 15 point, the number of sequential searches will be 15 or smaller. Therefore, the data processing speed with the sequential search will rise.
- *If the data processing speed with the binary search rises:*
  If the maximum number of searches is 15 and the input value s1 is out of the coordinate range, 16 or over, the number of binary searches will be equal to the number of sequential numbers or smaller. Therefore, the data processing speed with the binary search will rise.



Operation Errors In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The X coordinates of the scaling conversion data positioned before the point specified by s1 are not set in ascending order. (However, this error is not detected when SM750 is on.) (Error code 4100)
- The input value specified by s1 is out of the range of the scaling conversion data set. (Error code 4100)
- The number of X and Y coordinates of the device specified by s2 is out of the range from 1 to 32767. (Error code 4100)

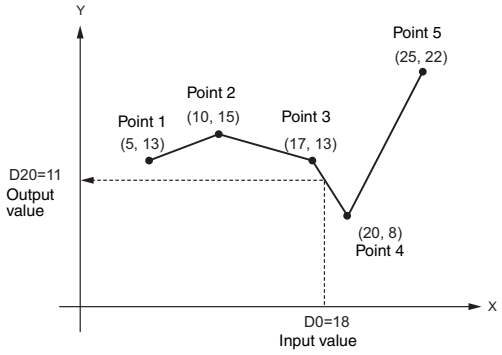
Program Example

SCLP

The following program executes scaling for the scaling conversion data of which the devices specified at D100 and up are set with the input value specified at D0. The result is stored in D20.

MELSEC Instruction List	Ladder Diagram
IEC Instruction List	

Setting Item		Device	Setting Contents
Number of coordinate points		D100	K5
Point 1	X coordinate	D101	K5
	Y coordinate	D102	K13
Point 2	X coordinate	D103	K10
	Y coordinate	D104	K15
Point 3	X coordinate	D105	K17
	Y coordinate	D106	K13
Point 4	X coordinate	D107	K20
	Y coordinate	D108	K8
Point 5	X coordinate	D109	K25
	Y coordinate	D110	K22



7.13.5 SCL2, SCL2P, DSCL2, DSCL2P

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

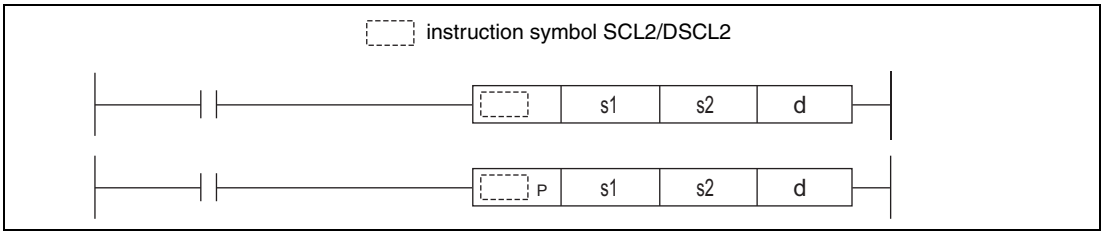
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
s1	—	●	●	●	●	●	●	●	—
s2	—	●	●	—	—	—	—	—	—
d	—	●	●	●	●	●	●	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s1	Input values for scaling or first number of the device where input values are stored	BIN 16/32-bit
s2	First number of the devices where scaling conversion data are stored	
d	First number of of the devices where output values depending on scaling are stored	

Functions

Scaling (Point-by-point coordinate data)

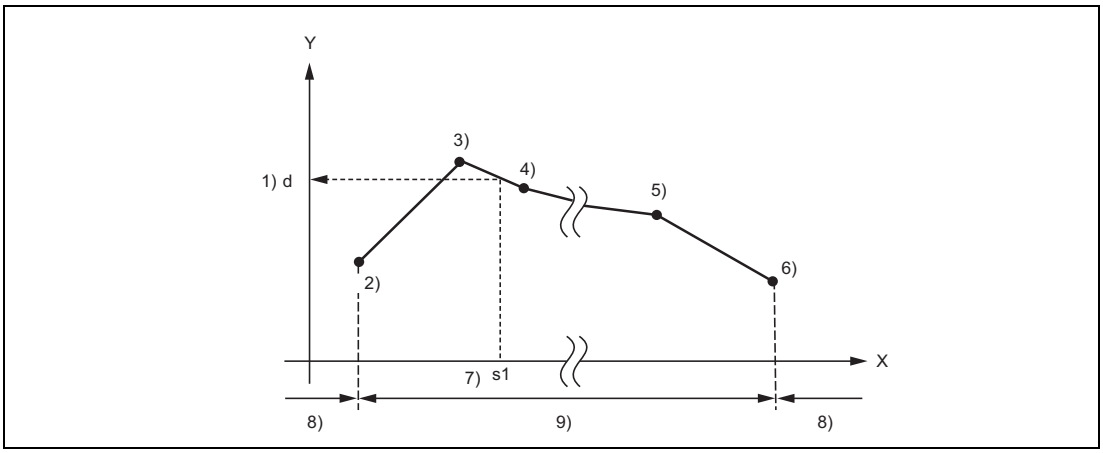
SCL2 Scaling of BIN 16-bit data

This instruction executes scaling for the scaling conversion data (16-bit data units) specified by s2 with the input value specified by s1, and then stores the operation result into the devices specified by d.

The scaling conversion is executed based on the scaling conversion data stored in the device specified by s2 and up (see following table).

Setting Item		Device Assignment
Number of coordinate points		s2
X coordinate	Point 1	(s2)+1
	Point 2	(s2)+2
	...	...
	Point n [1)]	(s2)+n
Y coordinate	Point 1	(s2)+(n+1)
	Point 2	(s2)+(n+2)
	...	...
	Point n [1)]	(s2)+2n

[1] n indicates the number of coordinates specified by s2.



[1] Output value (d)
[2] Point 1
[3] Point 2
[4] Point 3
[5] Point n–1
[6] Point n
[7] Input value (s1)
[8] Operation error
[9] Operable range

If the value does not result in an integer, this instruction rounds the value to the whole number.

Set the X coordinate of the scaling conversion data in ascending order.

Set the input value s1 within the range of the scaling conversion data (within the range of s2 devices).

If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.

Specify the number of coordinate points of scaling conversion data from 1 to 32767.

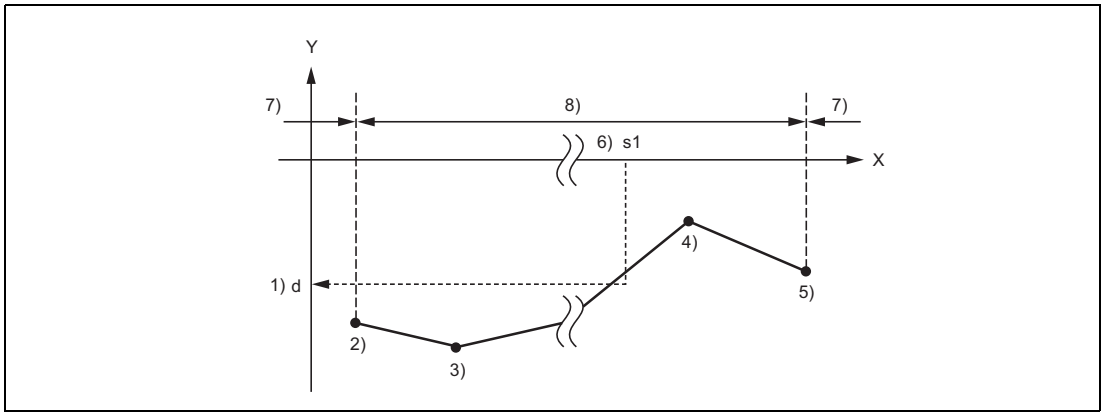
DSCL2 Scaling of BIN 32-bit data

This instruction executes scaling for the scaling conversion data (32-bit data units) specified by s2 with the input value specified by s1, and then stores the operation result into the devices specified by d.

The scaling conversion is executed based on the scaling conversion data stored in the device specified by s2 and up (see following table).

Setting Item		Device Assignment
Number of coordinate points		(s2)+1, (s2)
X coordinate	Point 1	(s2)+3, (s2)+2
	Point 2	(s2)+5, (s2)+4
	...	...
	Point n [1)]	(s2)+(2n+1), (s2)+(2n)
Y coordinate	Point 1	(s2)+(2n+3), (s2)+(2n+2)
	Point 2	(s2)+(2n+5), (s2)+(2n+4)
	...	...
	Point n [1)]	(s2)+(4n+1), (s2)+4n

[1] n indicates the number of coordinates specified by s2



[1] Output value (d)
[2] Point 1
[3] Point 2
[4] Point n–1
[5] Point n
[6] Input value (s1)
[7] Operation error
[8] Operable range

If the value does not result in an integer, this instruction rounds the value to the whole number.

Set the X coordinate of the scaling conversion data in ascending order.

Set the input value s1 within the range of the scaling conversion data (within the range of s2 and ((s2)+1) devices).

If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.

Specify the number of coordinate points of scaling conversion data from 1 to 32767.

NOTE

When the scaling conversion data are set in ascending order, the searching methods change from one to the other depending on the SM750 status. Therefore, the processing speed also changes. The number of searches determines the processing speed. Fewer number of serches make the processing run faster.
For details refer to section 7.13.4.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The X coordinates are not set in ascending order.
  (Error code 4100)
- The input value specified by s1 is out of the range of the scaling conversion data set.
  (Error code 4100)
- The number of X and Y coordinates of the device specified by s2 is out of the range from 1 to 32767.
  (Error code 4100)

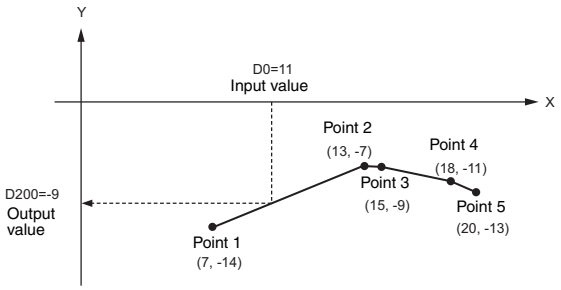
Program Example

SCL2P

The following program executes scaling for the scaling conversion data of which the devices specified at D100 and up are set with the input value specified at D0. The result is stored in D20.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Setting Item		Device	Setting Contents
Number of coordinate points		D110	K5
X coordinate	Point 1	D111	K7
	Point 2	D112	K13
	Point 3	D113	K15
	Point 4	D114	K18
	Point 5	D115	K20
Y coordinate	Point 1	D116	K-14
	Point 2	D117	K-7
	Point 3	D118	K-15
	Point 4	D119	K-11
	Point 5	D120	K-18



7.14 File register switching instructions

The switching instructions enable switching between file register blocks and between file names in file registers.

The table below gives an overview of the instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Setting file register blocks	RSET	RSET_MD
		RSET_K_MD
	RSETP	RSET_P_MD
		RSET_K_P_MD
Setting file register files	QDRSET	QDRSET_M
	QDRSETP	QDRSET_P_MD
Setting comment files	QCDSET	QCDSET_M
	QCDSET	QCDSET_P_MD

7.14.1 RSET, RSETP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●[1)]	●

[1] Universal model QCPU: Other than Q00UJCPU

Devices

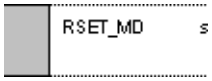
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	●	●	—

GX IEC Developer

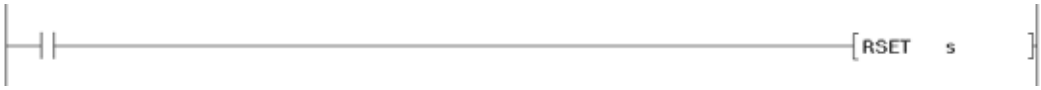
MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2



Variables

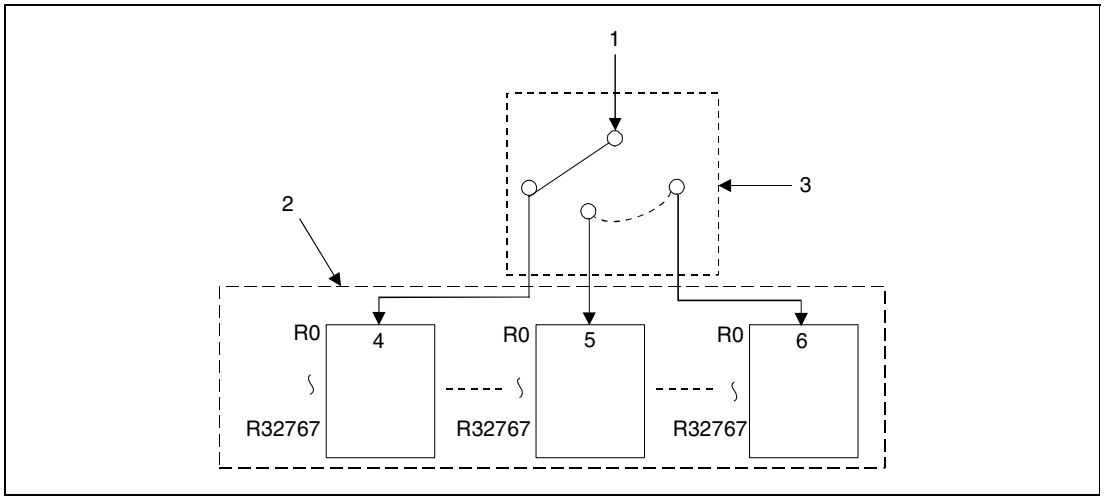
Set Data	Meaning	Data Type
s	Number of file register block or first number of device storing this number	BIN 16-bit

Functions

Setting file register blocks

RSET Switch instruction for file register blocks

The RSET instruction switches from a file register block being in use by a program to a file register block with the number specified by s. After switching over, the sequence program exclusively accesses file registers (R0–R32767) in the specified block.



[1] Processing with file register access

[2] File used by program

[3] Number of file register block (s)

[4] Block 0

[5] Block 1

[6] Block n

NOTE *When a file register (R) is refreshed and the block No. of the file register is switched with the RSET instruction, follow restrictions. For restrictions on file registers, refer to section 3.13.*

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The block number specified by s does not exist. (Error code 4100)
- There are no file registers in the block specified by s. (Error code 4101)

Program Example

RSETP

The following program compares the file register R0 in register block 0 to the file register R0 in register block 1. The file register blocks 0 and 1 are addressed via the RSET instruction. Both file registers R0 are read via the MOV instruction.

If the value in R0 (block 0) is equal to the value in R0 (block 1), the output Y40 is set.

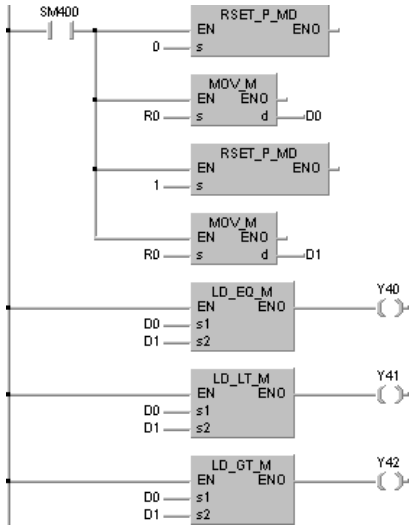
If the value in R0 (block 0) is less than the value in R0 (block 1), the output Y41 is set.

If the value in R0 (block 0) is greater than the value in R0 (block 1), the output Y42 is set.

MELSEC Instruction List

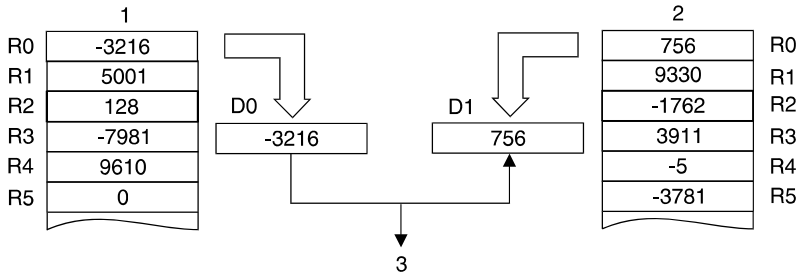
```
MELSEC   LD      SM400
         RSETP   K0
         MOV     R0
                 D0
         RSETP   K1
         MOV     R0
                 D1
         LD=     D0
                 D1
         OUT     Y40
         LD<     D0
                 D1
         OUT     Y41
         LD>     D0
                 D1
         OUT     Y42
```

Ladder Diagram



IEC Instruction List

```
         LD      SM400
         RSET_P_MD  0
         MOV_M   R0 , D0
         RSET_P_MD  1
         MOV_M   R0 , D1
         LD      D0
         EQ      D1
         ST      Y40
         LD      D0
         LT      D1
         ST      Y41
         LD      D0
         LT      D1
         ST      Y42
         LD_GT_M D0 , D1
         ST      Y42
```



[1] Block 0

[2] Block 1

[3] Y41 is set because D0 is less than D1.

7.14.2 QDRSET, QDRSETP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●[1)]	

[1] Universal model QCPU: Other than Q00UJCPU

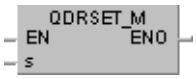
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—

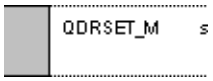
GX IEC Developer

MELSEC Instruction List

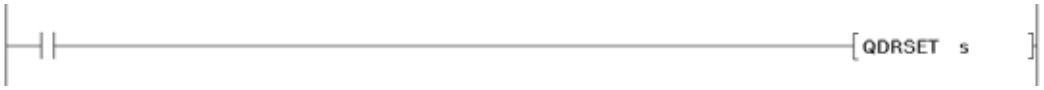
Ladder Diagram



IEC Instruction List



GX Works2



Variables

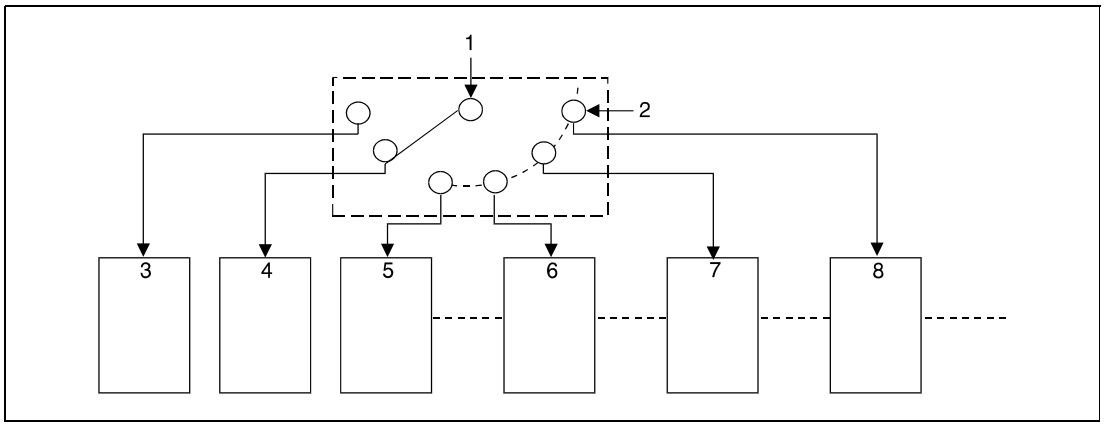
Set Data	Meaning	Data Type
s	Drive number and file name of file register file to be switched to or first number of device storing such data	Character string

Functions

Setting file register files

QDRSET Switch instruction for file register files

The QDRSET instruction switches from a file register file being in use by a program to a file register file specified by s. After switching over, the sequence program exclusively accesses file registers (R0–R32767) in block 0 of the specified file register file. The file register blocks are selected via the RSET instruction.



[1] Processing with file register access
[2] Setting the drive and file(s)
[3] Drive 1, file A
[4] Drive 1, file B
[5] Drive 1, file C
[6] Drive 2, file A
[7] Drive 3, file A
[8] Drive 4, file A

In total, 4 drives can be assigned (1–4). The drive number 0 cannot be assigned; this range is reserved for internal memory. Note that available drives vary depending on the CPU module used. Refer to the manual of the CPU module and check the drives that can be specified.

The extension .QDR is not needed to be entered for file specification.

A file name setting can be cleared by specifying the NULL character (00H) for the file name.

File register files selected by the QDRSET instruction are given priority even if a drive number and file name were specified by the parameters.

NOTES

If the file name is changed with the QDRSET instruction, the file name returns to the name specified by the parameter when the CPU module is switched from STOP to RUN. To maintain the file name even after the CPU mode is changed from STOP to RUN, execute the QDRSET instruction with the SM402 special relay, which turns ON during one scan when the CPU enters from STOP to RUN mode.

For refreshing a file register, do not change the file name of the file register with the QDRSET instruction. For restrictions on file registers, refer to section 3.13.

Operation Errors

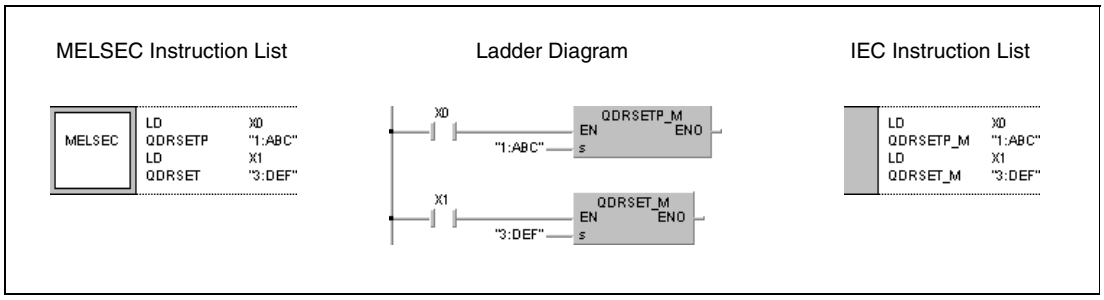
In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The file register file does not exist on the drive specified by s.
  (Error code 2410)

Program Example

QDRSET/QDRSETP

With leading edge from X0, the following program switches to the file register file ABC.QDR on drive 1. While X1 is set, the file register file DEF.QDR on drive 3 is accessed.



7.14.3 QCDSET, QCDSETP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	

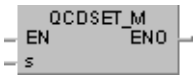
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—

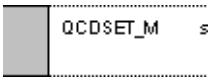
GX IEC Developer

MELSEC Instruction List

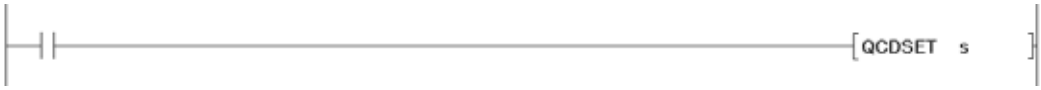
Ladder Diagram



IEC Instruction List



GX Works2



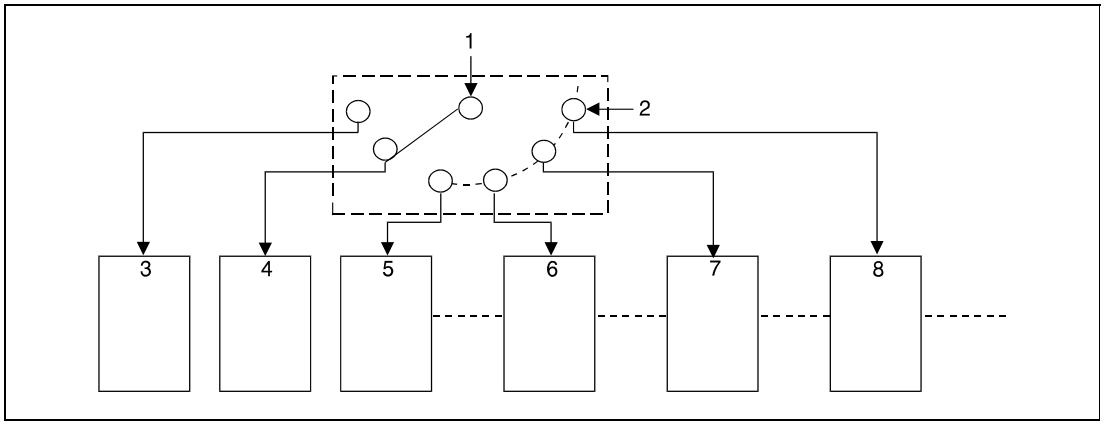
Variables

Set Data	Meaning	Data Type
s	Drive number and file name of comment file to be switched to or first number of device storing such data	Character string

Functions **Setting comment files**

QCDSET Switch instruction for comment files

The QCDSET instruction switches from a comment file being in use by a program to a comment file specified by s. After switching over, the sequence program exclusively accesses comment data of the specified comment file.



[1] Processing with comment data access

[2] Setting the drive and comment file (s)

[3] Drive 1, file A

[4] Drive 1, file B

[5] Drive 1, file C

[6] Drive 2, file A

[7] Drive 3, file A

[8] Drive 4, file A

In total, 4 drives can be assigned (1–4). The drive number 0 cannot be assigned; this range is reserved for internal memory. Note that available drives vary depending on the CPU module used. Refer to the manual of the CPU module and check the drives that can be specified.

The extension .QCD is not needed to be entered for file specification.

A file name setting can be cleared by specifying the NULL character (00H) for the file name.

Comment files selected by the QCDSET instruction are given priority even if a drive number and file name were specified by the parameters.

NOTE *If the file name is changed with the QCDSET instruction, the file name returns to the name specified by the parameter when the CPU module is switched from STOP to RUN.*
To maintain the file name even after the CPU mode is changed from STOP to RUN, execute the QCDSET instruction with the SM402 special relay, which turns ON during one scan when the CPU enters from STOP to RUN mode.

NOTES

This instruction will not be executed even when the execution command of this instruction is ON while SM721 (file access in execution) is ON for the Universal model QCPU and LCPU. Execute this instruction when SM721 is OFF.

For the LCPU, when drive 2 (SD memory card) is specified as the drive number, this instruction cannot be executed while SM606 (SD memory card forced disable instruction) is ON. Even if the instruction is attempted to be executed, the command will be ignored.

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

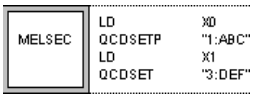
- The comment file does not exist on the drive specified by s.
  (Error code 2410)

Program Example

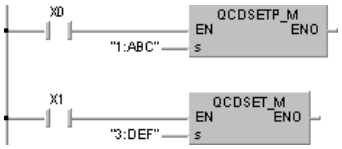
QCDSET/QCDSETP

With leading edge from X0, the following program switches to the comment file ABC.QCD on drive 1. While X1 is set, the comment file DEF.QCD on drive 3 is accessed.

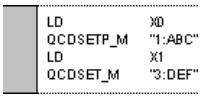
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



7.15 Clock instructions

The clock instructions read and write, add and subtract, change and compare the data format of clock data of the internal CPU clock.

The table below gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Reading clock data	DATERD	DATERD_MD
	DATERDP	DATERD_P_MD
Writing clock data	DATEWR	DATEWR_MD
	DATEWRP	DATEWR_P_MD
Adding clock data	DATE+	DATEPLUS_M
	DATE+P	DATEPLUSP_M
Subtracting clock data	DATE-	DATEMINUS_M
	DATE-P	DATEMINUSP_M
Changing clock data format from hh:mm:ss to seconds	SECOND	SECOND_M
	SECONDP	SECONDP_M
Changing clock data format from seconds to hh:mm:ss	HOUR	HOUR_M
	HOURP	HOURP_M
Date comparison	LDDT=	
	ANDDT=	
	ORDT=	
	LDDT<>	
	ANDDT<>	
	ORDT<>	
	LDDT>	
	ANDDT>	
	ORDT>	
	LDDT<=	
	ANDDT<=	
	ORDT<=	
	LDDT<	
	ANDDT<	
	ORDT<	
	LDDT>=	
	ANDDT>=	
	ORDT>=	

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Clock comparison	LDTM=	
	ANDTM=	
	ORTM=	
	LDTM<>	
	ANDTM<>	
	ORTM<>	
	LDTM>	
	ANDTM>	
	ORTM>	
	LDTM<=	
	ANDTM<=	
	ORTM<=	
	LDTM<	
	ANDTM<	
	ORTM<	
	LDTM>=	
	ANDTM>=	
	ORTM>=	

NOTE *The expansion clock instructions described in section 7.16 can process the milliseconds of the internal CPU clock as well.*

7.15.1 DATERD, DATERDP

CPU

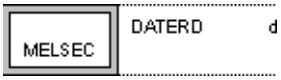
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

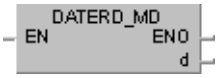
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

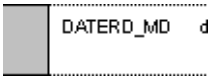
MELSEC Instruction List



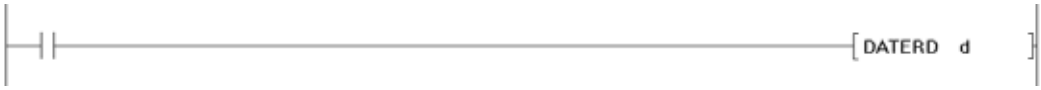
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
d	First number of device storing clock data being read	BIN 16-bit	Array [0..6] of ANY16

Functions

Reading clock data

DATERD Read instruction

The DATERD instruction reads year, month, day, hour, minute, second, and weekday from the clock element of the CPU module and stores the clock data in binary format in the devices specified by d+0 (Array_d[0]) through d+6 (Array_d[6]). The assignment of registers to clock data is illustrated below:

d+0, Array_d[0] = year (1)

d+1, Array_d[1] = month (January = 1, December = 12) (2)

d+2, Array_d[2] = day (3)

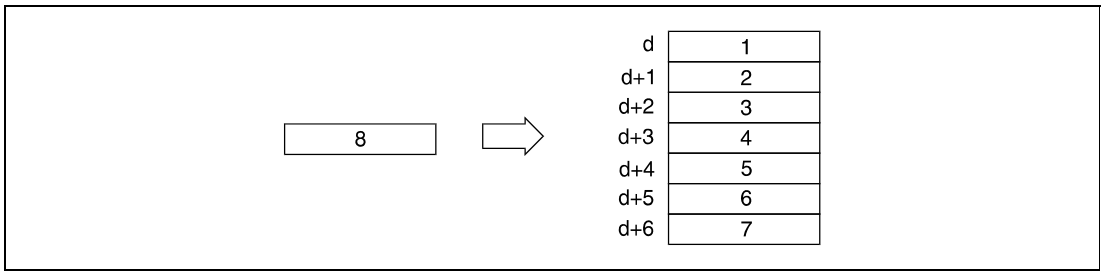
d+3, Array_d[3] = hour (24 hour format) (4)

d+4, Array_d[4] = minute (5)

d+5, Array_d[5] = second (6)

d+6, Array_d[6] = day of the week (7)

The clock element is indicated as 8.



The following table contains the value range of clock data in d+0 (Array_d[0]) through d+6 (Array_d[6]):

Clock data	Year	Month	Day	Hour	Minute	Second	Day of the week
Input range	1980–2079	1–12	1–31	0–23	0–59	0–59	0–6
Devices	d+0 (Array_d[0])	d+1 (Array_d[1])	d+2 (Array_d[2])	d+3 (Array_d[3])	d+4 (Array_d[4])	d+5 (Array_d[5])	d+6 (Array_d[6])

The "year" is stored as four-digit indication.

The day of the week stored in d+6 (Array_d[6]) is indicated from 0 to 6. The table below shows the assignment of weekdays:

Weekday	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Storage value	0	1	2	3	4	5	6

Leap years are calculated automatically by the CPU clock.

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

DATERD

The following program reads clock data from the internal CPU clock and outputs it in BCD format at the outputs as follows:

Y70 - Y7F = year Y68 - Y6F = month

Y60 - Y67 = day Y58 - Y5F = hour

Y50 - Y57 = minute Y48 - Y4F = second

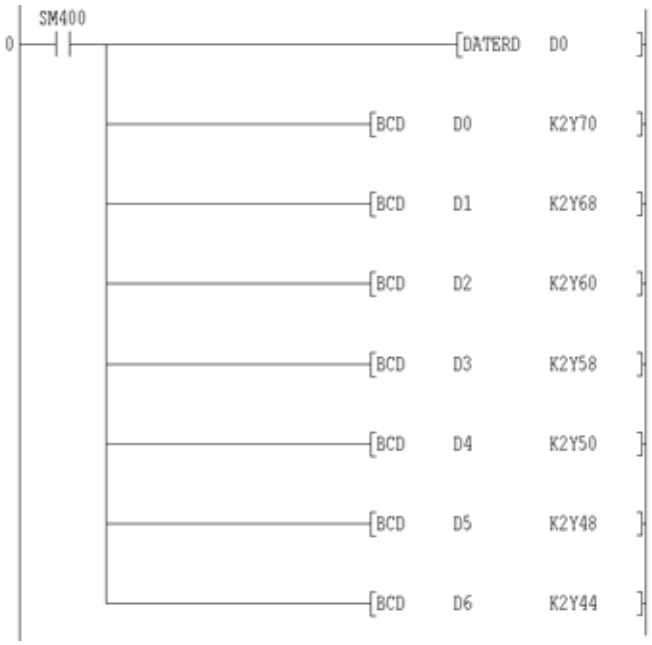
Y44 - Y47 = day of the week

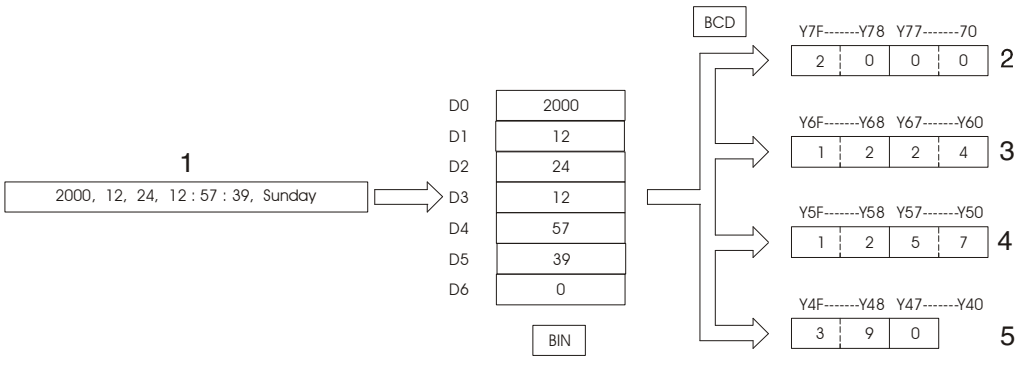
Clock data	Year	Month	Day	Hour	Minute	Second	Day of the week
Devices	D0	D1	D2	D3	D4	D5	D6

Instruction List

```
 0   LD      SM400
 1   DATERD  D0
 3   BCD     D0      K2Y70
 6   BCD     D1      K2Y68
 9   BCD     D2      K2Y60
12   BCD     D3      K2Y58
15   BCD     D4      K2Y50
18   BCD     D5      K2Y48
21   BCD     D6      K2Y44
24   END
```

Ladder Diagram





[1] Clock data

[2] Year

[3] Month, day

[4] Hour, minute

[5] Second, day of the week

7.15.2 DATEWR, DATEWRP

CPU

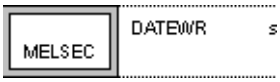
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

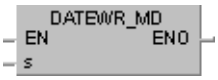
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DY
s	—	●	●	—	—	—	—	—	—

GX IEC Developer

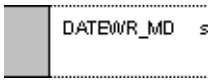
MELSEC Instruction List



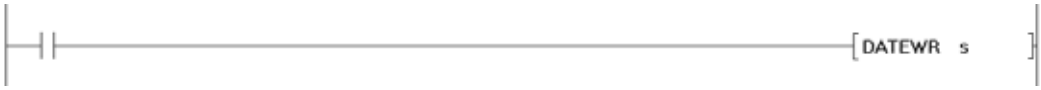
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	First number of device storing the data to be written to the internal CPU clock	BIN 16-bit	Array [0..6] of ANY16

Functions **Writing clock data**

DATEWR Write instruction

The DATEWR instruction writes clock data of year, month, day, hour, minute, second, and weekday stored in the devices specified by s+0 (Array_s[0]) through s+6 (Array_s[6]) to the internal CPU clock. The clock data are stored in binary format. The assignment of registers to clock data is illustrated below:

s+0, Array_s[0] = year (1)

s+1, Array_s[1] = month (January = 1, December = 12) (2)

s+2, Array_s[2] = day (3)

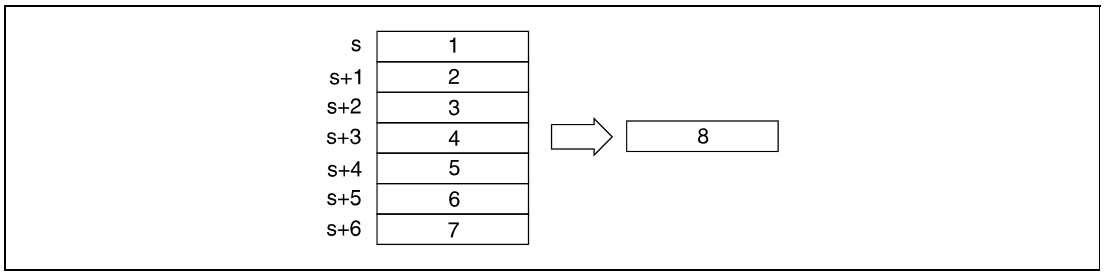
s+3, Array_s[3] = hour (24 hour format, 0 to 23 hours) (4)

s+4, Array_s[4] = minute (5)

s+5, Array_s[5] = second (6)

s+6, Array_s[6] = day of the week (7)

The clock element is indicated as 8.



The following table contains the value range of clock data in s+0 (Array_s[0]) through s+6 (Array_s[6]):

Clock data	Year	Month	Day	Hour	Minute	Second	Day of the week
Input range	1980–2079	1–12	1–31	0–23	0–59	0–59	0–6
Devices	s+0 (Array_s[0])	s+1 (Array_s[1])	s+2 (Array_s[2])	s+3 (Array_s[3])	s+4 (Array_s[4])	s+5 (Array_s[5])	s+6 (Array_s[6])

The "year" is designated as four-digit indication.

The weekday stored in s+6 (Array_s[6]) is indicated from 0 to 6. The table below shows the assignment of weekdays:

Weekday	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Storage value	0	1	2	3	4	5	6

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The clock data specified in s+0 (Array_s[0]) through s+6 (Array_s[6]) exceed the relevant value range.
  (Error code 4100)
- The device specified by s exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

DATEWRP

With leading edge from X40, the following program writes the clock data in binary format at the inputs to the internal CPU clock. The inputs are assigned to the clock data as follows:

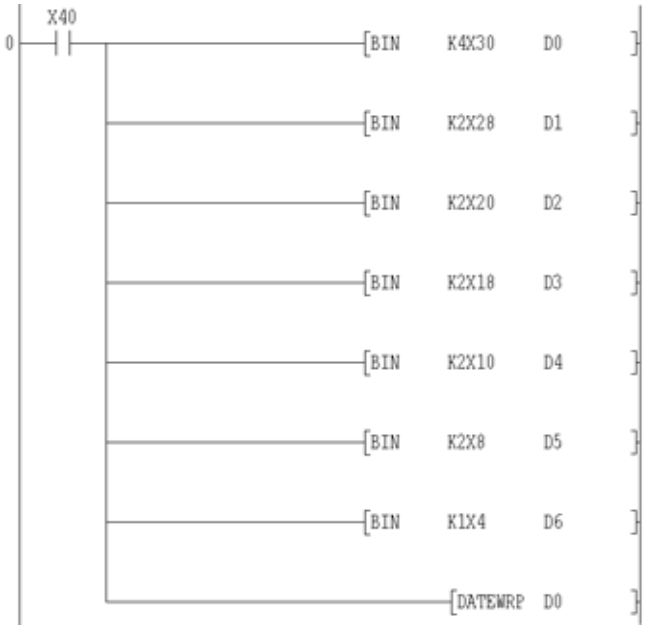
X30 - X3F = year
X28 - X2F = month
X20 - X27 = day
X18 - X1F = hour
X10 - X17= minute
X8 - XF = second

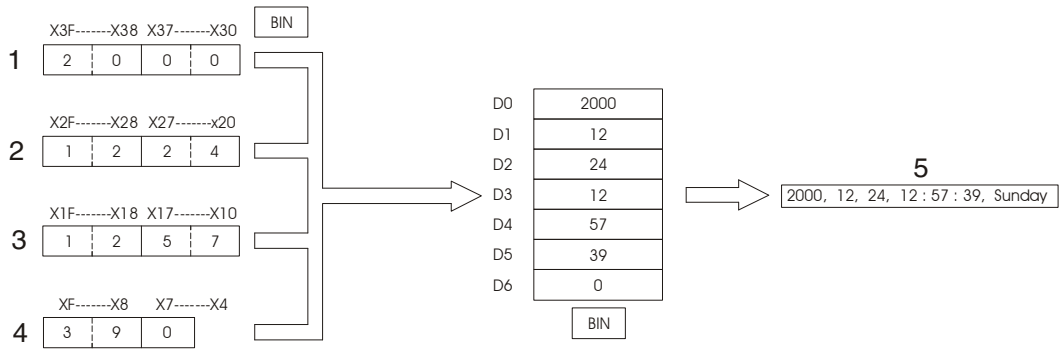
Clock data	Year	Month	Day	Hour	Minute	Second	Day of the week
Devices	D0	D1	D2	D3	D4	D5	D6

Instruction List

```
 0  LD       X40
 1  BIN      K4X30    D0
 4  BIN      K2X28    D1
 7  BIN      K2X20    D2
10  BIN      K2X18    D3
13  BIN      K1X10    D4
16  BIN      K2X8     D5
19  BIN      K1X4     D6
22  DATEWRP  D0
24  END
```

Ladder Diagram





[1] Year
[2] Month, day
[3] Hour, minute
[4] Second, day of the week
[5] Clock data

7.15.3 DATE+, DATE+P

CPU

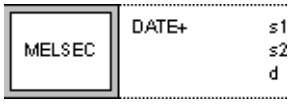
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

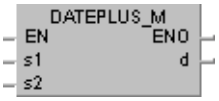
	UsableDevices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

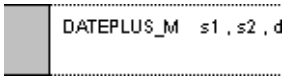
MELSEC Instruction List



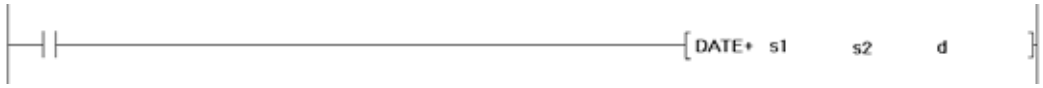
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s1	Clock data to be added to	BIN 16-bit	Array [0..2] of ANY16
s2	Clock data to be added		
d	First number of device storing the clock data of the operation result		

Functions

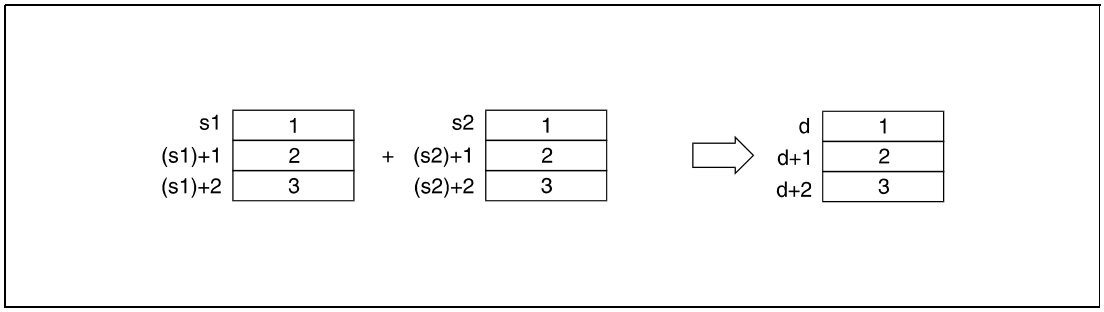
Adding clock data

DATE+ Addition instruction

The DATE+ instruction adds the clock data stored in the devices specified from s2 on to the clock data stored in the devices specified from s1 on. The clock data of the operation result is stored in the devices specified from d.

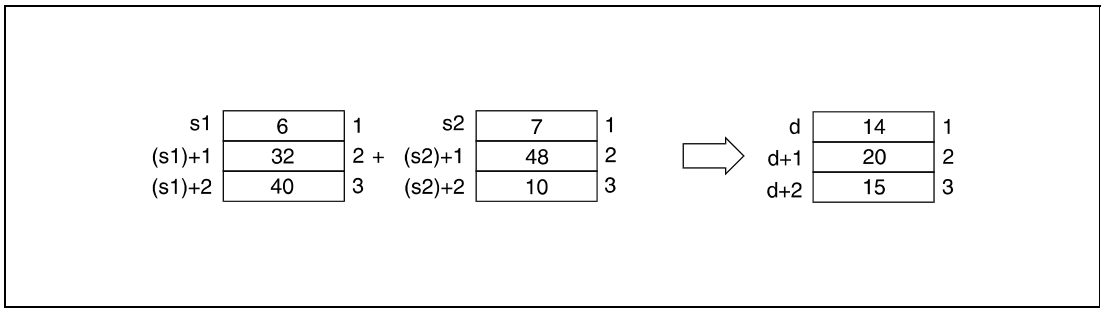
The following table contains the value range of clock data in (s1)+0 through (s1)+2 (Array_s1[0] through Array_s1[2]), (s2)+0 through (s2)+2 (Array_s2[0] through Array_s2[2]), and d+0 through d+2 (Array_d[0] through Array_d[2]):

Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week
Input range	—	—	—	0–23	0–59	0–59	—
Devices	—	—	—	s1+0 (Array_s1[0])	s1+1 (Array_s1[1])	s1+2 (Array_s1[2])	—
Devices	—	—	—	s2+0 (Array_s2[0])	s2+1 (Array_s2[1])	s2+2 (Array_s2[2])	—
Devices	—	—	—	d+0 (Array_d[0])	d+1 (Array_d[1])	d+2 (Array_d[2])	—



[1] Hour
[2] Minute
[3] Second

In the following diagram to the clock data
6 hours, 32 minutes, 40 seconds ((s1)+0 through (s1)+2, (Array_s1[0] through Array_s1[2])) the clock data
7 hours, 48 minutes, 10 seconds ((s2)+0 through (s2)+2, (Array_s2[0] through Array_s2[2])) is added. The result
14 hours, 20 minutes, 50 seconds is stored in d+0 through d+2 (Array_d[0] through Array_d[2]).

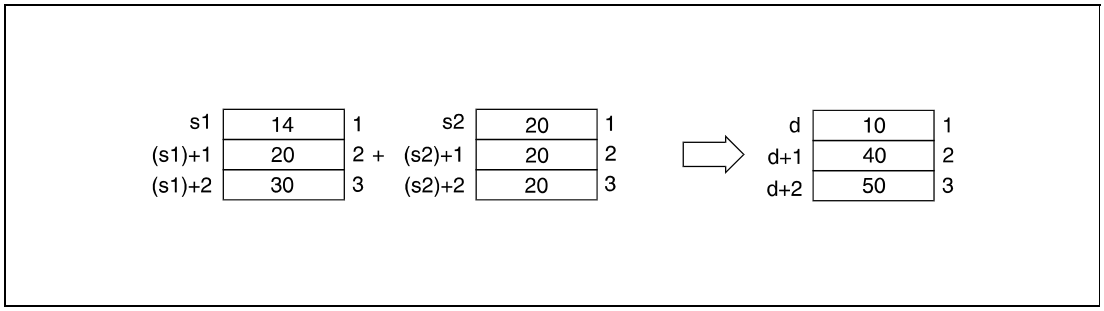


[1] Hour
[2] Minute
[3] Second

If the addition result of clock data exceeds 24 hours, 24 hours are subtracted automatically to achieve a correct time value.

The following diagram illustrates the addition of
14 hours, 20 minutes, and 30 seconds
to 20 hours, 20 minutes, and 20 seconds.
The result would be 34 hours, 40 minutes, and 50 seconds.

Since this result is not a correct time format, after the subtraction of 24 hours, the correct result is 10 hours, 40 minutes, and 50 seconds (10:40:50 the next day).



[1] Hour

[2] Minute

[3] Second

NOTE *Refer to section 7.15.2 "DATEWR, DATEWRP" for further information on that topic.*

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The clock data in (s1)+0 through (s1)+2 (Array_s1[0] through Array_s1[2]) and (s2)+0 through (s2)+2 (Array_s2[0] through Array_s2[2]) exceed the input range.
  (Error code 4100)
- The device specified by s1, s2 or d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

DATE+P

With leading edge from X20, the following program reads the clock data from the internal CPU clock via the DATERDP instruction and stores it in the registers D0 through D6 (see first diagram after the program example below).

The DATE+P instruction adds one hour (D10, D11, D12) to the read data. The result is stored in D100 through D102 (see second diagram after the program example below).

Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week
Devices	D0 (var_D0[0])	D1 (var_D0[1])	D2 (var_D0[2])	D3 (var_D0[3])	D4 (var_D0[4])	D5 (var_D0[5])	D6 (var_D0[6])
Devices	—	—	—	D20 (var_D20[0])	D21 (var_D20[1])	D22 (var_D20[2])	—
Devices	—	—	—	D10 (var_D10[0])	D11 (var_D10[1])	D12 (var_D10[2])	—
Devices	—	—	—	D100 (var_D100[0])	D101 (var_D100[1])	D102 (var_D100[2])	—

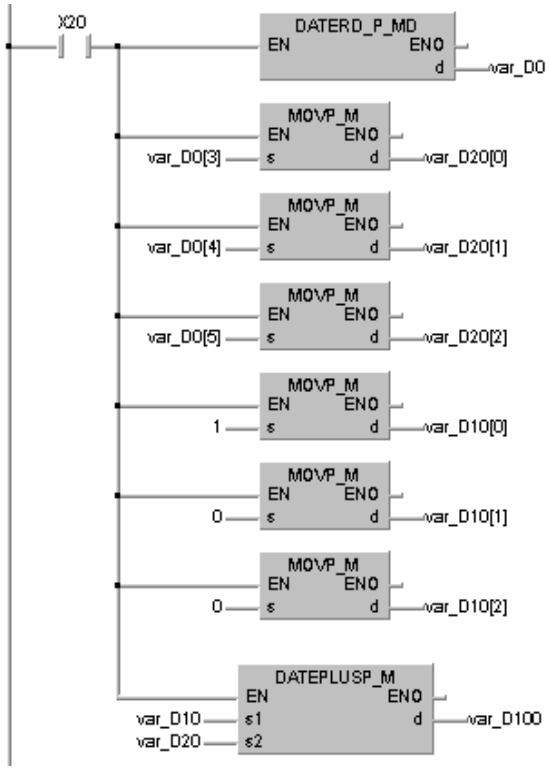
MELSEC Instruction List

```
MELSEC  LD      X20
        DATERDP D0
        MOVP    K1
                D10
        DMOVP   K0
                D11
        DATE+P  D3
                D10
                D100
```

IEC Instruction List

```
LD              X20
DATERD_MD       var_D0
MOVP_M          var_D0[3] , var_D20[0]
MOVP_M          var_D0[4] , var_D20[1]
MOVP_M          var_D0[5] , var_D20[2]
MOVP_M          1 , var_D10[0]
MOVP_M          0 , var_D10[1]
MOVP_M          0 , var_D10[2]

DATEPLUSP_M     var_D10 , var_D10 , var_D100
```

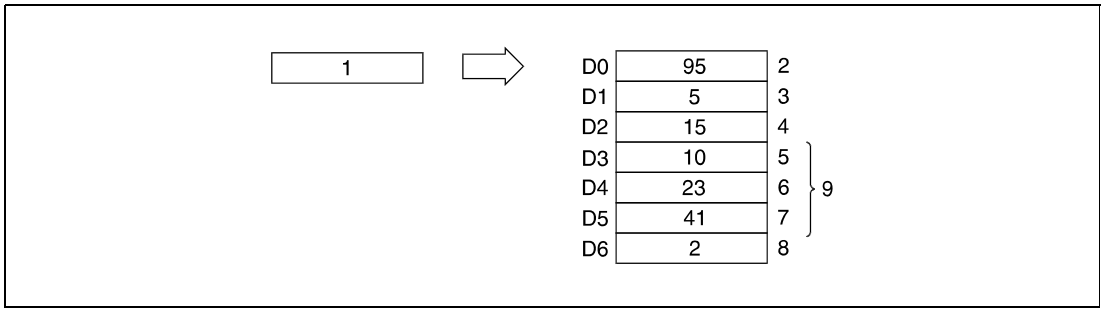
Ladder Diagram



NOTE

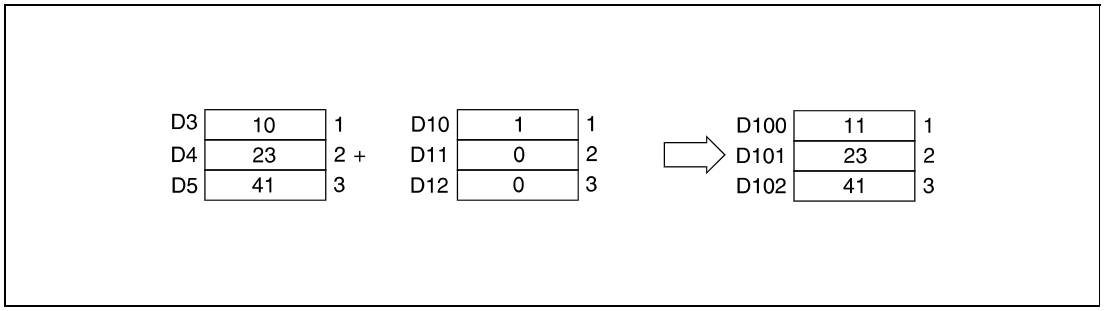
This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

The diagram below illustrates reading clock data via the DATERDP instruction.



[1] Clock element
[2] Year
[3] Month (January = 1, December = 12)
[4] Day
[5] Hour (24-hour format)
[6] Minute
[7] Second
[8] Day of the week
[9] Clock data

The diagram below illustrates the addition via the DATE+P instruction.



[1] Hour
[2] Minute
[3] Second

7.15.4 DATE-, DATE-P

CPU

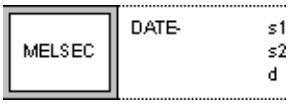
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

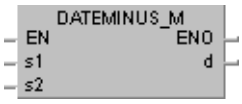
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				DY
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

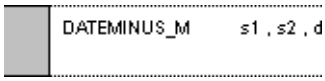
MELSEC Instruction List



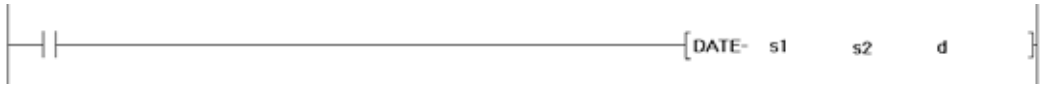
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s1	First number of device storing clock data to be subtracted from	BIN 16-bit
s2	First number of device storing clock data to be subtracted	
d	First number of device storing the clock data of the subtraction result	

Functions

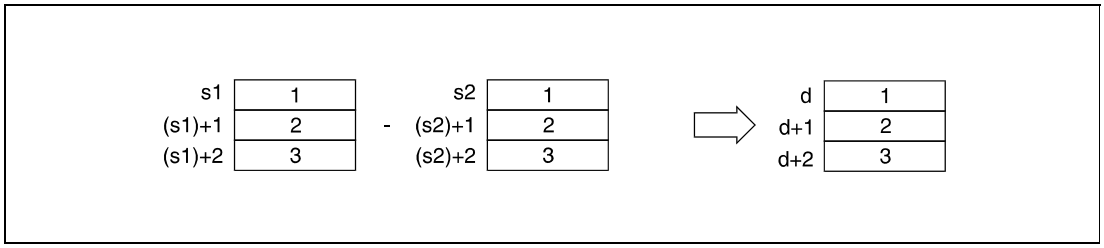
Subtracting clock data

DATE- Subtraction instruction

The DATE instruction subtracts clock data stored in the device specified from s2 on from the clock data in the device specified from s1 on. The clock data of the operation result is stored in the device specified from d on.

The following table shows the input ranges of clock data stored in (s1)+0 through (s1)+2 (Array_s1[0] through Array_s1[2]), (s2)+0 through (s2)+2 (Array_s2[0] through Array_s2[2]) and d+0 through d+2 (Array_d[0] through Array_d[2]).

Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week
Input range	—	—	—	0–23	0–59	0–59	—
Devices	—	—	—	s1+0 (Array_s1[0])	s1+1 (Array_s1[1])	s1+2 (Array_s1[2])	—
Devices	—	—	—	s2+0 (Array_s2[0])	s2+1 (Array_s2[1])	s2+2 (Array_s2[2])	—
Devices	—	—	—	d+0 (Array_d[0])	d+1 (Array_d[1])	d+2 (Array_d[2])	—

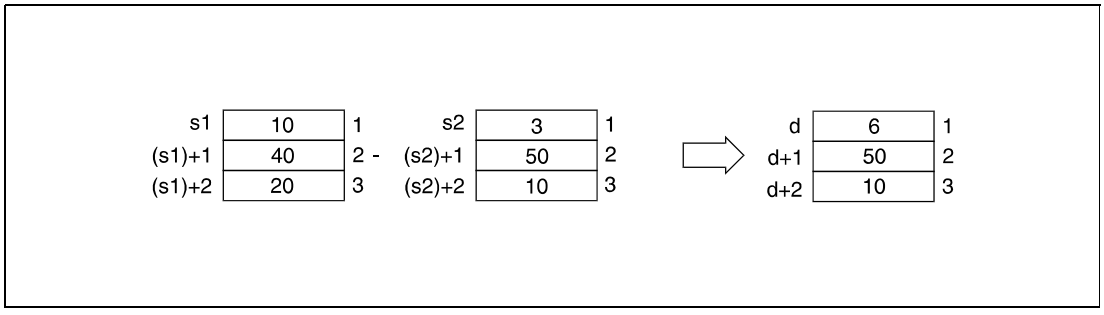


[1] Hour

[2] Minute

[3] Second

The following diagram illustrates the subtraction of 3 hours, 50 minutes, and 10 seconds ((s2)+0 through (s2)+2, (Array_s2[0] through Array_s2[2])) from 10 hours, 40 minutes, and 20 ((s1)+0 through (s1)+2, (Array_s1[0] through Array_s1[2])). The result, 6 hours, 50 minutes, and 10 seconds is stored in d+0 through d+2 (Array_d[0] through Array_d[2]).



[1] Hour

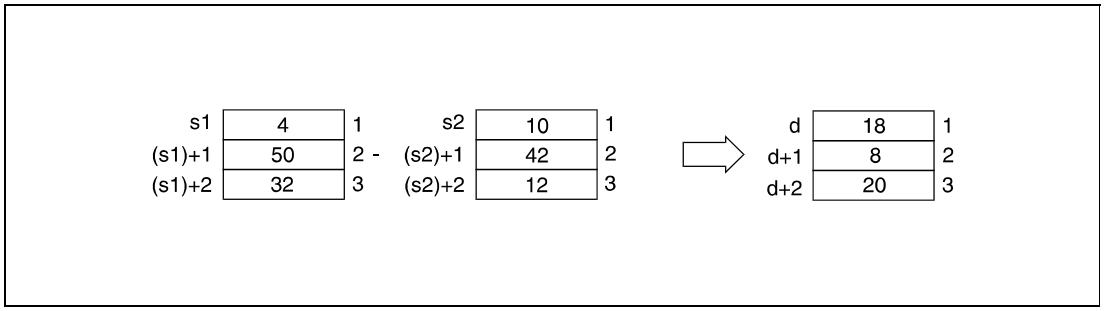
[2] Minute

[3] Second

If the subtraction result of clock data becomes negative, 24 hours are added automatically to achieve a correct time value.

The following diagram illustrates the subtraction of
10 hours, 42 minutes, and 12 seconds
from 4 hours, 50 minutes, and 32 seconds.
The result would be -6 hours, 8 minutes, and 20 seconds.

Since this result is not a correct time format, after the addition of 24 hours, the correct result is 18 hours, 8 minutes, and 20 seconds (18:08:20 the day before).



[1] Hour
[2] Minute
[3] Second

NOTE *Refer to section 7.15.2 "DATEWR, DATEWRP" for further information on that topic.*

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The clock data in (s1)+0 through (s1)+2 (Array_s1[0] through Array_s1[2]) and (s2)+0 through (s2)+2 (Array_s2[0] through Array_s2[2]) exceed the input range.
  (Error code 4100)
- The device specified by s1 or s2 or d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

DATE-P

With leading edge from X1C, the following program reads the clock data from the internal CPU clock via the DATERDP instruction and stores it in the registers D100 through D106 (see first diagram after the program example below).

The DATE-P instruction subtracts 10 hours (D10), 40 minutes (D11) and 10 seconds (D12) from the read data. To the negative subtraction result, -8 hours, 41 minutes and 10 seconds is added 24 hours automatically. The correct result, 16 hours, 41 minutes and 10 seconds (16:41:10 the day before) is stored in R10 through R12 (see second diagram after the program example below).

Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week
Devices	D100 (var_D100[0])	D101 (var_D100[1])	D102 (var_D100[2])	D103 (var_D100[3])	D104 (var_D100[4])	D105 (var_D100[5])	D106 (var_D100[6])
Devices	—	—	—	D1000 (var_D1000[0])	D1001 (var_D1000[1])	D1002 (var_D1000[2])	—
Devices	—	—	—	D10 (var_D10[0])	D11 (var_D10[1])	D12 (var_D10[2])	—
Devices	—	—	—	R10 (var_R10[0])	R11 (var_R10[1])	R12 (var_R10[2])	—

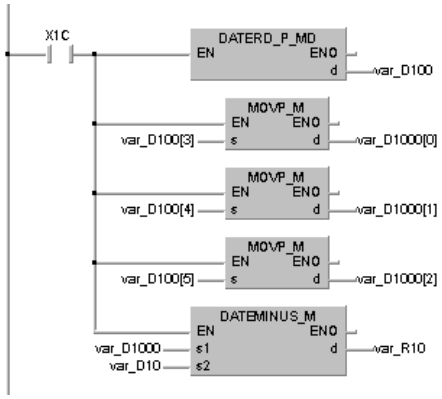
MELSEC Instruction List

```
MELSEC   LD        X1C
         DATERDP   D100
         DATE-P    D1000
                   D10
                   R10
```

IEC Instruction List

```
LD              X1C
DATERD_P_MD     var_D100
MOVP_M          var_D100[3] , var_D1000[0]
MOVP_M          var_D100[4] , var_D1000[1]
MOVP_M          var_D100[5] , var_D1000[2]
DATEMINUSP_M    var_D1000, var_D10 , var_R10
```

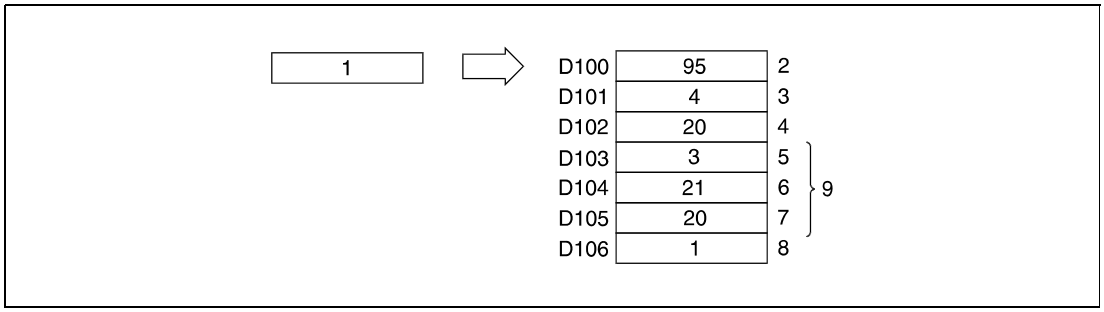
Ladder Diagram



NOTE

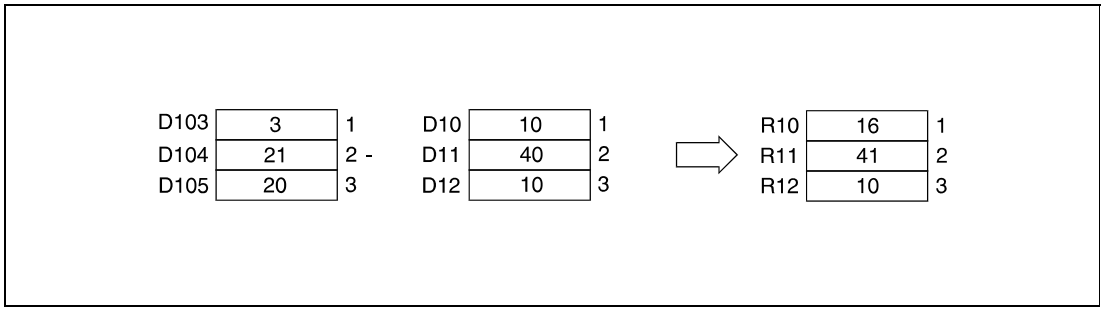
This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

The diagram below illustrates reading clock data via the DATERDP instruction.



[1] Clock element
[2] Year
[3] Month (January = 1, December = 12)
[4] Day
[5] Hour (24-hour format)
[6] Minute
[7] Second
[8] Day of the week
[9] Clock data

The diagram below illustrates the subtraction via the DATE-P instruction.



[1] Hour
[2] Minute
[3] Second

7.15.5 SECOND, SECONDP, HOUR, HOURP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

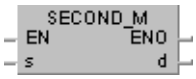
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
SECOND									
s	—	●	●	—	—	—	—	—	—
d	●	●	●	●	●	●	●	—	—
HOUR									
s	●	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—

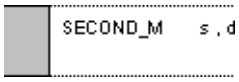
GX IEC Developer

MELSEC Instruction List

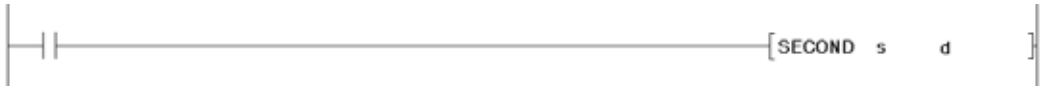
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
SECOND			
s	Hours, minutes, seconds	BIN 16-/32-bit	Array [0..2] of ANY16
d	Seconds		ANY32
HOUR			
s	Seconds	BIN 16-/32-bit	ANY32
d	Hours, minutes, seconds		Array [0..2] of ANY16

Functions

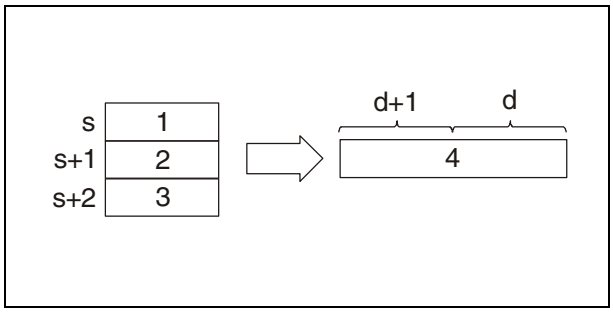
Changing the clock data format

SECOND Changing time format from hh:mm:ss to seconds

The SECOND instruction changes the clock data in the devices s+0 through s+2 (Array_s[0]) through (Array_s[2]) from the time format hh:mm:ss to the format seconds only. The result is stored in the devices specified by d and d+1 (Array_d[0]) through (Array_d[1]).

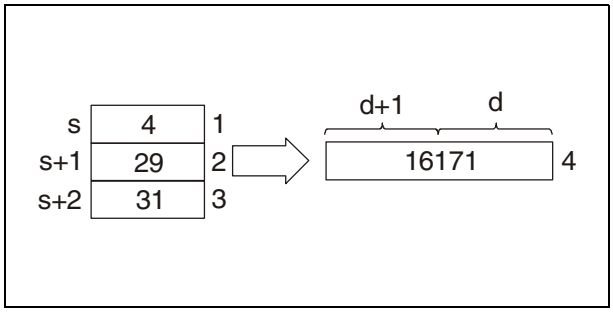
The following table shows the input ranges of clock data stored in s+0 through s+2 (Array_s[0]) through (Array_s[2]):

Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week
Input range	—	—	—	0–23	0–59	0–59	—
Devices	—	—	—	s+0 (Array_s[0])	s+1 (Array_s[1])	s+2 (Array_s[2])	—
Devices	—	—	—	—	—	d+0 (Array_d[0]) through d+1 (Array_d[1])	—



[1]Hour
[2]Minute
[3]Second
[4]Time value in seconds

The following diagram shows the conversion of 4 hours, 29 minutes, and 31 seconds into 16171 seconds.



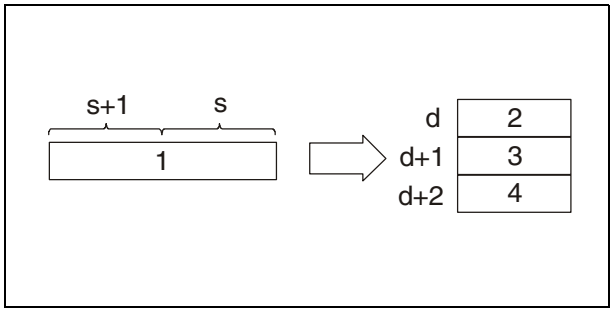
[1]Hour
[2]Minute
[3]Second
[4]Time value in seconds

HOUR Changing time format from seconds to hh:mm:ss

The HOUR instruction changes the clock data in the devices s+0 through s+1 (Array_s[0]) through (Array_s[1]) from the time format seconds only to the format hh:mm:ss.

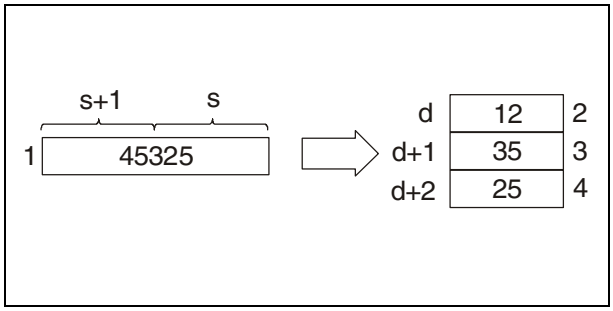
The following table shows the input ranges of clock data to be stored in d+0 through d+2 (Array_d[0]) through (Array_d[2]):

Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week
Input range	—	—	—	0 - 23	0 - 59	0 - 59	—
Devices	—	—	—	d+0 (Array_d[0])	d+1 (Array_d[1])	d+2 (Array_d[2])	—
Devices	—	—	—	—	—	s+0 (Array_s[0]) through s+1 (Array_s[1]	—



[1]Time value in seconds
[2]Hour
[3]Minute
[4]Second

The following diagram shows the conversion of 45325 seconds into 12 hours, 35 minutes, and 25 seconds.



[1]Time value in seconds
[2]Hour
[3]Minute
[4]Second

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The clock data in s+0 (Array_s[0]) through s+2 (Array_s[2]) for the SECOND instruction or in s+0 and s+1 for the HOUR instruction exceed the input range.
  (Error code 4100)
- The device specified by s exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU.)
  (Error code 4101)

Program Example 1

SECONDP

With leading edge from X20, the following program reads clock data from the internal CPU clock via the DATERDP instruction and stores it in the registers D10 through D16 (see first diagram after the program example below).

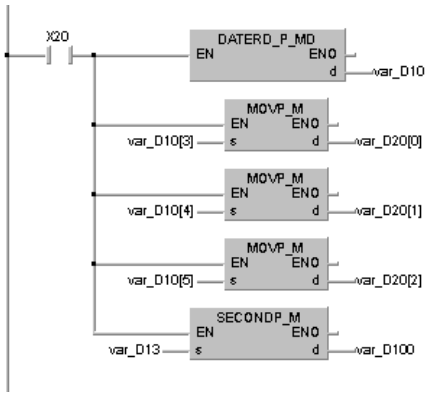
The hours, minutes, and seconds of clock data are converted into seconds only via the SECONDP instruction. The result is stored in D100 and D101 (see second diagram after the program example below).

Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week
Devices	D10 (var_D10[0])	D11 (var_D10[1])	D12 (var_D10[2])	D13 (var_D10[3])	D14 (var_D10[4])	D15 (var_D10[5])	D16 (var_D10[6])
Devices	—	—	—	D20 (var_D20[0])	D21 (var_D20[1])	D22 (var_D20[2])	—
Devices	—	—	—	—	—	D100 (var_D10[0]) to D101 (var_D10[1])	—

MELSEC Instruction List

```
MELSEC   LD        X20
         DATERDP   D10
         SECONDP   D20
                   D100
```

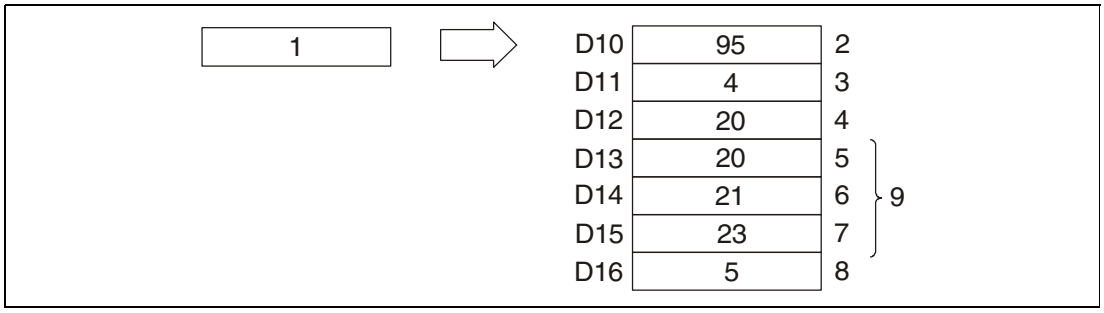
Ladder Diagram



IEC Instruction List

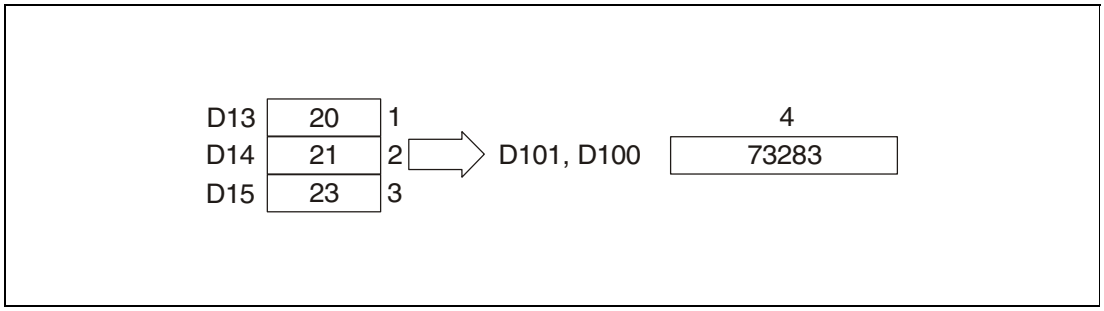
```
LD          X20
DATERD_P_MD var_D10
MOVP_M      var_D10[3] , var_D20[0]
MOVP_M      var_D10[4] , var_D20[1]
MOVP_M      var_D10[5] , var_D20[2]
SECONDP_M   var_D20 , var_D100
```

The diagram below illustrates reading clock data via the DATERDP instruction.



[1] Clock element
[2] Year
[3] Month (January = 1, December = 12)
[4] Day
[5] Hour (24-hour format)
[6] Minute
[7] Second
[8] Day of the week
[9] Clock data

The diagram below illustrates the conversion into seconds via the SECONDP instruction.



[1] Hour
[2] Minute
[3] Second
[4] Converted seconds

Program Example 2

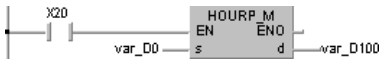
HOURP

With leading edge from X20, the following program converts the seconds stored in D0 and D1 into hours, minutes, and seconds. The result is stored in D100 through D102.

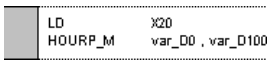
Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week
Devices	—	—	—	D0 (var_D0[1])	D1 (var_D0[2])	D2 (var_D0[3])	—
Devices	—	—	—	—	—	D100 (var_D100[0]) to D101 (var_D100[1])	—

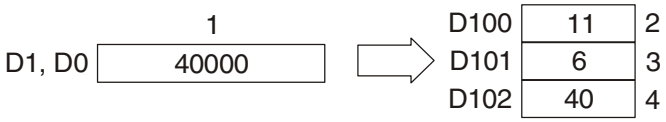
MELSEC Instruction List

Ladder Diagram



IEC Instruction List





[1] Value to be converted into seconds

[2] Hour

[3] Minute

[4] Second

NOTE *These program examples will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

7.15.6 DT=, DT<>, DT>, DT<=, DT<, DT>=

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

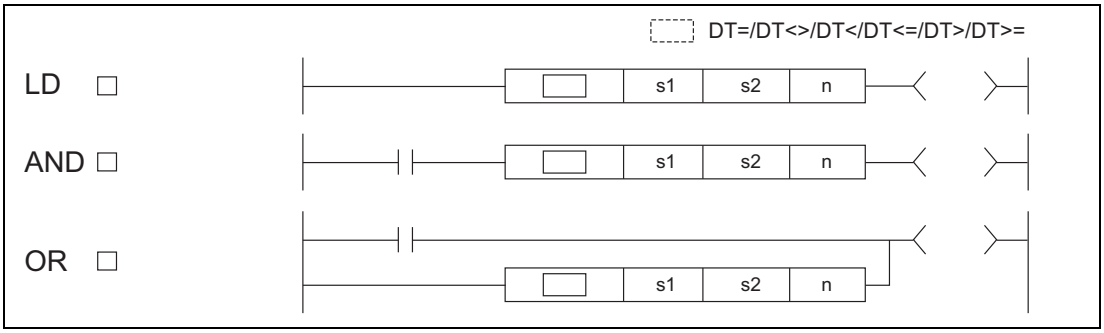
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
n	—	●	●	●	●	●	●	●	●

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s1	First number of device storing the data to be compared	BIN 16-bit
s2		
n	Value of the data to be compared or the number of the stored data to be compared	

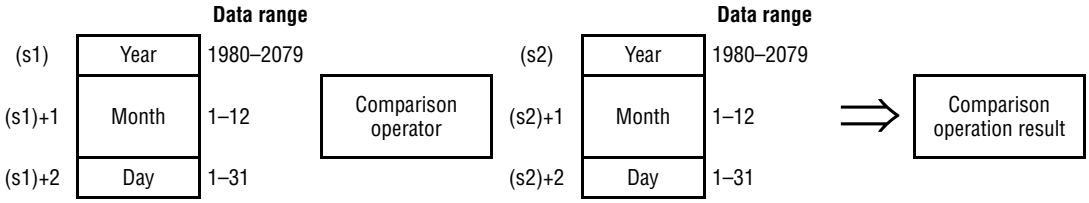
Functions

Date comparison

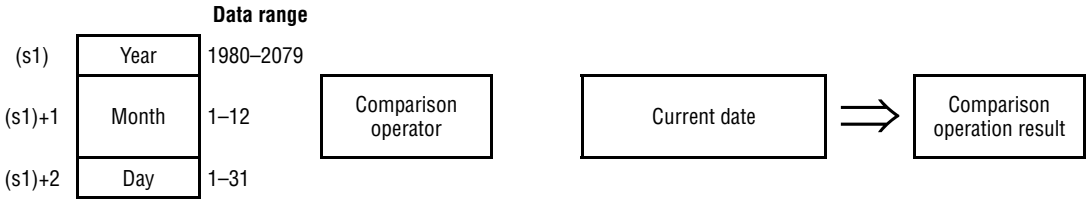
DT=, DT<>, DT>, DT<=, DT<, DT>= Date comparison

This instruction compares the date data specified by s1 with those specified by s2, or the date data specified by s1 with current date data. Setting n determines the data to be compared.

- Comparison of given date data
  This instruction treats the date data specified by s1 and s2 as a normally open contact, and then compares the data in accordance with the value of n.



- Comparison of current date data
  This instruction treats the date data specified by s1 and the current date data as a normally open contact, and then compares the data in accordance with the value of n. Date data specified by s2 is treated as dummy data, and is ignored.



NOTE

When either s1 or s2 corresponds to any of the following conditions in comparing given or current date data with given date data, an operation error (error code 4101) or a malfunction may occur.

- *The range of the devices to be used for the index modification is specified over the range of the device specified by s1 or s2.*
- *File registers are specified by by s1 or s2 without a register set.*

This instruction sets BIN values for each item.

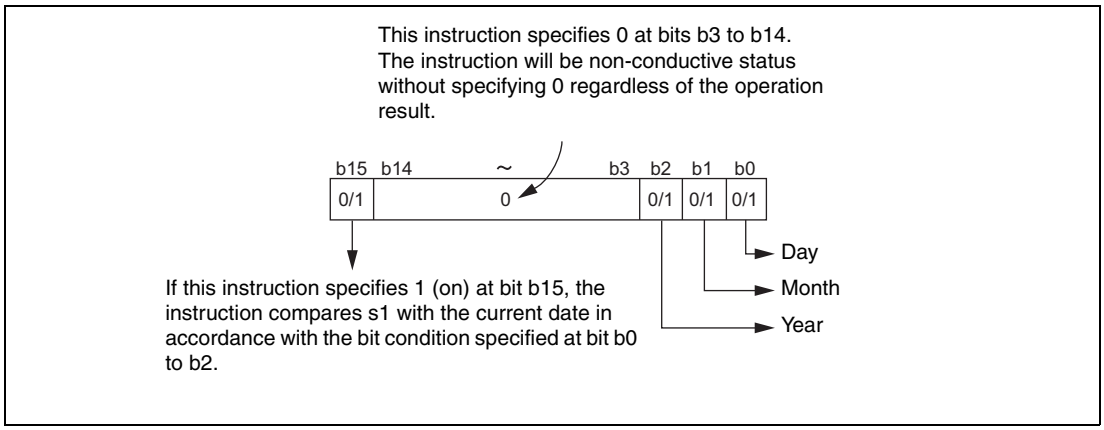
This instruction sets the year of four digits selected from 1980 to 2079 with the BIN value specified by s1 or s2.

This instruction sets the month selected from 1 to 12 (January to December) with the BIN value specified by s1+1 or s2+1.

This instruction sets the day selected from 1 to 31 (1st to 31st) for with the BIN value specified by s1+2 or s2+2.

This instruction specifies the following values at n so that the data to be compared can be specified.

The bit configuration specified at n is as follows.



- Date data to be compared (from bit 0 to bit 2)
  0: Does not compare specified date data (year/month/day).
  1: Compares specified date data (year/month/day).
- Operation data to be compared (bit 15)
  0: Compares the date data specified by s1 with the date data specified by s2.
  1: Compares the date data specified by s1 with the current date data.
  Ignores the date data specified by s2.
- The following table shows processing details.

n value for comparison of specified date data with given date data	n value for comparison of specified date data with current date data	Date to be compared	Processing details
0001H	8001H	Day	Comparison of days (s1+2)
0002H	8002H	Month	Comparison of months (s1+1)
0003H	8003H	Month, day	Comparison of months (s1+1) and days (s1+2)
0004H	8004H	Year	Comparison of years (s1)
0005H	8005H	Year, day	Comparison of years (s1) and days (s1+2)
0006H	8006H	Year, month	Comparison of years (s1) and months (s1+1)
0007H	8007H	Year, month, day	Comparison of years (s1), months (s1+1) and days (s1+2)
Other than 0001H to 0007H, 8001H to 8007H		No objects	No comparison of years (s1), months (s1+1) and days (s1+2) (Non-conductive)

If the data stored in the devices to be compared are not recognized as date data, SM709 will be turned on after the instruction execution and no-conductive status will be made. Even if they are not recognized as date data but the range of the devices is within the setting range, SM709 will not be turned on.

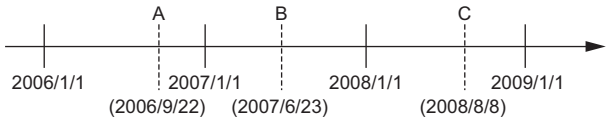
Moreover, if the range of devices specified by s1 to s1+2 or s2 to s2+2 exceeds the range of specified devices, SM709 will be turned on after the instruction execution and no-conductive status will be made.

Once SM709 is turned on, on-status will be retained till when the CPU modules are reset or powered off. Therefore, turn off SM709 if necessary.

The following table shows the comparison operation results for each instruction:

Instruction Symbol	Comparison Operation Results	
	1	0
=	s1 = s2	s1 ≠ s2
<>	s1 ≠ s2	s1 = s2
>	s1 > s2	s1 ≤ s2
<=	s1 ≤ s2	s1 > s2
<	s1 < s2	s1 ≥ s2
>=	s1 ≥ s2	s1 < s2

- The following figure shows the comparison example of dates:



The following table shows the conductive states resulting from performing the comparison operation of the dates A, B, and C shown above. Even if the objects to be compared are under the same condition, the comparison operation results vary depending on the objects selected.

Comparison Objects	Comparison Condition		
	A<B	B<C	A<C
Day	●	○	○
Month	○	●	○
Month, day	○	●	○
Year	●	●	●
Year, day	●	●	●
Year, month	●	●	●
Year, month, day	●	●	●
No objects	○	○	○

●Conductive
○Non-conductive

- Even if the dates to be compared do not exist practically, this instruction executes the comparison operation for the objects with the settable dates in accordance with the following condition.
  – Date A: 2006/02/30 (This date is settable, though it does not exist.)
  – Date B: 2007/03/29
  – Date C: 2008/02/31 (This date is settable, though it does not exist.)

Comparison Objects	Comparison Condition		
	A<B	B<C	A<C
Day	○	○	●
Month	○	○	○
Month, day	●	○	●
Year	●	●	●
Year, day	●	●	●
Year, month	●	●	●
Year, month, day	●	●	●
No objects	○	○	○

●Conductive
○Non-conductive

Program Example 1

LDDT=

The following program compares the data stored in D0 with the data (year, month, and day) stored in D10, and turns Y33 ON when the data stored in D0 meet the data stored in D10.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 2

ANDDT<>

The following program compares the data stored in D0 with the current date data (year and month), and turns Y33 ON when the data stored in D0 do not meet the current date data, when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 3

ANDDT>

The following program compares the data stored in D0 with the data (year and day) stored in D10, and turns Y33 ON when the data value stored in D10 is smaller than the data value stored in D0, when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 4

ORDT<=

The following program compares the data stored in D0 with the current date data (year), and turns Y33 ON when the value of the current date data is the data value stored in D0 or larger.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

7.15.7 TM=, TM<>, TM>, TM<=, TM<, TM>=

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	●

[1] QnU(D)(H)CPU: The serial number (first five digits) is "10102" or higher.
QnUDE(H)CPU: The serial number (first five digits) is "10102" or higher.

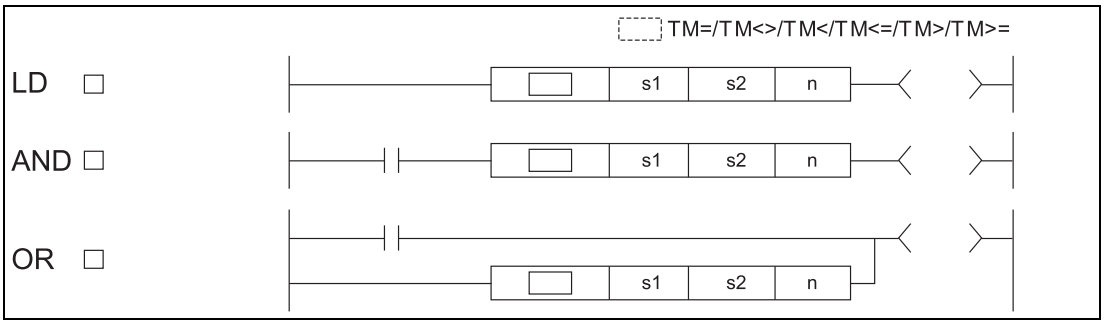
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
n	—	●	●	●	●	●	●	●	●

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s1	First number of device storing the data to be compared	BIN 16-bit
s2		
n	Value of the data to be compared or the number of the stored data to be compared	

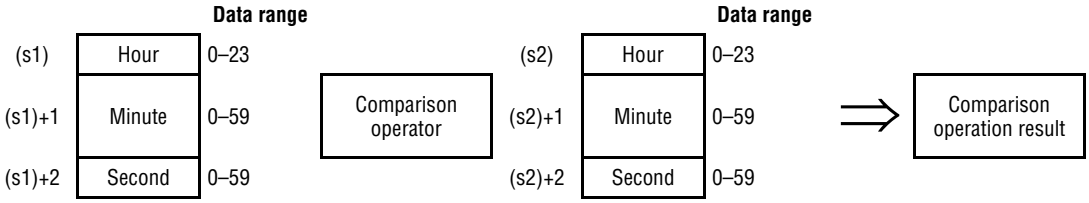
Functions **Clock comparison**

TM=, TM<>, TM>, TM<=, TM<, TM>= Clock comparison

This instruction compares the clock data specified by s1 with those specified by s2, or the clock data specified by s1 with current time data. Setting n determines the data to be compared.

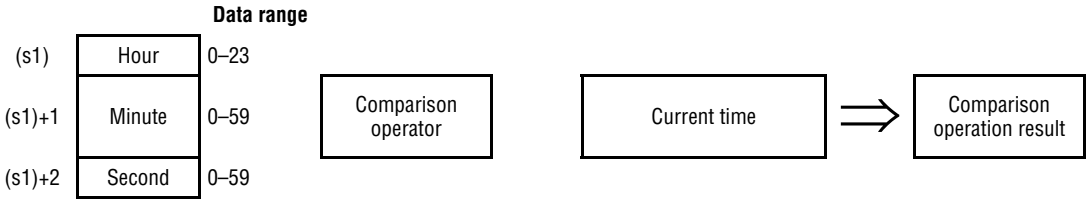
- Comparison of given clock data

  This instruction treats the clock data specified by s1 and s2 as a normally open contact, and then compares the data in accordance with the value of n.



- Comparison of current time data

  This instruction treats the clock data specified by s1 and the current time data as a normally open contact, and then compares the data in accordance with the value of n.
  Time data specified by s2 is treated as dummy data, and is ignored.



NOTE *When either s1 or s2 corresponds to any of the following conditions in comparing given or current time data with given clock data, an operation error (error code 4101) or a malfunction may occur.*

- *The range of the devices to be used for the index modification is specified over the range of the device specified by s1 or s2.*
- *File registers are specified by by s1 or s2 without a register set.*

This instruction sets BIN values for each item.

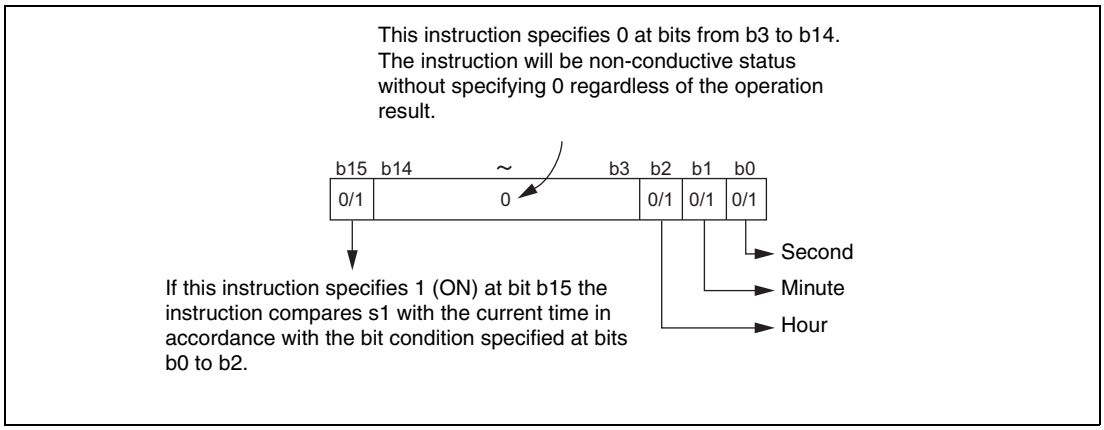
This instruction sets the time selected from 0 to 23 (midnight to 23 o'clock) with the BIN value specified by s1 or s2 (uses 24-hour clock).

This instruction sets the minute selected from 0 to 59 with the BIN value specified by s1+1 or s2+1.

This instruction sets the second selected from 0 to 59 for with the BIN value specified by s1+2 or s2+2.

This instruction specifies the following values at n so that the data to be compared can be specified.

The bit configuration specified at n is as follows.



- Clock data to be compared (from bit 0 to bit 2)
  0: Does not compare specified clock data (hour/minute/second).
  1: Compares specified clock data (hour/minute/second).
- Operation data to be compared (bit 15)
  0: Compares the clock data specified by s1 with the clock data specified by s2.
  1: Compares the clock data specified by s1 with the current time data.
  Ignores the clock data specified by s2.
- The following table shows processing details.

n value for comparison of specified clock data with given clock data	n value for comparison of specified clock data with current time data	Time to be compared	Processing details
0001H	8001H	Second	Comparison of seconds (s1+2)
0002H	8002H	Minute	Comparison of minutes (s1+1)
0003H	8003H	Minute, second	Comparison of minutes (s1+1) and seconds (s1+2)
0004H	8004H	Hour	Comparison of hours (s1)
0005H	8005H	Hour, second	Comparison of hours (s1) and seconds (s1+2)
0006H	8006H	Hour, minute	Comparison of hours (s1) and minutes (s1+1)
0007H	8007H	Hour, minute, second	Comparison of hours (s1), minutes (s1+1) and seconds (s1+2)
Other than 0001H to 0007H, 8001H to 8007H		No objects	No comparison of hours (s1), minutes (s1+1) and seconds (s1+2) (Non-conductive)

If the data stored in the devices to be compared are not recognized as clock data, SM709 will be turned on after the instruction execution and no-conductive status will be made.

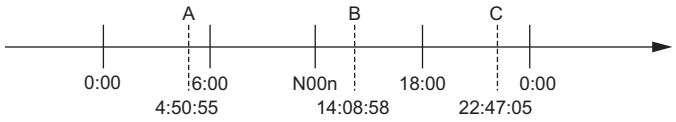
Moreover, if the range of devices specified by s1 to s1+2 or s2 to s2+2 exceeds the range of specified devices, SM709 will be turned on after the instruction execution and no-conductive status will be made.

Once SM709 is turned on, on-status will be retained until the CPU modules are reset or powered off. Therefore, turn off SM709 if necessary.

The following table shows the comparison operation results for each instruction.

Instruction Symbol	Comparison Operation Results	
	1	0
=	s1 = s2	s1 ≠ s2
<>	s1 ≠ s2	s1 = s2
>	s1 > s2	s1 ≤ s2
<=	s1 ≤ s2	s1 > s2
<	s1 < s2	s1 ≥ s2
>=	s1 ≥ s2	s1 < s2

- The following figure shows the comparison example of time.



The following table shows the conductive states resulting from performing the comparison operation of the clock data A, B, and C shown above. Even if the objects to be compared are under the same condition, the comparison operation results vary depending on the objects selected.

Comparison Objects	Comparison Condition		
	A<B	B<C	A<C
Second	●	○	○
Minute	○	●	○
Minute, second	○	●	○
Hour	●	●	●
Hour, second	●	●	●
Hour, minute	●	●	●
Hour, minute, second	●	●	●
No objects	○	○	○

●Conductive
○Non-conductive

Program Example 1

LDTM=

The following program compares the data stored in D0 with the data (hour, minute, and second) stored in D10, and turns Y33 ON when the data stored in D0 meet the data stored in D10.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 2

ANDTM<>

The following program compares the data stored in D0 with the current time data (hour and minute), and turns Y33 ON when the data stored in D0 do not meet the current date data, when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 3

ANDTM>

The following program compares the data stored in D0 with the data (hour and second) stored in D10, and turns Y33 ON when the data value stored in D10 is smaller than the data value stored in D0, when M0 is turned on.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 4

ORTM<=

The following program compares the data stored in D0 with the current time data (hour), and turns Y33 ON when the value of the current time data is the data value stored in D0 or larger.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

7.16 Expansion clock instructions

The expansion clock instructions read, add and subtract the clock data of the internal CPU clock. In contrast to the instructions described in section 7.15 these instructions can process milliseconds as well.

The table below gives an overview of these instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Reading clock data	S.DATERD	
	SP.DATERD	
Adding clock data	S.DATE+	
	SP.DATE+	
Subtracting clock data	S.DATE-	
	SP.DATE-	

7.16.1 S.DATERD, SP.DATERP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●[1)]	●[1)]	●[1)]	●	●

[1] High performance model QCPU, Process CPU, Redundant CPU:
The serial number (first five digits) is "07032" or higher.

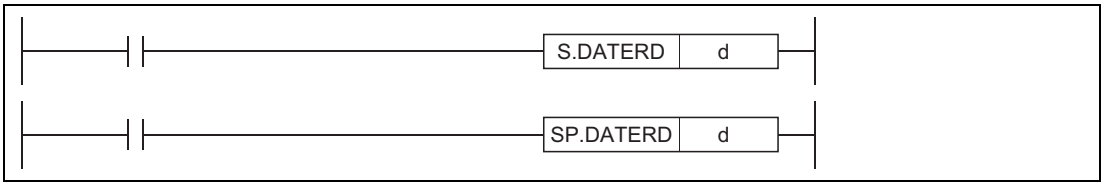
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
d	First number of device storing clock data being read	BIN 16-bit	Array [0..7] of ANY16

Functions **Reading expansion clock data**

S.DATERD Read instruction

The S.DATERD instruction reads year, month, day, hour, minute, second, weekday, and millisecond from the clock element of the CPU module and stores the clock data in binary format in the devices specified by d+0 (Array_d[0]) through d+7 (Array_d[7]). The assignment of registers to clock data is illustrated below:

d+0, Array_d[0] = year (1)

d+1, Array_d[1] = month (January = 1, December = 12) (2)

d+2, Array_d[2] = day (3)

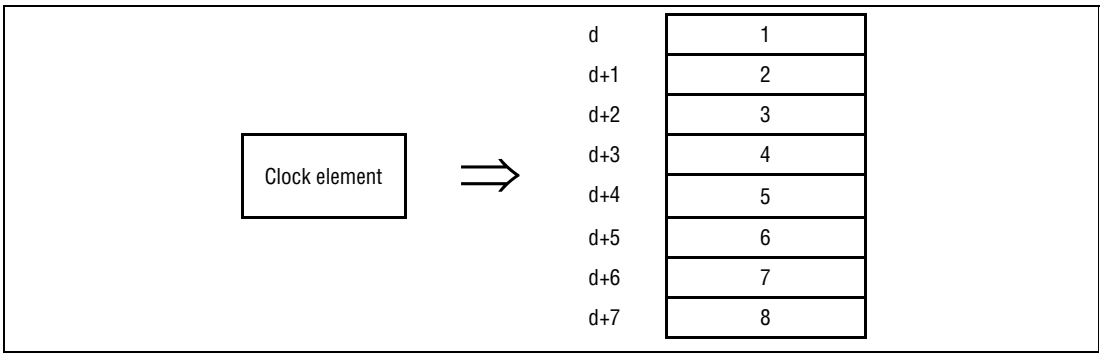
d+3, Array_d[3] = hour (24 hour format) (4)

d+4, Array_d[4] = minute (5)

d+5, Array_d[5] = second (6)

d+6, Array_d[6] = day of the week (7)

d+7, Array_d[7] = millisecond (8)



The following table contains the value range of clock data in d+0 through d+7 (Array_d[0]) through (Array_d[7]):

Clock data	Year	Month	Day	Hour	Minute	Second	Day of the week	Millisecond
Input range	1980–2079	1–12	1–31	0–23	0–59	0–59	0–6	0–999
Devices	d+0 (Array_d[0])	d+1 (Array_d[1])	d+2 (Array_d[2])	d+3 (Array_d[3])	d+4 (Array_d[4])	d+5 (Array_d[5])	d+6 (Array_d[6])	d+7 (Array_d[7])

The "year" is stored as four-digit indication.

The day of the week stored in d+6 (Array_d[6]) is indicated from 0 to 6. The table below shows the assignment of weekdays:

Weekday	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Storage value	0	1	2	3	4	5	6

Leap years are calculated automatically by the CPU clock.

Operation Errors In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

Program Example

SP.DATERD

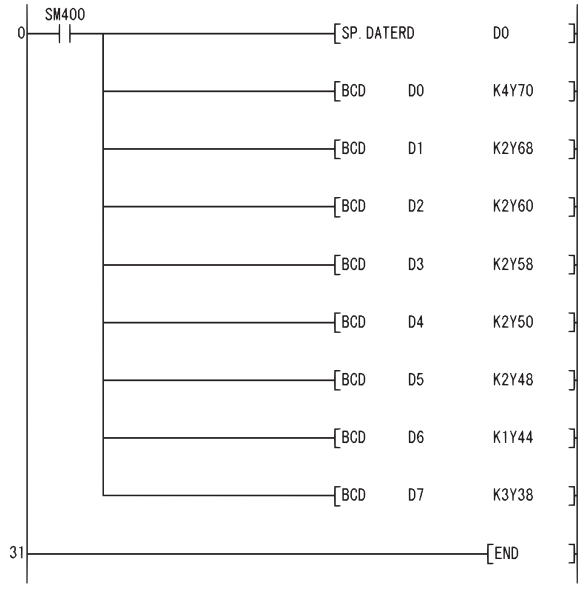
The following program reads clock data from the internal CPU clock and outputs it in BCD format at the outputs as follows:

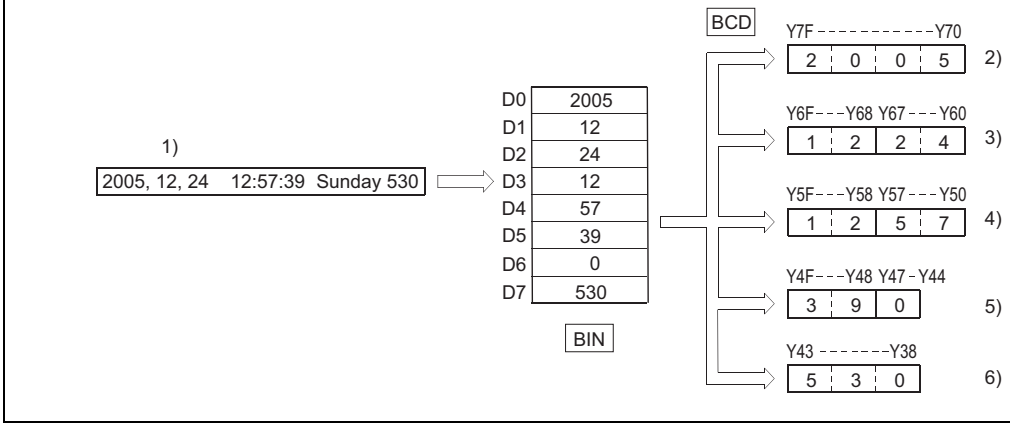
Y70 - Y7F = year
Y68 - Y6F = month
Y60 - Y67 = day
Y58 - Y5F = hour
Y50 - Y57 = minute
Y48 - Y4F = second
Y44 - Y47 = day of the week
Y38 - Y43 = millisecond

Clock data	Year	Month	Day	Hour	Minute	Second	Day of the week	Millisecond
Devices	D0	D1	D2	D3	D4	D5	D6	D7

Instruction List

Ladder Diagram





[1] Clock data
[2] Year
[3] Month, day
[4] Hour, minute
[5] Second, day of the week
[6] Millisecond

NOTES

This instruction reads clock data and stores those to a specified device even if a wrong clock data is set to the CPU module (example: Feb. 30th). When setting clock data with the DATEWR instruction or witha programming tool, make sure to set a correct data.

Time error of reading a clock data of millisecond is a maximum of 2 ms. (Difference between the data memorized by clock element inside of the CPU module and the data read by this function.)

Specifying digit for the bit device can be used only when the following two conditions are met:

- *Digit specification: K4*
- *Head of device: multiple of 16*

When the above conditions are not met, INSTRCT CODE ERR. (Error code 4004) will occur.

7.16.2 S.DATE+, SP.DATE+

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●[1)]	●[1)]	●[1)]	●	●

[1] High performance model QCPU, Process CPU, Redundant CPU:
The serial number (first five digits) is "07032" or higher.

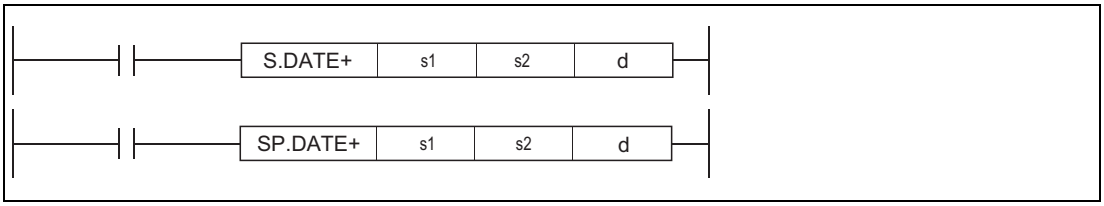
Devices

	UsableDevices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s1	Clock data to be added to	BIN 16-bit	Array [0..4] of ANY16
s2	Clock data to be added		
d	First number of device storing the clock data of the operation result		

Functions

Adding expansion clock data

S.DATE+ Addition instruction

The S.DATE+ instruction adds the clock data stored in the devices specified from s2 on to the clock data stored in the devices specified from s1 on. The clock data of the operation result is stored in the devices specified from d.

The following table contains the value range of clock data in (s1)+0 through (s1)+4 (Array_s1[0] through Array_s1[4]), (s2)+0 through (s2)+4 (Array_s2[0] through Array_s2[4]), and d+0 through d+4 (Array_d[0] through Array_d[4]):

Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week	Millisecond
Input range	—	—	—	0–23	0–59	0–59	—	0–999
Devices	—	—	—	(s1)+0 (Array_s1[0])	(s1)+1 (Array_s1[1])	(s1)+2 (Array_s1[2])	—	(s1)+4 (Array_s1[4])
Devices	—	—	—	(s2)+0 (Array_s2[0])	(s2)+1 (Array_s2[1])	(s2)+2 (Array_s2[2])	—	(s2)+4 (Array_s2[4])
Devices	—	—	—	d+0 (Array_d[0])	d+1 (Array_d[1])	d+2 (Array_d[2])	—	d+4 (Array_d[4])

s1	Hour		s2	Hour		d	Hour
(s1)+1	Minute		(s2)+1	Minute		d+1	Minute
(s1)+2	Second	+	(s2)+2	Second	⇒	d+2	Second
(s1)+3	—		(s2)+3	—		d+3	—
(s1)+4	Millisecond		(s2)+4	Millisecond		d+4	Millisecond

For example, adding the time 7:48:10:500 to 6:32:40:875 would result in the following operation:

s1	Hour: 6		s2	Hour: 7		d	Hour: 14
(s1)+1	Minute: 32		(s2)+1	Minute: 48		d+1	Minute: 20
(s1)+2	Second: 40	+	(s2)+2	Second: 10	⇒	d+2	Second: 51
(s1)+3	—		(s2)+3	—		d+3	—
(s1)+4	Millisecond: 875		(s2)+4	Millisecond: 500		d+4	Millisecond: 375

If the addition result of clock data exceeds 24 hours, 24 hours are subtracted automatically to achieve a correct time value.

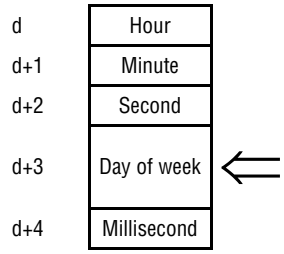
For example, when the time 20:20:20:500 is added to 14:20:30:875, the result is not 34:40:51:375, but 10:40:51:375.

s1	Hour: 14		s2	Hour: 20		d	Hour: 10
(s1)+1	Minute: 20		(s2)+1	Minute: 20		d+1	Minute: 40
(s1)+2	Second: 30	+	(s2)+2	Second: 20	⇒	d+2	Second: 51
(s1)+3	—		(s2)+3	—		d+3	—
(s1)+4	Millisecond: 875		(s2)+4	Millisecond: 500		d+4	Millisecond: 375

NOTE

Devices s1+3, s2+3, and d+3 are not used for operation.

A clock data read by the S(P).DATERD instruction can be directly added.



When the clock data is read by the S(P).DATERD instruction, day of week is inserted between "second" and "millisecond". If the S(P).DATE+ instruction is used to add the clock data, the data can be directly used for addition since it does not perform the calculation for the day of a week.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The clock data in s1 and s2 exceed the input range.
  (Error code 4100)
- The device specified by s1, s2 or d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

NOTE

Specifying digit for the bit device can be used only when the following two conditions are met:

- *Digit specification: K4*
- *Head of device: multiple of 16*

When the above conditions are not met, INSTRCT CODE ERR. (Error code 4004) will occur.

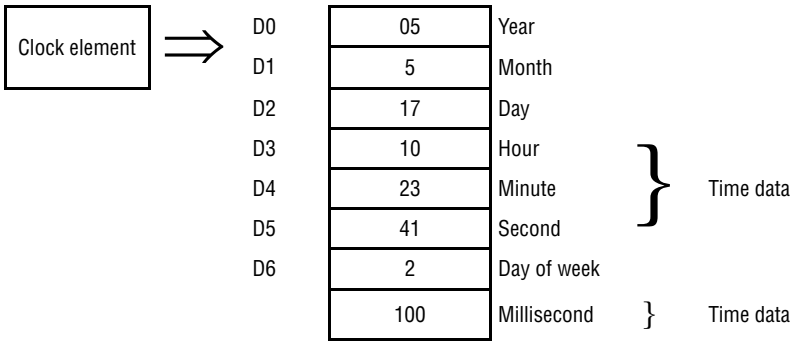
Program Example

SP.DATE+P

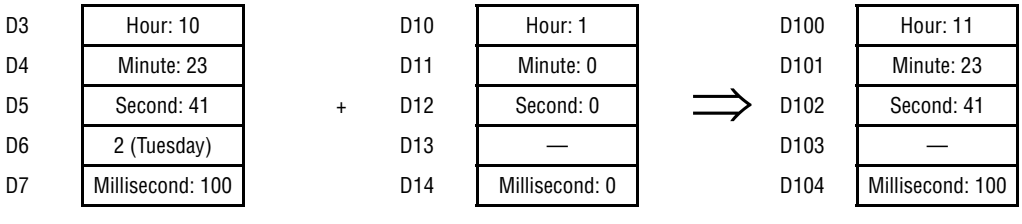
With leading edge from X20, the following program adds 1 hour to the clock data read from the clock element, and stores the results into the area starting from D100.

Instruction List	Ladder Diagram

Time data read operation by SP.DATERD instruction



Addition by SP.DATE+ instruction



7.16.3 S.DATE-, SP.DATE-

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●[1)]	●[1)]	●[1)]	●	●

[1] High performance model QCPU, Process CPU, Redundant CPU with serial number (first five digits) of "07032" or higher

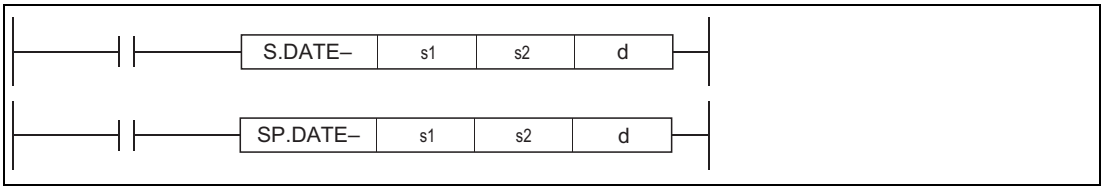
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s1	First number of device storing clock data to be subtracted from	BIN 16-bit	Array [0..4] of ANY16
s2	First number of device storing clock data to be subtracted		
d	First number of device storing the clock data of the subtraction result		

Functions **Subtracting expansion clock data**

S.DATE- **Subtraction instruction**

The S.DATE instruction subtracts clock data stored in the device specified from s2 on from the clock data in the device specified from s1 on. The clock data of the operation result is stored in the device specified from d on.

The following table shows the input ranges of clock data stored in (s1)+0 through (s1)+4 (Array_s1[0] through Array_s1[4]), (s2)+0 through (s2)+4 (Array_s2[0] through Array_s2[4]) and d+0 through d+4 (Array_d[0] thsrough Array_d[4]) .

Clock Data	Year	Month	Day	Hour	Minute	Second	Day of the week	Millisecond
Input range	—	—	—	0–23	0–59	0–59	—	0–999
Devices	—	—	—	s1+0 (Array_s1[0])	s1+1 (Array_s1[1])	s1+2 (Array_s1[2])	—	s1+4 (Array_s1[4])
Devices	—	—	—	s2+0 (Array_s2[0])	s2+1 (Array_s2[1])	s2+2 (Array_s2[2])	—	s2+4 (Array_s2[4])
Devices	—	—	—	d+0 (Array_d[0])	d+1 (Array_d[1])	d+2 (Array_d[2])	—	d+4 (Array_d[4])

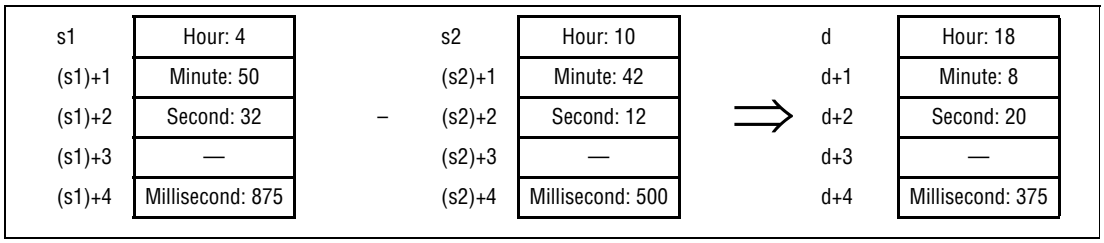
s1	Hour		s2	Hour		d	Hour
(s1)+1	Minute		(s2)+1	Minute		d+1	Minute
(s1)+2	Second	–	(s2)+2	Second	⇒	d+2	Second
(s1)+3	—		(s2)+3	—		d+3	—
(s1)+4	Millisecond		(s2)+4	Millisecond		d+4	Millisecond

For example, subtracting the clock time 3:50:10:500 from 10:40:20:875 would result in the following operation:

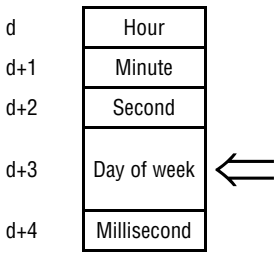
s1	Hour: 10		s2	Hour: 3		d	Hour: 6
(s1)+1	Minute: 40		(s2)+1	Minute: 50		d+1	Minute: 50
(s1)+2	Second: 20	–	(s2)+2	Second: 10	⇒	d+2	Second: 10
(s1)+3	—		(s2)+3	—		d+3	—
(s1)+4	Millisecond: 875		(s2)+4	Millisecond: 500		d+4	Millisecond: 375

If the subtraction result of clock data becomes negative, 24 hours are added automatically to achieve a correct time value.

For example, when the clock time 10:42:12:500 is subtracted from 4:50:32:875, the result is not -6:8:20:375, but 18:8:20:375.



NOTE *Devices (s1)+3, (s2)+3, and d+3 are not used for operation.*
A clock data read by the S(P).DATERD instruction can be directly subtracted.



When the clock data is read by the S(P).DATERD instruction, day of week is inserted between "second" and "millisecond". If the S(P).DATE– instruction is used to read the clock data, the data can be directly used for subtraction since it does not perform the calculation for the day of a week.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The clock data in (s1)+0 through (s1)+4 ((Array_s1[0] through Array_s1[4])) and (s2)+0 through (s2)+4 ((Array_s2[0] through Array_s2[4])) exceed the input range.
  (Error code 4100)
- The device specified by s1, s2 or d exceeds the range of the corresponding device.
  (For the Universal model QCPU, LCPU)
  (Error code 4101)

NOTE *Specifying digit for the bit device can be used only when the following two conditions are met:*

- *Digit specification: K4*
- *Head of device: multiple of 16*

When the above conditions are not met, INSTRCT CODE ERR. (Error code 4004) will occur.

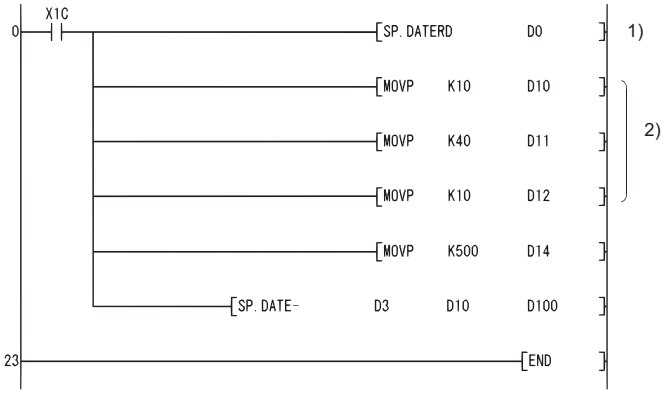
Program Example

SP.DATE–

With leading edge from X1C, the following program subtracts the time data stored in the area starting from D10 from the clock data read from the clock element, and stores the results into the area starting from D100.

Instruction List

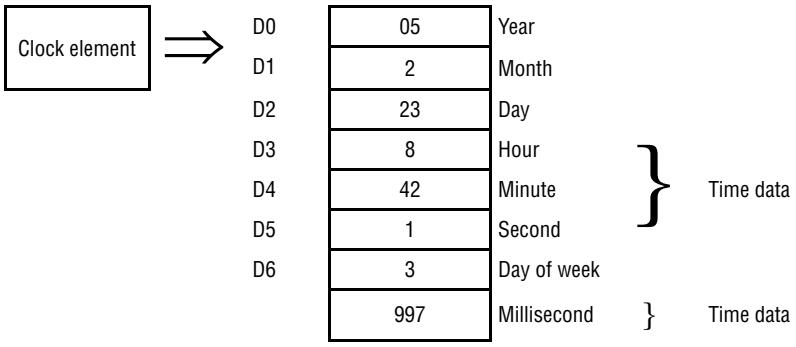
Ladder Diagram



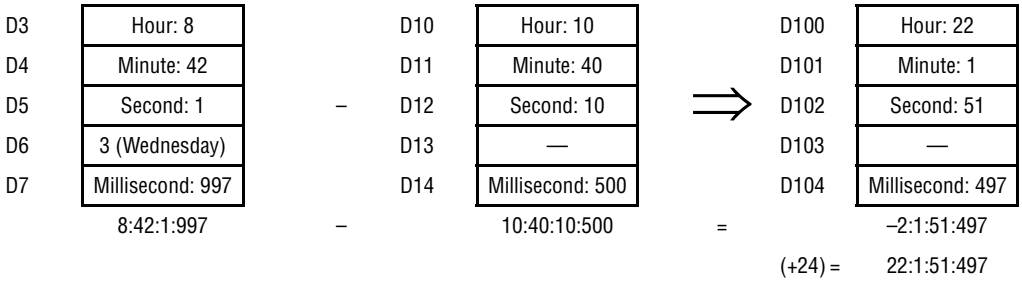
[1] Reads out the clock element data to D0 or later.

[2] Sets the time to D10 or later.

Time data read operation by SP.DATERD instruction



Subtraction by SP.DATE– instruction



7.17 Program control instructions

The program control instructions toggle different program operation modes and check the program execution status. The table below gives an overview of the instructions:

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Switching programs into stand-by mode	PSTOP	PSTOP_M
	PSTOPP	PSTOPP_M
Switching programs into stand-by mode and reset of outputs	POFF	POFF_M
	POFFP	POFFP_M
Switching programs into scan execution mode	PSCAN	PSCAN_M
	PSCANP	PSCANP_M
Switching programs into low-speed execution mode	PLOW	PLOW_M
	PLOWP	PLOWP_M
Checking the program execution status	LDPCHK	
	ANDPCHK	
	CRPCHK	

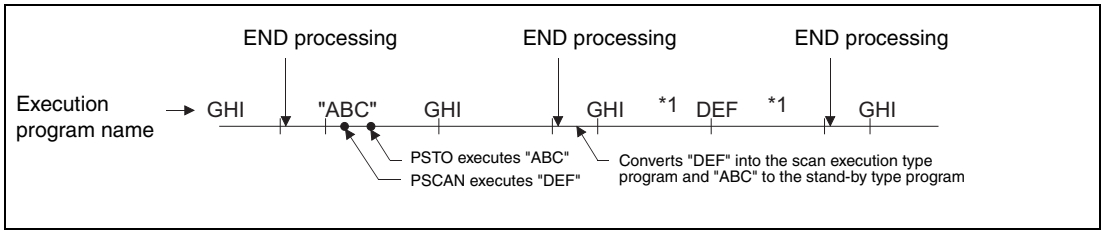
NOTE *Please check, whether these functions are available and supported by your version of the GX IEC Developer.*

Processing when the execution type is converted with the program control instruction is as follows:

Execution type before change	Executed Instruction			
	PSCAN	PSTOP	POFF	PLOW
Scan execution type	No change – remains scan type execution.	Becomes stand-by type.	Output turned OFF in next scan. Becomes stand-by type from the next scan after that.	Becomes low speed execution type.
Initial execution type	Becomes scan execution type.	No change-remains stand-by type	Ignored	
Stand-by type				
Low speed execution type	Low speed execution type execution is stopped, becomes scan execution type from the next scan. (Execution from step 0)	Low speed execution type execution is stopped, becomes stand-by type from the next scan.	Low speed execution type execution is stopped, and output is turned OFF in the next scan. Becomes stand-by type from the next scan after that.	No change – remains low speed execution type.
Fixed scan execution type	Becomes scan execution type.	Becomes stand-by type.	Output turned OFF in next scan. Becomes stand-by type from the next scan after that.	Becomes low speed execution type.

NOTE *Once the fixed scan execution type program is changed to another execution type, it cannot be returned to the fixed scan execution type.*

As program execution type conversions by PSCAN and PSTOP instructions occur at the END processing, such conversions are impossible during program execution.
When different execution types have been set for the same program in the same scan, the execution type will be that specified by the execution switching command that was executed last.



[1] The order of "GHI" and "DEF" program execution is determined by the program settings parameters.

Switching from the fixed scan execution type program to the execution type program is performed in the following timing.

- For the Universal model QCPU, LCPU

  The execution type is changed when the execution of the fixed scan execution type is stopped at the END processing after the program control instruction execution.

- Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

  The execution of the fixed scan execution type is stopped at the execution of the program control instruction, and the execution type is changed at the END processing.

When the POFF instruction is executed, the output is turned OFF at the next scan, and the execution type will be the stand-by type at the second next scan and later.

If executed prior to the output OFF processing, the program control instruction is ignored.

7.17.1 PSTOP, PSTOPP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

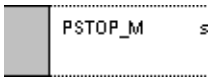
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—

GX IEC Developer

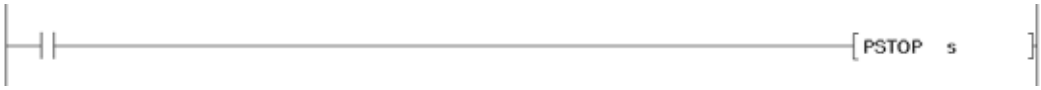
MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	File name of program file to be set into stand-by mode or first number of device storing such data	Character string

Functions

Setting a program into the stand-by mode

PSTOP Switch instruction for the stand-by mode

The PSTOP instruction sets the program specified by the device in s into the stand-by mode. In this mode the program is only executed if requested.

Only program files stored in the internal memory (drive 0) can be set into the stand-by mode.

The stand-by mode is only entered after END processing.

The PSTOP instruction is even given priority if the execution mode is specified via parameters.

The file extension .QPG is not needed to be entered for file specification since the type of file is recognized automatically.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The specified program file does not exist.
  (Error code 2410)
- The program type of the file name specified by s is an SFC program.
  (Error code 2412)
- The file name storage destination device of s exceeds the range of the corresponding device.
  (Error code 4101)

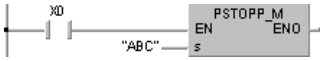
Program Example

PSTOPP

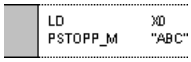
With leading edge from X0, the following program sets a program named "ABC" into the stand-by mode.

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



7.17.2 POFF, POFFP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

Devices

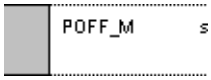
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—

GX IEC Developer

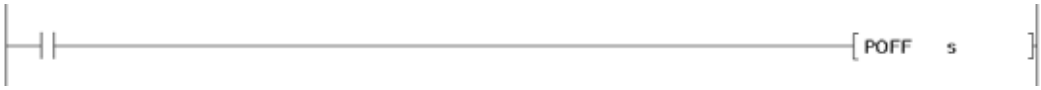
MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	First number of device storing file name of program file to be set into stand-by mode including reset of outputs	Character string

Functions

Setting a program into the stand-by mode including reset of the outputs

POFF Switch instruction for the stand-by mode with reset outputs

The POFF instruction sets the program specified by the device in s into the stand-by mode and resets the outputs addressed by the program.

- Scan execution type:
  Turns OFF outputs at the next scan (Non-execution processing). Programs are set as the stand-by type after the subsequent scan.
- Low speed execution type:
  Stops the execution of the low speed execution type program and turns OFF outputs at the next scan. Programs are set as the stand-by type after the subsequent scan.

Only program files stored in the internal memory (drive 0) can be set into the stand-by mode.

The POFF instruction is even given priority if the execution mode is specified via parameters.

The file extension .QPG is not needed to be entered for file specification since the type of file is recognized automatically.

NOTE

Non-execution processing is identical to the processing that is conducted when the condition contacts for the individual coil instructions are in the OFF state.

The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

Instruction	Condition of contacts and coils
OUT instruction	All contacts and coils, designated by the OUT instruction are reset.
SET instruction	All contacts and coils, designated by these instructions remain their condition.
RST instruction	
SFT instruction	
Basic instructions	
Application instructions	
PLS instruction	All contacts and coils, designated by these instructions adopt a condition as if the execution conditions of the instructions were not set.
Instructions generating an output pulse	
Setting values of low- and high-speed timers	The setting values are reset to 0.
Setting values of retentive timers	The setting values remain set.
Setting values of counters	

Operation Errors

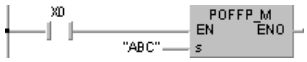
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The specified program file does not exist.
  (Error code 2410)
- The file name storage destination device of s exceeds the range of the corresponding device.
  (Error code 4101)

Program Example

POFFP

With leading edge from X0, the following program sets a program named "ABC" into the stand-by mode. First in this mode all outputs, addressed by the program "ABC" are reset to the same status as if the execution conditions for the instructions addressing them were not set. Then the program "ABC" enters the stand-by mode.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

7.17.3 PSCAN, PSCANP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	●

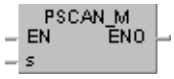
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—

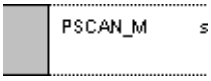
GX IEC Developer

MELSEC Instruction List

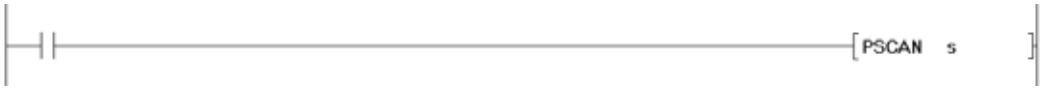
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	File name of program file to be set into scan execution mode or first number of device storing such data	Character string

Functions

Setting a program into the scan execution mode

PSCAN Switch instruction for the scan execution mode

The PSCAN instruction sets the program specified by the device in s into the scan execution mode. In this mode the program is only executed once during one program scan.

Only program files stored in the internal memory (drive 0) can be set into the scan execution mode.

The scan execution mode is only entered after END processing.

The PSCAN instruction is even given priority if the execution mode is specified via parameters.

The file extension .QPG is not needed to be entered for file specification since the type of file is recognized automatically.

Operation Errors

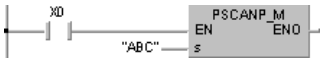
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The specified program file does not exist.
  (Error code 2410)
- The file name storage destination device of s exceeds the range of the corresponding device.
  (Error code 4101)
- The specified file name is an SFC program, and the SFC program for the other file name has been already started. (Dual activation error of the SFC program)
  (For the Universal model QCPU, LCPU: Error code 4131)
  (For the High Performance model QCPU, Process CPU, Redundant CPU: Error code 2504)

Program Example

PSCANP

With leading edge from X0, the following program sets a program named "ABC" into the scan execution mode.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

7.17.4 PLOW, PLOWP

CPU

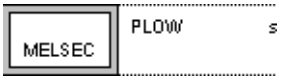
Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●			

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—

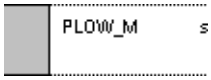
GX IEC Developer

MELSEC Instruction List

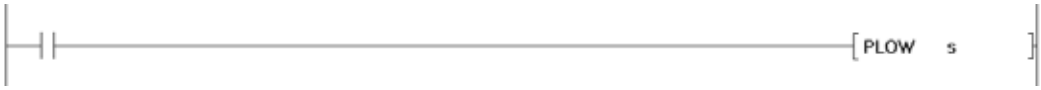


Ladder Diagram

IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
s	File name of program file to be set into low-speed execution mode or first number of device storing such data	Character string

Functions

Setting a program into the low-speed execution mode

PLOW Switch instruction for the low-speed execution mode

The PLOW instruction sets the program specified by the device in s into the low-speed execution mode. In this mode the program is only executed at low processing speed.

Only program files stored in the internal memory (drive 0) can be set into the scan execution mode.

The low-speed execution mode is only entered after END processing.

The PLOW instruction is even given priority if the execution mode is specified via parameters.

The file extension .QPG is not needed to be entered for file specification since the type of file is recognized automatically.

Operation Errors

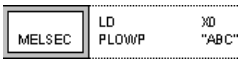
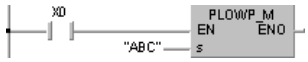
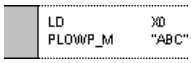
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The specified program file does not exist.
  (Error code 2410)
- The program file contains a CHK instruction.
  (Error code 4235)

Program Example

PLOWP

With leading edge from X0, the following program sets a program named "ABC" into the low-speed execution mode.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

7.17.5 PCHK

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●		

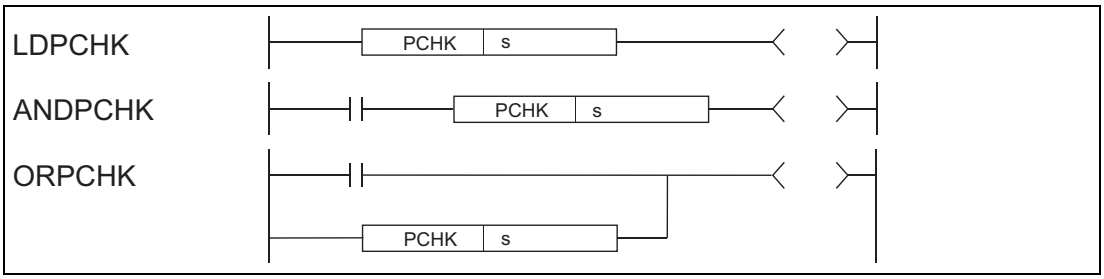
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	—	—	—	—	—	—	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s	File name of the program whose execution status will be checked	Character string

Functions

Program execution status check instruction

PCHK Check instruction for the program execution status

Checks whether the program of the specified file name is in execution or not (non-execution).

The instruction is in conduction when the program of the specified file name is in execution, and the instruction is in non-conduction when the program is in non-execution.

Specify the file name without an extension (.QPG). For example, specify "ABC" when the file name is ABC.QPG.

Operation Errors

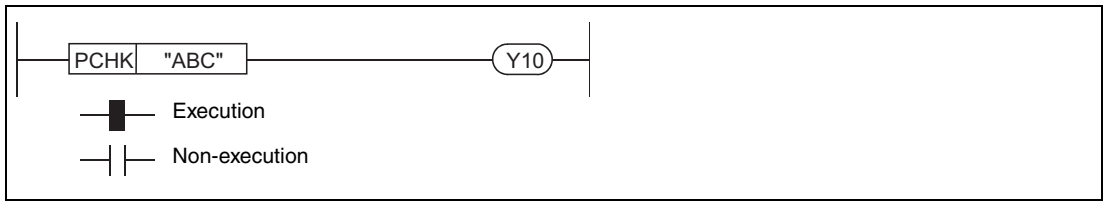
In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The specified program file does not exist.
  (Error code 2410)

Program Example

PCHK

Program that keeps Y10 ON when the program file "ABC.QPG" is being executed.



Non-execution indicates that the program execution type is a stand-by type.

Execution indicates that the program execution type is one of the following:

– a scan execution type (including during output OFF (during non-execution processing))

– a low speed execution type or

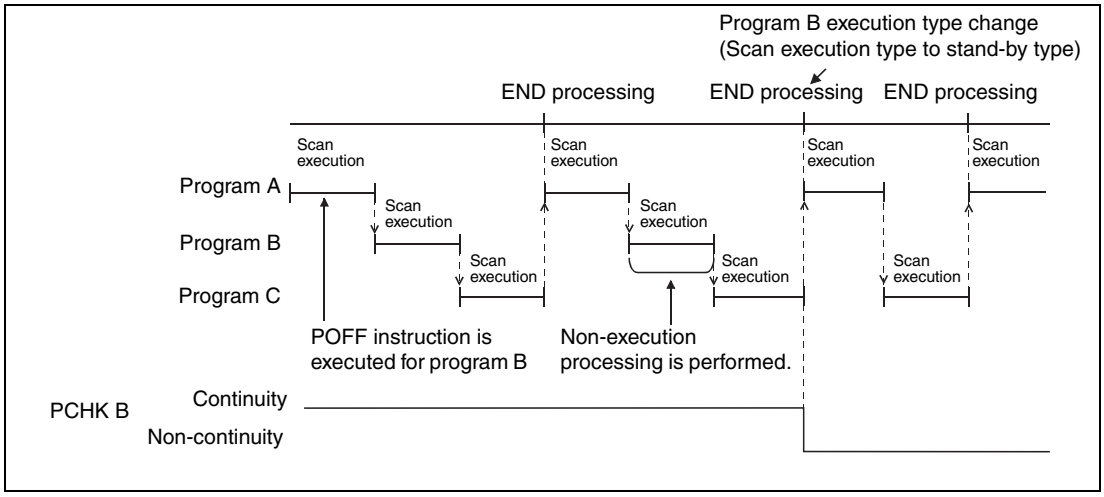
– a fixed scan execution type.

NOTE *The PCHK instruction is in conduction when the program of the specified file name (target program) is in execution, and the instruction is in non-conduction when the program is in non-execution.*

When the target program is set to non-execution (stand-by type) with the POFF instruction, the PCHK instruction is in conduction while the non-execution processing of the target program is being performed. At the END processing of the scan where the non-execution processing is completed, the target program is put into non-execution (stand-by type), and the PCHK instruction is brought into non-conduction.

Therefore, note that if the PCHK instruction is executed for the program where the non-execution processing has been completed by the POFF instruction, the PCHK instruction may be brought into conduction.

The following chart shows the operation performed when program A executes the POFF instruction of program B and program C executes the PCHK instruction of program B with the programs being executed in order of program A, program B and program C.



7.18 Other convenient instructions

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Reset watchdog timer	WDT	WDT_M
	WDTP	WDTP_M
Preset number of execution scans	DUTY	DUTY_M
Time check	TIMCHK	TIMCHK_M
Direct read of one byte	ZRRDB	ZRRDB_M
	ZRRDBP	ZRRDBP_M
Direct write of one byte	ZRWRB	ZRWRB_M
	ZRWRBP	ZRWRBP_M
Store device for indirect designation	ADRSET	ADRSET_M
	ADRSETP	ADRSETP_M
Numerical key input from keyboard	KEY	KEY_MD
Batch save of index register contents	ZPUSH	ZPUSH_M
	ZPUSHP	ZPUSHP_M
Batch recovery of index register contents	ZPOP	ZPOP_M
	ZPOPP	ZPOPP_M
Reading module information	UNIRD	UNIRD_M
	UNIRDP	UNIRDP_M
Reading module model name	TYPERD	
	TYPERDP	
Trace set	TRACE	TRACE_M
Trace reset	TRACER	TRACER_M
Writing data to a designated file	SP.FWRITE	
Reading data from a designated file	SP.FREAD	
Writing data to standard ROM	SP.DEVST	
Reading data from standard ROM	S.DEVLD	
	SP.DEVLD	
Loading program from memory	PLOADP	PLOADP_M
Unloading program from program memory	PUNLOADP	PUNLOADP_M
Load and unload	PSWAPP	PSWAPP_M
Highspeed block transfer of file register	RBMOV	RBMOV_M
	RBMOVP	RBMOVP_M
User message	UMSG	UMSG_M

NOTE *The instructions ADRSET and ADRSETP are not supported by the GX IEC Developer.*

7.18.1 WDT, WDTP

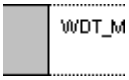
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

GX Works2

Variables

Set Data	Meaning	Data Type
—	—	—

Functions

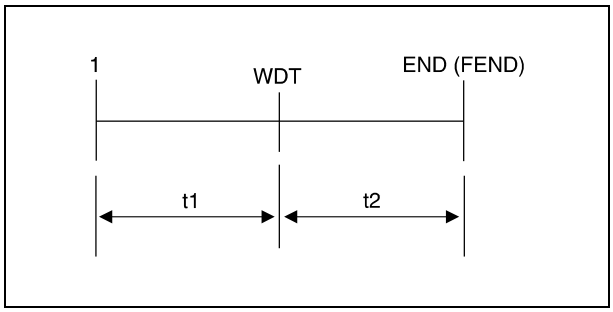
Resetting the watchdog timer

WDT Reset

The WDT instruction resets the watchdog timer (WDT) during execution of a sequence program.

The WDT instruction is only needed, if the program scan time of a sequence program from program step 0 up to the END/FEND instruction exceeds the default time setting of the WDT under certain conditions. If the default time setting of the WDT is exceeded any program scan the parameter setting of the WDT has to be adjusted accordingly.

The setting value of the WDT has to be adjusted so that neither the time period t1 (step 0 to WDT instruction) nor t2 (WDT and END/FEND instructions) exceed the WDT setting value.



[1]Step 0

The WDT instruction can be set any number of times within one program scan. Nevertheless, for programming remind that the outputs are not reset (0) at once.

The values of the program scan time stored in the registers are not cleared via the WDT instruction. Therefore, the stored values may be greater than the WDT values set through parameters.

7.18.2 DUTY

CPU

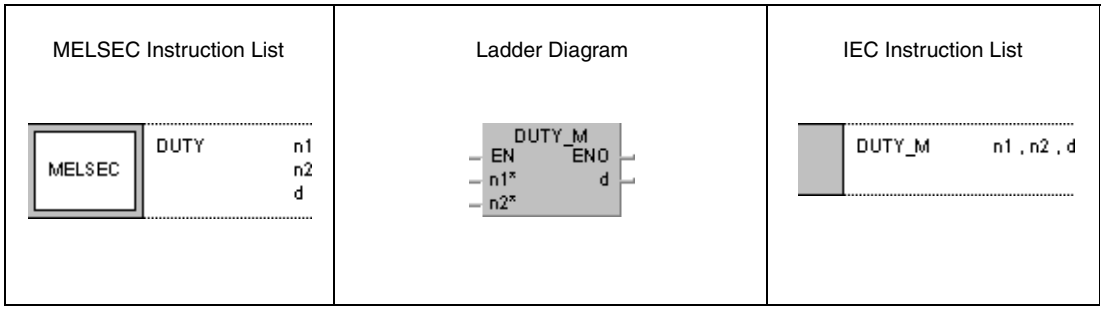
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

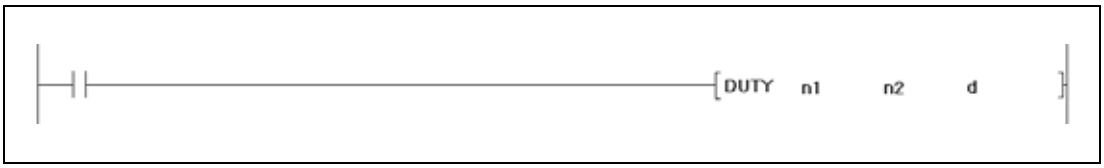
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
n1	●	●	●	●	●	●	●	●	●
n2	●	●	●	●	●	●	●	●	●
d	●[1)]	—	—	—	—	—	—	—	—

[1] SM420 through SM424 and SM430 through SM434

GX IEC Developer



GX Works2



Variables

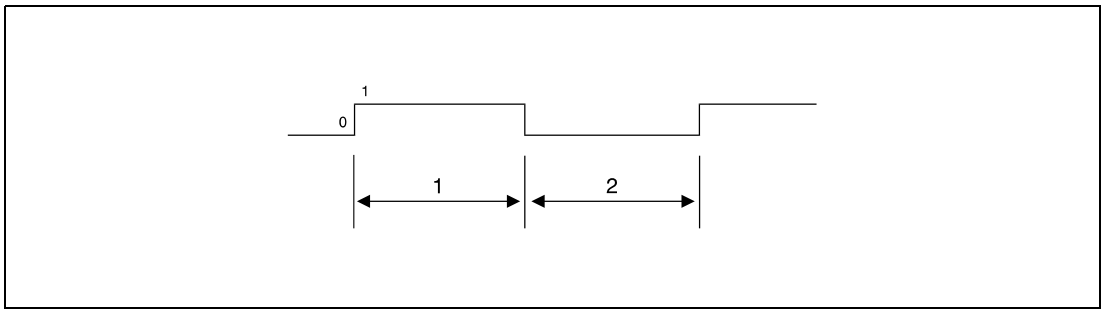
Set Data	Meaning	Data Type
n1	Number of scans the special relays are set	BIN 16-bit
n2	Number of scans the special relays are reset	
d	Address of special relay SM420–SM424 and SM430–SM434)	Bit

Functions

Presetting the number of execution scans of a device

DUTY Preset execution scans

The DUTY instruction turns the devices specified by d (SM420 through SM424 and SM430 through SM434) ON for the number of program scans specified by n1 and OFF for the number of program scans specified by n2. The according special relay serves as input condition for following operations.



[1] Number of program scans with execution

[2] Number of program scans without execution

Programs being executed once per program scan apply the relays SM420 through SM424.

Low-speed execution programs apply the relays SM430 through SM434.

At the beginning of the execution (initializing) the relays (SM420 through SM424 and SM430 through SM434) are reset.

If the value in n1 = 0, the relays remain reset.

If the value in n2 = 0 and the value in n1 is greater than 0, the relays will be and remain set.

The values in n1, n2, and d are set when the DUTY instruction is invoked. The scan pulse (relay) is set ON or OFF when the END instruction is reached.

Operation Errors

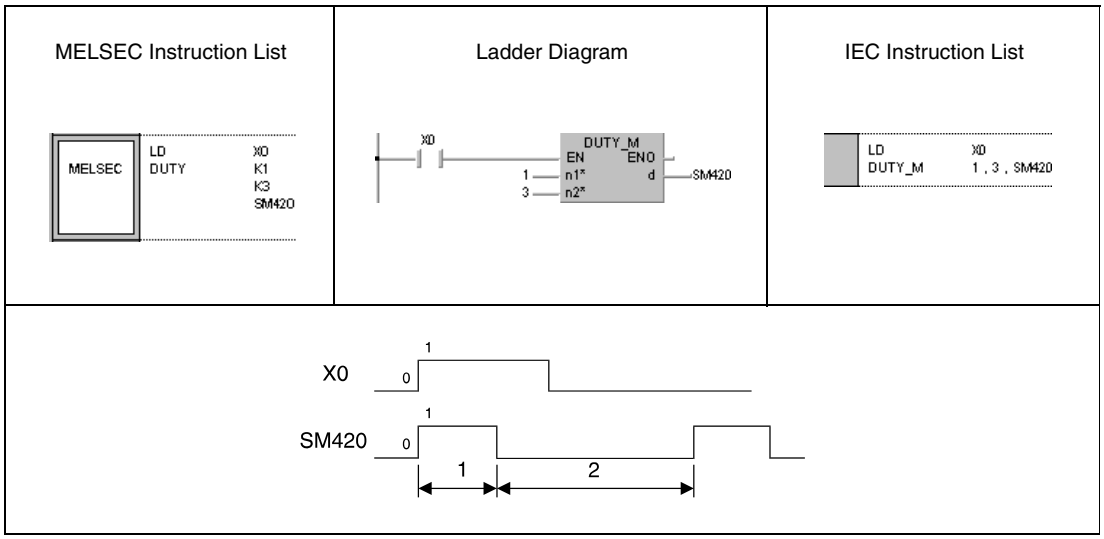
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by d is not from SM420 to SM424 or SM430 to SM434. (Error code 4101)
- The values in n1 and n2 are less than 0. (Error code 4100)

Program Example

DUTY

With leading edge from X0, the following program sets SM420 for one program scan and resets it for 3 program scans. This operations are repeated as long as the program is executed (see NOTE below).

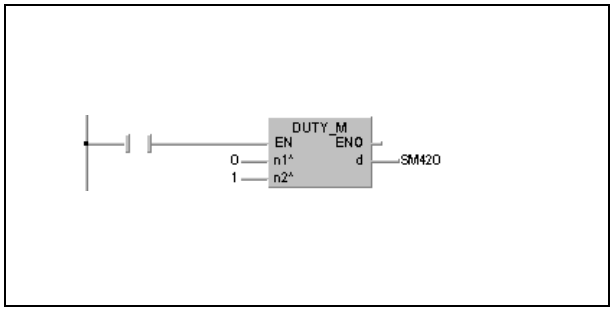


[1] One program scan ON

[2] Three program scans OFF

NOTE

After the execution condition is reset (X0 = OFF) the output of scan pulse of the DUTY instruction and the cyclic setting / resetting of the specified relay are proceeded. In order to stop the continued output of scan pulses the following program part has to be inserted.



7.18.3 TIMCHK

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●	●	●	●	●

[1] Basic model QCPU: The first five digits of the serial No. are "04122" or higher.

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	●	●	●	●	●	●	●	●	—
d	●	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2

Variables

Set Data	Meaning	Data Type
s1	Device where the measured current value will be stored	BIN 16-bit
s2	Set value or device where the set value of measurement is stored	
d	Device to be turned ON at time-out	Bit

Functions

Time check instruction

TIMCHK Time check instruction

Measures the ON time of the device used as a condition, and turns ON the device specified by d if the condition device remains ON for longer than the time set to the device specified by s2.

The current value of the device specified by s1 is cleared to 0 and the device specified by d is turned OFF at the leading edge of the execution command.

The current value of the device designated by s1 and the ON status of the device designated by d are retained after the execution command turns OFF.

Set the set value of measurement in units of 100 ms.

Operation Errors

In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device that cannot be specified has been specified.
  (Error code 4100)

Program Example

TIMCHK

Program where the ON time of X0 is set to 5 s, the current value storage device to D0, and the device that will turn ON at time-out to Y10.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

7.18.4 ZRRDB, ZRRDBP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

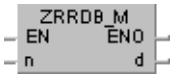
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
n	●	●	●	●	●	●	●	●	—
d	●	●	●	●	●	●	●	—	—

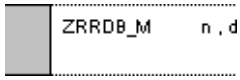
GX IEC Developer

MELSEC Instruction List

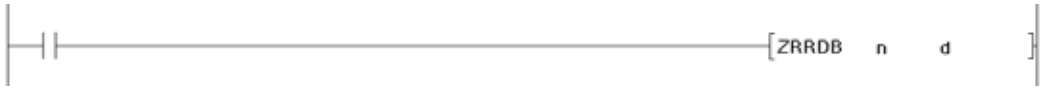
Ladder Diagram



IEC Instruction List



GX Works2



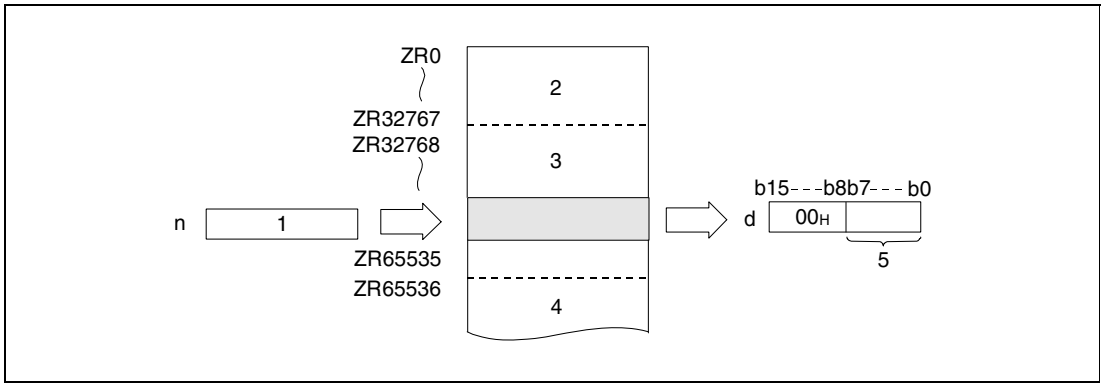
Variables

Set Data	Meaning	Data Type
n	Serial byte number for file register to be read	BIN 32-bit
d	Number of device storing the read byte	BIN 16-bit

Functions **Direct read of one byte from a file register**

ZRRDB Read one byte

The ZRRDB instruction reads one byte specified by n via the serial byte number from a file register. The byte number does not specify a block address. The byte is stored in the lower byte of the device specified by d. The upper byte in the device specified by d stores the value "00H".



[1] Serial byte number

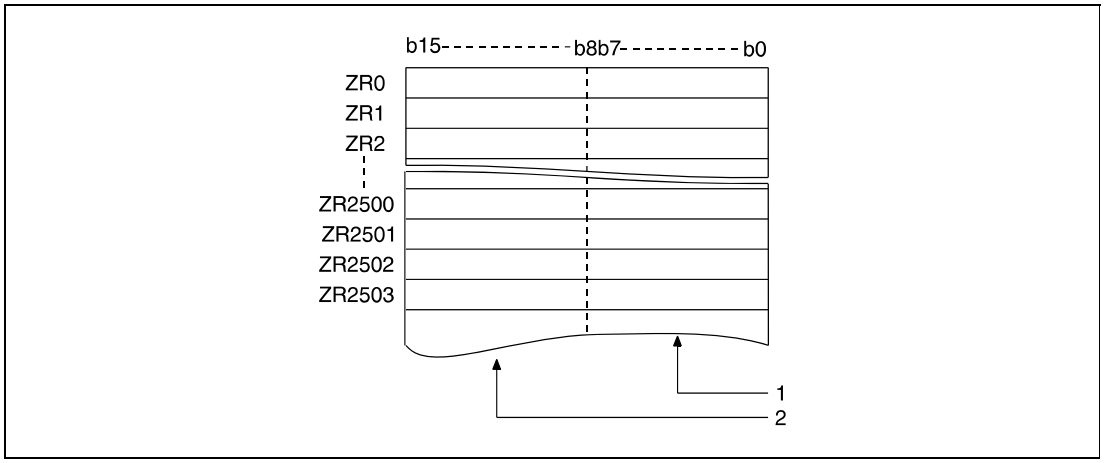
[2] File register area for block 0

[3] File register area for block 1

[4] File register area for block 2

[5] Read byte

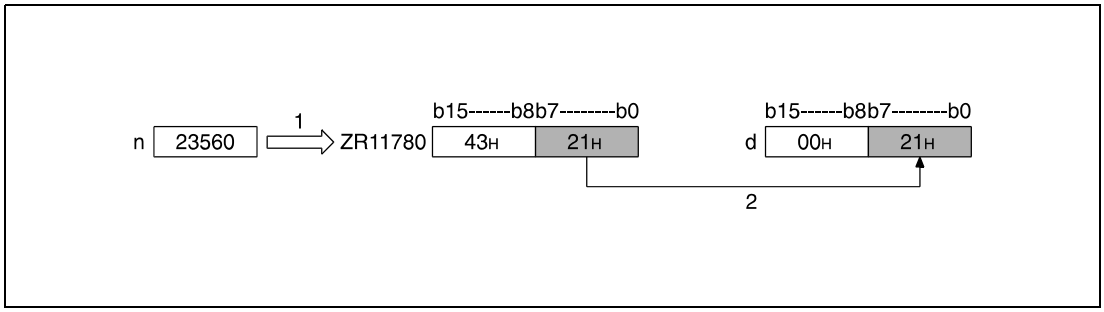
The assignment of file register numbers to the according serial byte numbers is shown below:



[1] Storage area for even byte numbers (here: address 0 through address 5006)

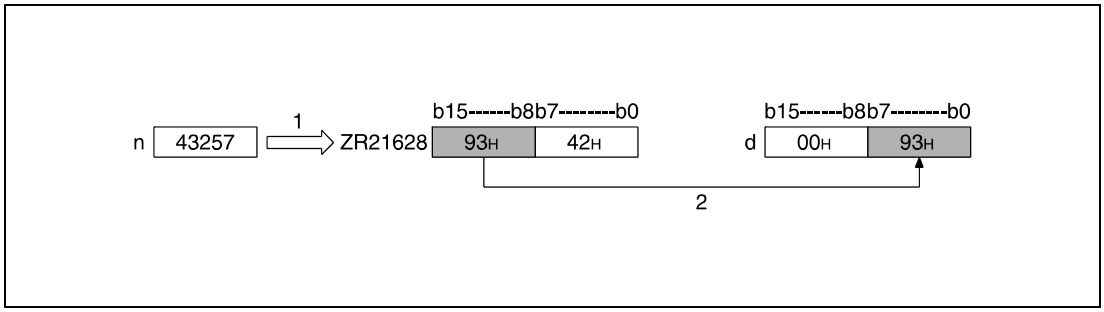
[2] Storage area for odd byte numbers (here: address 1 through address 5007)

If the byte number 23560 is specified, the lower byte of the file register ZR11780 is read.



[1] Address
[2] Storage

If the byte number 43257 is specified, the lower byte of the file register ZR21628 is read.



[1] Address
[2] Storage

Operation Errors

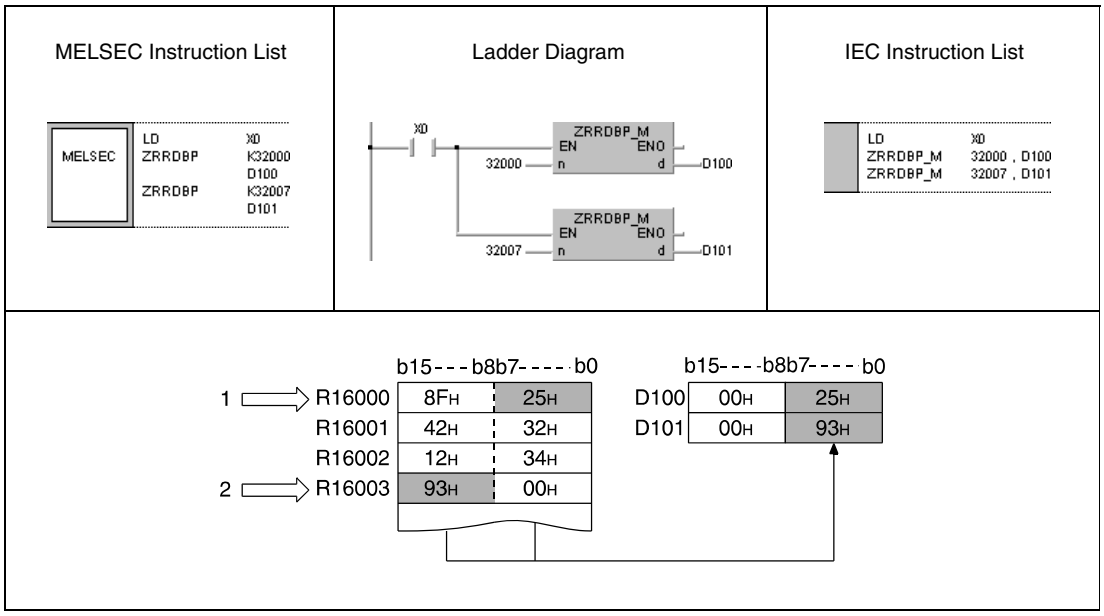
In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of device (serial byte address) exceeds the relevant storage device range. (Error code 4101)

Program Example

ZRRDBP

With leading edge from X0, the following program reads the lower byte of file registers R16000 (byte number 32000) and the upper byte of the file register R16003 (byte number 32007). The bytes are stored in D100 and D101.



[1] Serial byte number 32000 (lower byte in file register R16000)

[2] Serial byte number 32007 (upper byte in file register R16003)

7.18.5 ZRWRB, ZRWRBP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

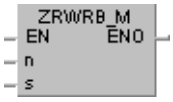
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
n	●	●	●	●	●	●	●	●	—
s	●	●	●	●	●	●	●	●	—

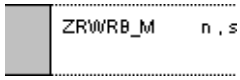
GX IEC Developer

MELSEC Instruction List

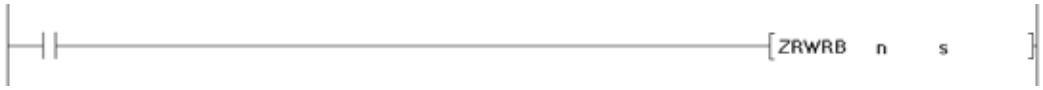
Ladder Diagram



IEC Instruction List



GX Works2



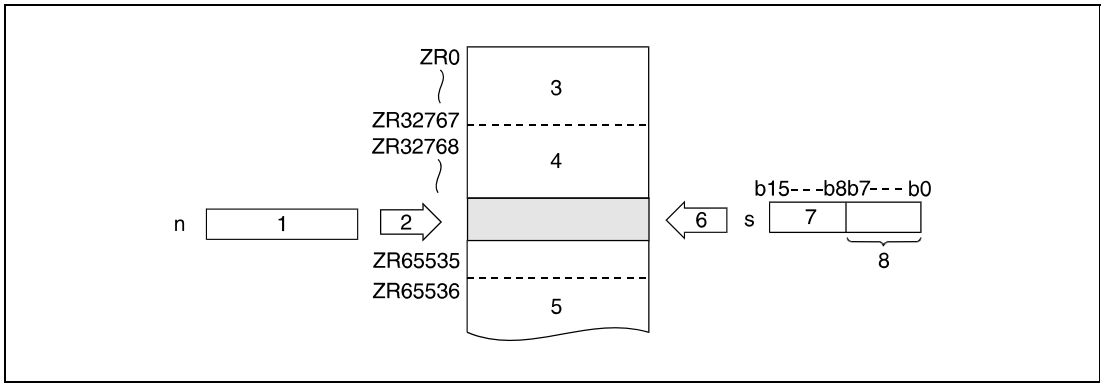
Variables

Set Data	Meaning	Data Type
n	Serial byte number in file register to be written to	BIN 32-bit
s	Device storing data to be written	BIN 16-bit

Functions **Direct write of one byte to a file register**

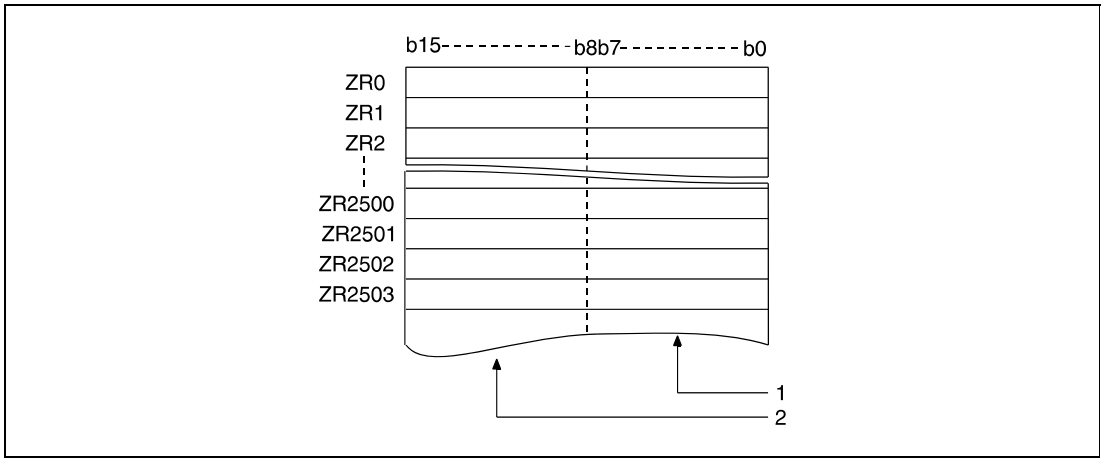
ZRWRB Write one byte

The ZRRDB instruction writes the contents of the lower byte in the device specified by s to the file register specified by n via serial byte number. The byte number in s does not specify a block address. The upper byte of the device in s is ignored.



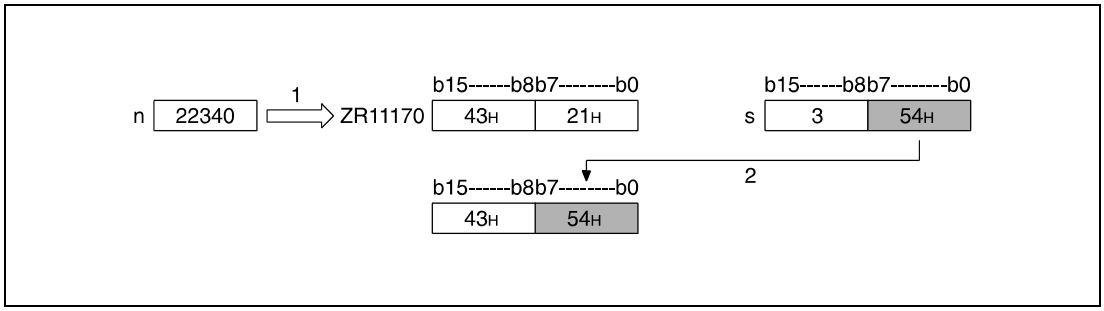
[1] Serial byte number
[2] Address
[3] File register area for block 0
[4] File register area for block 1
[5] File register area for block 2
[6] Write data
[7] This byte is ignored
[8] Byte to be written

The assignment of file register numbers to the according serial byte numbers is shown below:



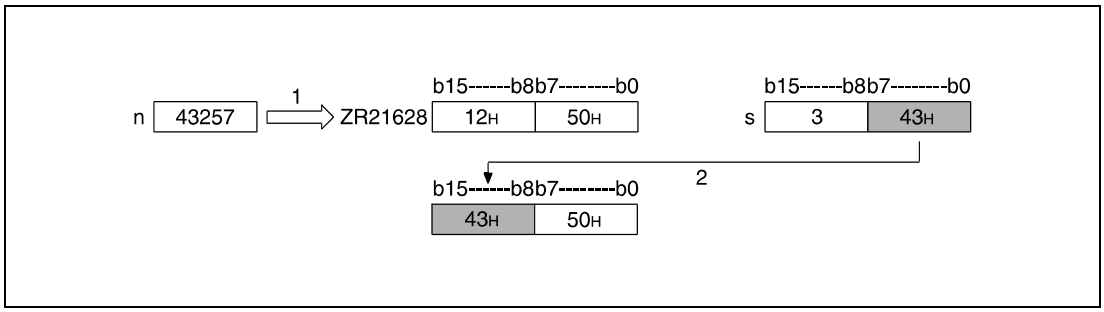
[1] Storage area for even byte numbers (here: address 0 through address 5006)
[2] Storage area for odd byte numbers (here: address 1 through address 5007)

If the byte number 22340 is specified, the lower byte of the device specified by s is written to the lower byte of the file register ZR11170.



[1] Address
[2] Write byte
[3] This byte is ignored.

If the byte number 43257 is specified, the lower byte of the device specified by s is written to the upper byte of the file register ZR21628.



[1] Address
[2] Write byte
[3] This byte is ignored.

Operation Errors

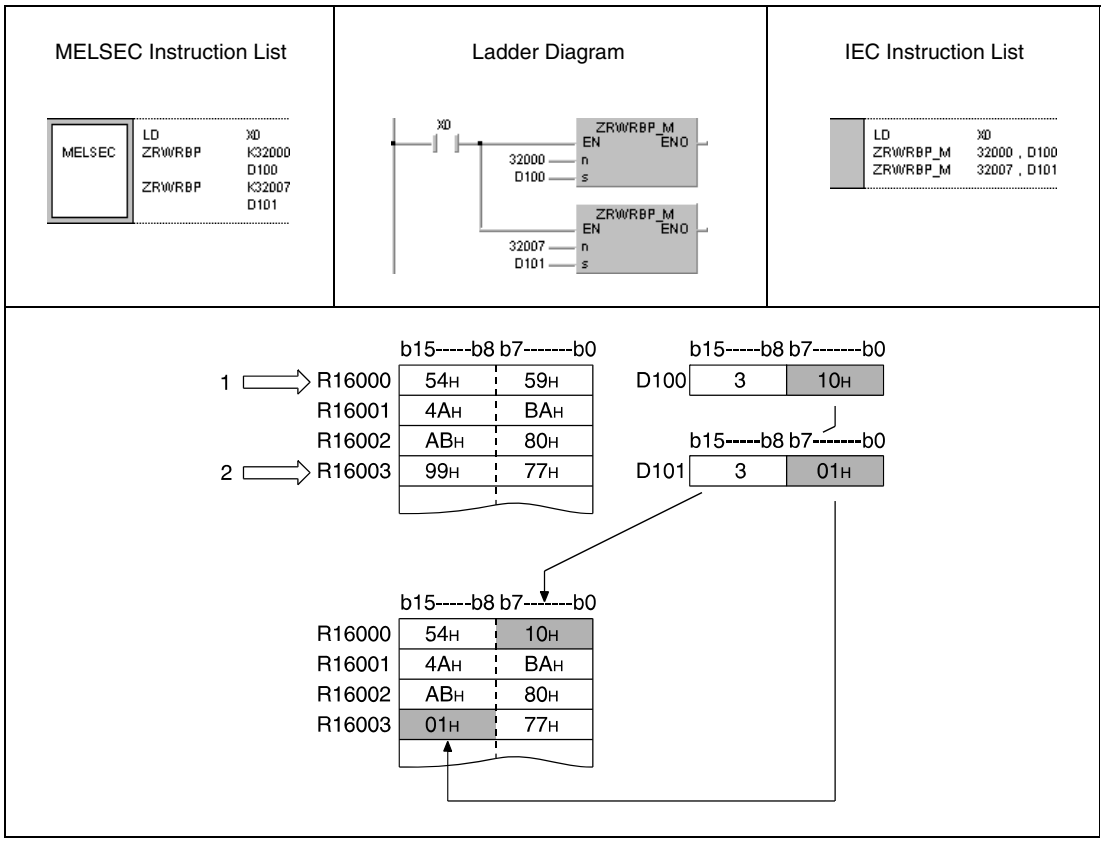
In the following case an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The number of device (serial byte number) specified by n exceeds the relevant storage device range.
  (Error code 4101)

Program Example

ZRWRBP

With leading edge from X0, the following program writes the contents of the lower bytes of the registers D100 and D101 to the lower byte of the file register R16000 (byte number 32000) and to the upper byte of the file register R16003 (byte number 32007).



[1] Serial byte number 32000 (lower byte of file register R16000)

[2] Serial byte number 32007 (upper byte of file register R16003)

[3] These bytes are ignored.

7.18.6 ADRSET, ADRSETP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	—	—	—	—	—	—
d	●	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
———	———	———

NOTE *The instructions ADRSET and ADRSETP are not supported by the GX IEC Developer.*

GX Works2

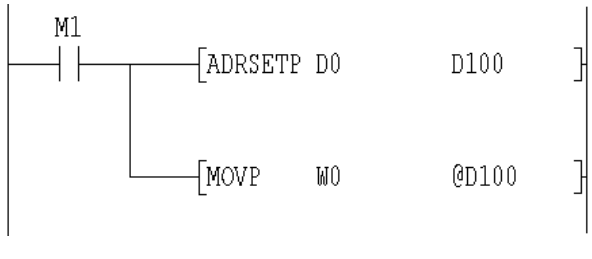
Variables

Set Data	Meaning	Data Type
s	Number of device for indirect address read	Device name
d	Number of device that will store the indirect address of the device designated by s	BIN 32-bit

Functions **Indirect address read operations**

ADRSET Stores the indirect adress

Stores the indirect adress of the device designated by s at d and d + 1. The adress stored at the device designated by d is used when reading of an indirect device adress is performed by the sequence program. A bit device designation cannot be made at s.



The indirect adress of D0 is stored at D100 and D101.

The data stored at W0 is stored at D0 by indirect setting.
(The device which contains the indirect adress is prefixed by „@“)

7.18.7 KEY

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●			

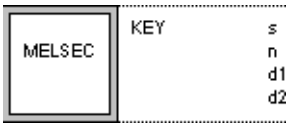
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●[1)]	—	—	—	—	—	—	—	—
n	●	●	●	●	●	●	●	●	—
d1	—	●	●	—	—	—	—	—	—
d2	●	●	●	●	●	●	—	—	—

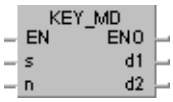
[1] X only

GX IEC Developer

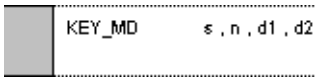
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

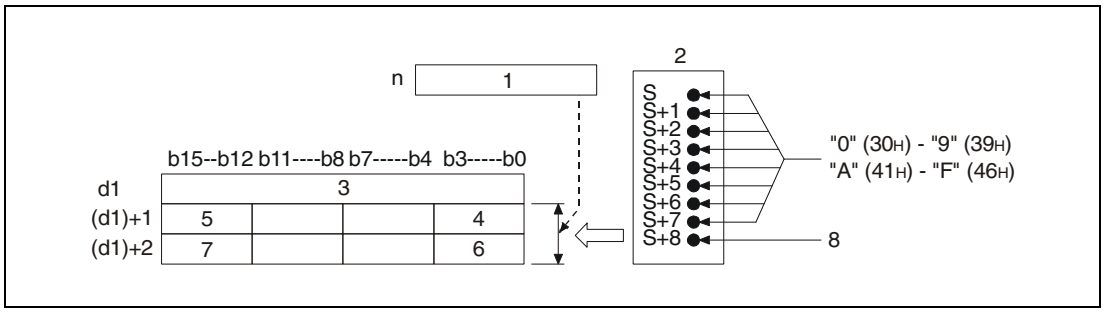
Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
s	First number of devices (X), receiving numerical key input	Bit	Array [0..8] of BOOL
n	Number of digits to be input	BIN 16-bit	ANY16
d1	First number of device storing numerical key input	BIN 16-bit	Array [0..2] of ANY16
d2	Number of bit device to be set after completion of key input	Bit	BOOL

Functions **Numerical key input**

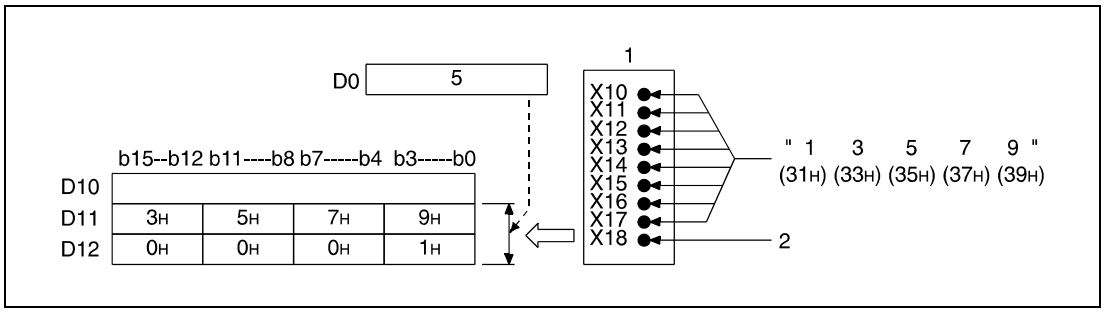
KEY Input instruction

The KEY instruction supports the key input of the ASCII characters 0 (30H) through 9 (39H) and A (41H) through F (46H) at the inputs specified by s+0 (Array_s[0]) through s+7 (Array_s[7]). The values entered at the inputs are encoded in hexadecimal format and stored in the devices specified by (d1)+0 (Array_d1[0]) through (d1)+2 (Array_d1[2]). The number of characters to be input is specified by n.



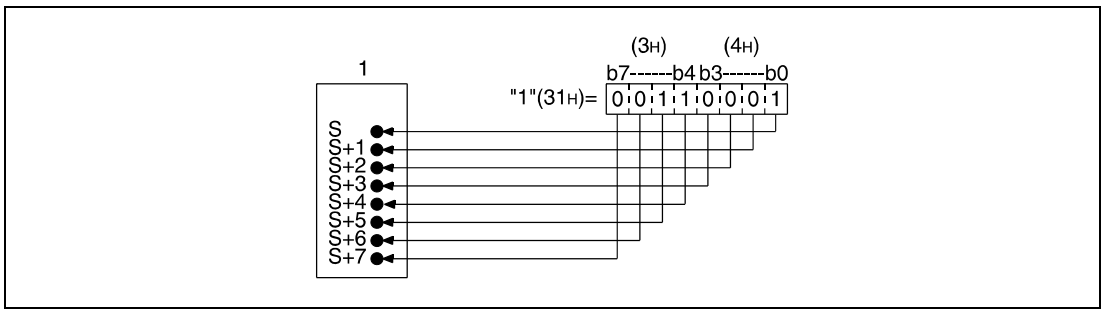
[1] Number of values to be entered
[2] Input module
[3] Number of entered values
[4] 8th entered character
[5] 5th entered character
[6] 4th entered character
[7] 1st entered character
[8] Strobe signal

In the following diagram n is specified 5 and the values 1 (31H) through 5 (35H) are entered at the inputs X10 through X18 of the input module.



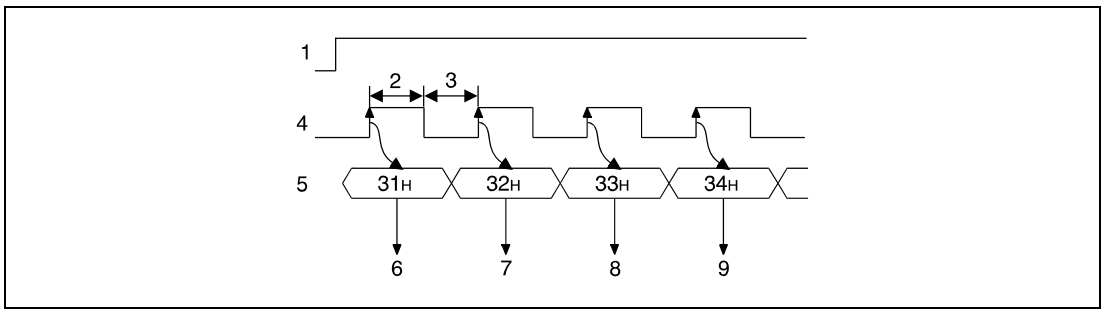
[1] Input module
[2] Strobe signal

The ASCII characters entered at the inputs (X) specified in s+0 (Array_s[0]) through s+7 (Array_s[7]) are encoded in 8-bit binary format as illustrated below:



[1] Input module

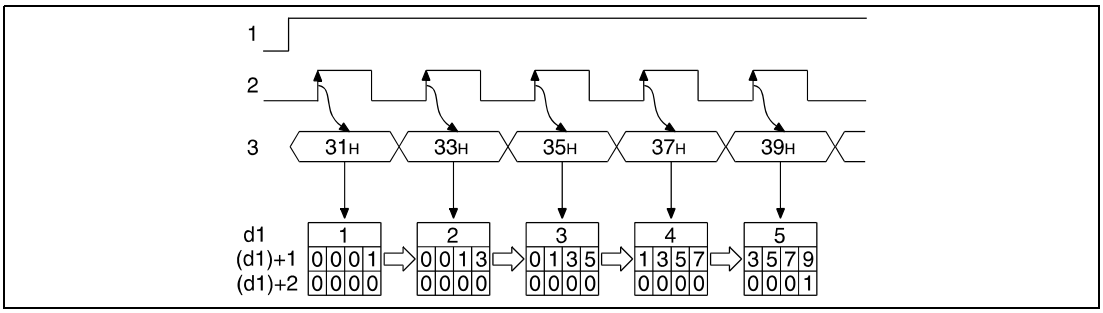
After the input of an ASCII character at s+0 (Array_s[0]) through s+7 (Array_s[7]) the strobe signal (s+8, Array_s[8]) is set, to link the input data internally. The time period the strobe signal remains set or reset must exceed one program scan time to ensure accurate linking of input data.



[1] Execution condition for the KEY instruction
[2] Set for more than one program scan
[3] Reset for more than one program scan
[4] Strobe signal (s+8, Array_s[8])
[5] ASCII input data (s+0 through s+7, Array_s[0] through Array_s[7])
[6] Reading "1"
[7] Reading "2"
[8] Reading "3"
[9] Reading "4"

The KEY instruction can only be executed with the execution condition set. The execution condition must remain set until the input of the number of characters specified by n is completed.

The number of entered values is stored in (d1)+0 (Array_d[0]). The entered ASCII characters are actually stored in the devices specified in (d1)+1 (Array_d[1]) and (d1)+2 (Array_d[2]) and (d1)+2 (Array_d[2]) as hexadecimal binary values; i.e. there are 4 bits per character supplied. The hexadecimal binary values of the characters 0H through FH range from "0000" through "1111".



[1] Execution condition for the KEY instruction

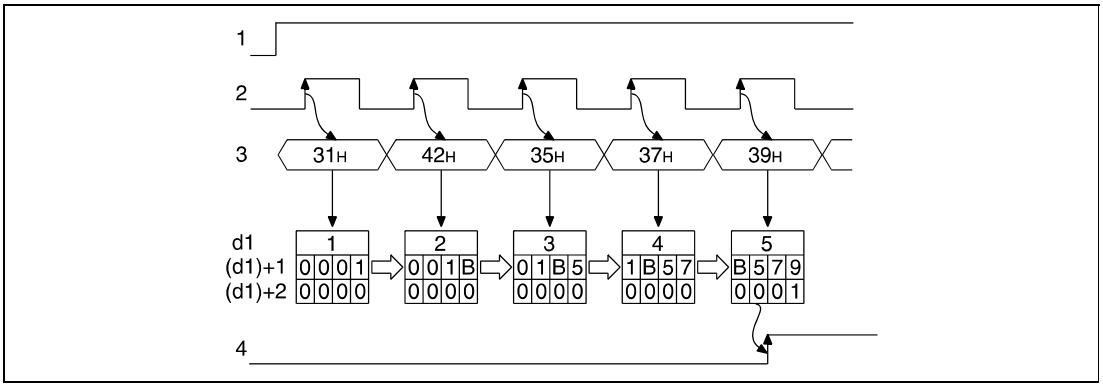
[2] Strobe signal (s+8, Array_s[8])

[3] ASCII input data (s+0 through s+7, Array_s[0] through Array_s[7])

The number of characters to be entered specified by n must range within 1 and 8.

If the specified number of characters or the character code "00H" are entered, the linking of the input data is completed and the device specified by d2 is set. The following diagrams illustrate these operations. For n 5 is specified.

In the following diagram the input is completed after 5 characters. In the next but one diagram the input is completed after the character code "00H".

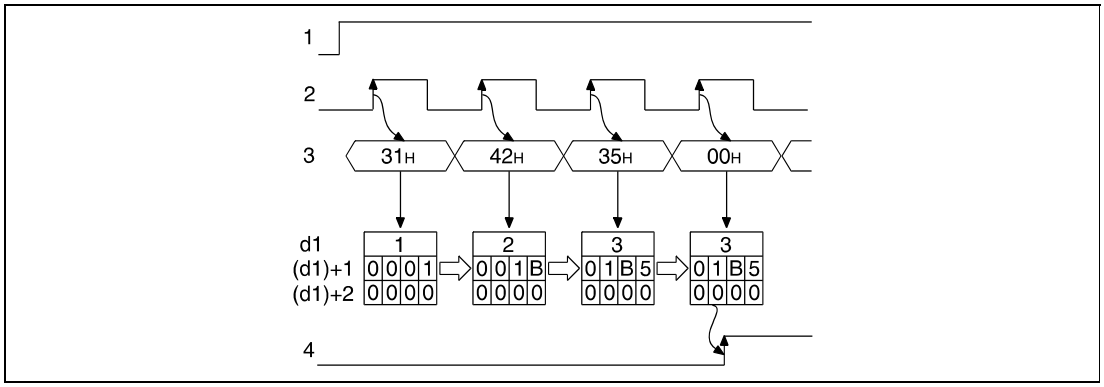


[1] Execution condition for the KEY instruction

[2] Strobe signal (s+8, Array_s[8])

[3] ASCII input data (s+0 through s+7, Array_s[0] through Array_s[7])

[4] Input of characters completed (the device specified by d2 is set)



[1] Execution condition for the KEY instruction

[2] Strobe signal (s+8, Array_s[8])

[3] ASCII input data (s+0 through s+7, Array_s[0] through Array_s[7])

[4] Input of characters completed (the device specified by d2 is set)

Prior to a new input of characters the contents of the devices specified in (d1)+0 (Array_d[0]) through (d1)+2 (Array_d[2]) have to be cleared and the device specified by d2 has to be reset; otherwise a new input of characters is not possible.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by s is not an input (X).
  (Error code 4100)
- The number of characters specified by n does not range within 1 and 8.
  (Error code 4100)

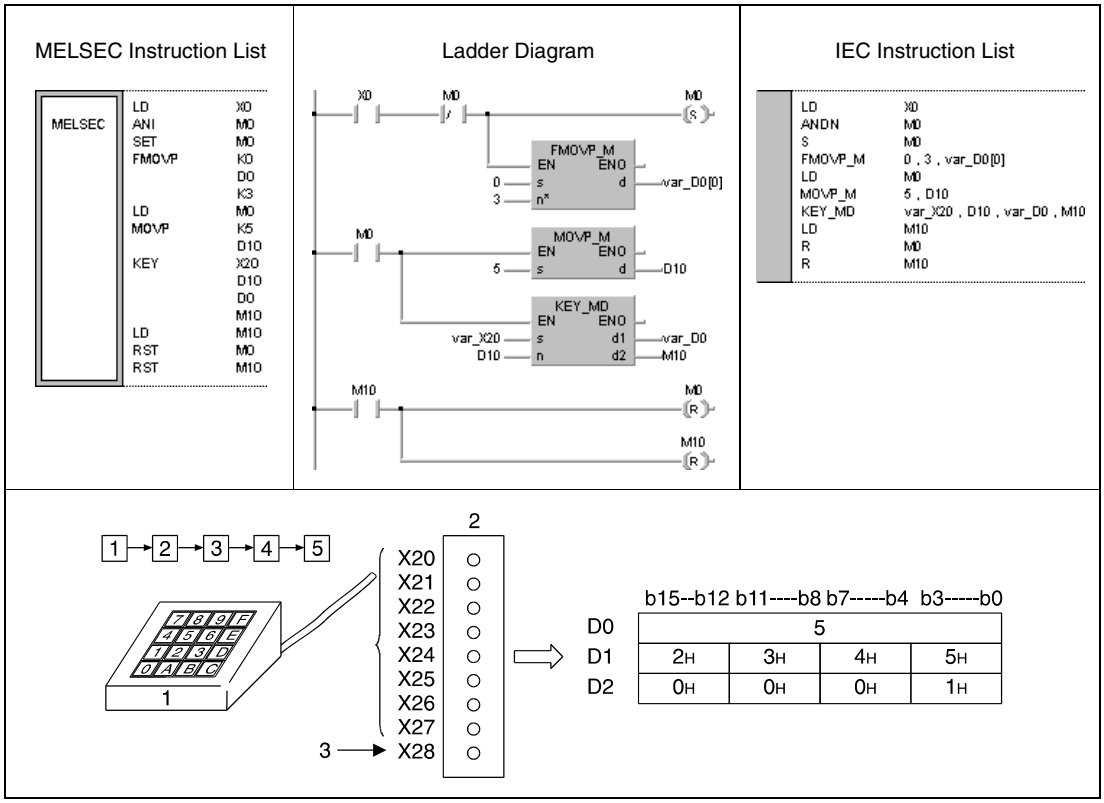
Program Example

KEY

The following program enables key input of up to 5 numerical values via the inputs X20 (var_X20[0]) through X27 (var_X20[7]).

The values are stored in the registers D1 (var_D0[1]) and D2 (var_D0[2]) binary coded in hexadecimal format. The number of values already entered is stored in D0 (var_D0[0]).

Prior to the execution of the KEY instruction the registers D0 (var_D0[0]) through D2 (var_D0[2]) are cleared and the number of input values (5) is stored. After execution of the KEY instruction the relay M10 (input completed) is reset. The strobe signal is supplied at the inputs X28 (var_X20[8]).



[1] Numerical key pad

[2] Input module

[3] Strobe signal

7.18.8 ZPUSH, ZPUSHP, ZPOP, ZPOPP

CPU

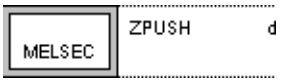
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

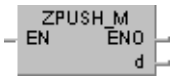
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

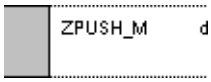
MELSEC Instruction List



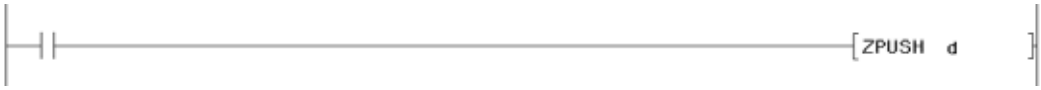
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type
d	First number of device storing index register contents	BIN 16-bit

Functions **Batch save and batch recovery of index register contents**

ZPUSH Batch save of index register contents

The ZPUSH instruction saves the contents of the following index registers in the devices specified from d on. (When contents of an index register are saved, d+0 (the number of saves made) is increased by 1.)

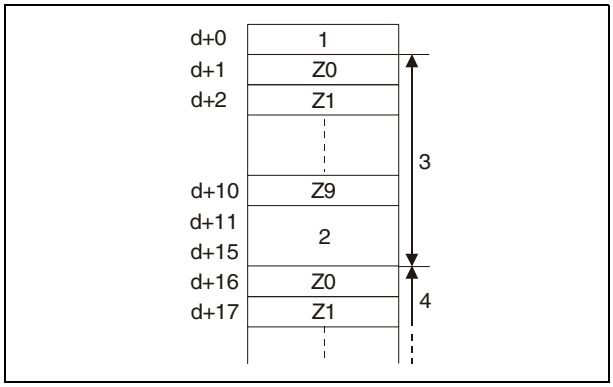
- Basic model QCPU: Z0 to Z9
- High Performance model QCPU/Process CPU/Redundant CPU: Z0 to Z15
- Universal model QCPU/LCPU: Z0 to Z19

These data can be recovered via the ZPOP instruction. The instruction can be applied to different nestings that are included in ZPUSH / ZPOP loop.

On execution of the instructions in different nestings each execution of the ZPUSH instruction requires an area of 18 registers with 16 bits in the devices specified from d on. Therefore, for the execution of the ZPUSH instruction the according amount of storage area has to be available.

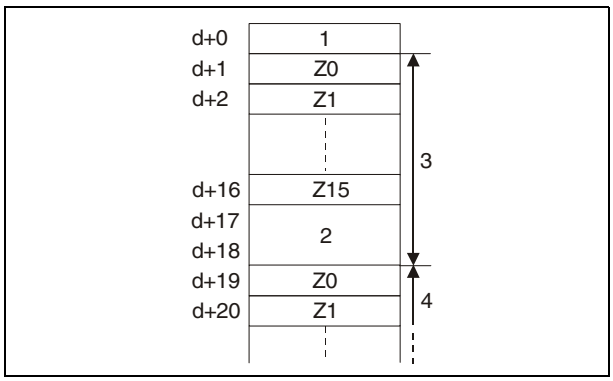
The following diagrams illustrate the organization of the storage area from d on:

- When using a Basic model QCPU



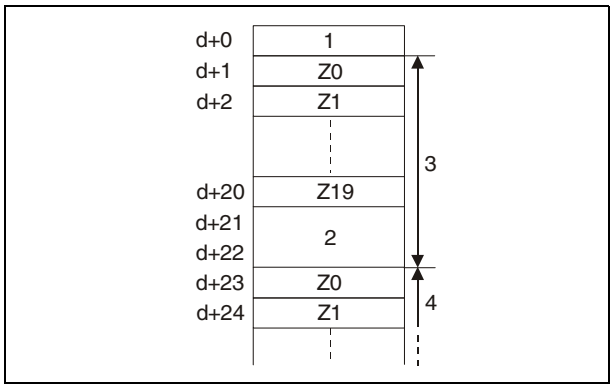
[1]Number of saved register contents
[2]Five data words (internal system use)
[3]First nesting level (15 data words max.)
[4]Second nesting level

- When using a High Performance model QCPU/Process CPU/Redundant CPU



[1]Number of saved register contents
[2]Two data words (internal system use)
[3]First nesting level (18 data words max.)
[4]Second nesting level

● When using Universal model QCPU/LCPU



[1]Number of saved register contents
[2]Two data words (internal system use)
[3]First nesting level (22 data words max.)
[4]Second nesting level

ZPOP Batch recovery of index register contents

The ZPOP instruction recovers the contents saved in the area starting from the device designated by d to the index register.

When the saved content is read out to the index register, d+ 0 (the number of saves made) is decreased by 1.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

● The storage area specified from d on exceeds the relevant storage device range.
(Error code 4101)

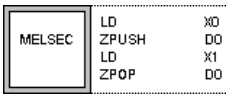
● The content of the device specified in d+0 (number of saved registers is 0 in the ZPOP(P) instruction.).
(Error code 4100)

Program Example

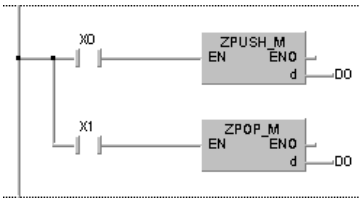
ZPUSH/ZPOP

The following program saves the contents of the index register to the fields following D0 before calling the subroutine following P0 that uses the index register.

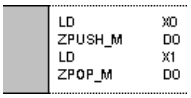
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



7.18.9 UNIRD, UNIRDP

CPU

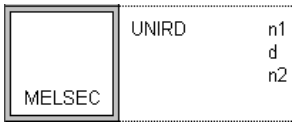
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

Devices

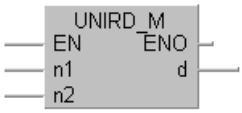
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
n1	●	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n2	●	●	●	—	—	—	—	●	—

GX IEC Developer

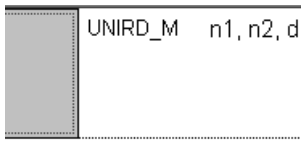
MELSEC Instruction List



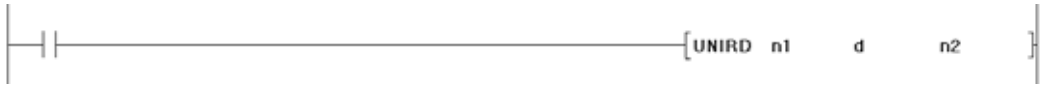
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Device	Meaning	Data Type
n1	Value obtained by dividing the head I/0 number of the module from which module information is read by 16 (0 to FFH)	BIN 16-Bit
d	Head number of the device which stores module information	Device name
n2	Number of points of read data (0 to 256)	BIN 16-Bit

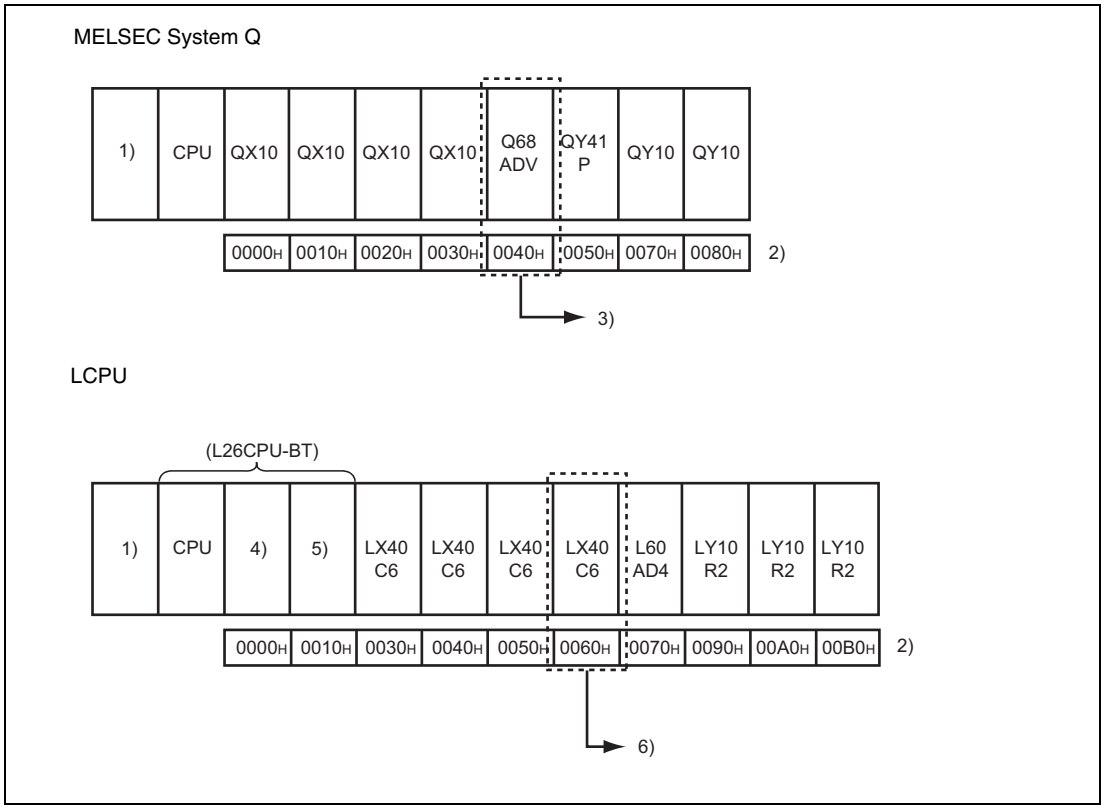
Functions **Reading module information**

UNIRD Read instruction

The UNIRD instruction reads the module information starting at the head I/O address, which is specified by n1 and stores the data at the address which is specified by d. The number of points is specified by n2. The value for n1 is calculated by dividing the head I/O number of the module by 16.

With the UNIRD instruction it is possible to read the statuses of the actually installed modules instead of the module type designated by I/0 assignment.

NOTE *The value of n1 is consists of the higher three digits of the head I/O number of the slot from which the module information is read. The head I/O number is expressed in 4 digits in hexadecimal notation.*



[1] Power supply module

[2] Head I/O number configured in the I/O assignment setting

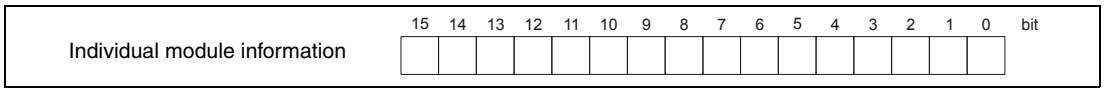
[3] Head address in n1: K4 or H4

[4] Built-in I/0

[5] Built-in CC-Link

[6] Head address in n1: K6 or H6

The details of the module information are described as follows:



Bit	Item	Meaning	
		QCPU	LCPU
0 1 2	Number of I/O points	000: 16 010: 48 100: 128 110: 512	001: 32 011: 64 101: 256 111: 1024
3 4 5	Module type	000: Input module 001: Output module 010: I/O mixed module 011: Intelligent function module	000: Input module 001: Output module 011: Intelligent function module 111: CPU Built-in I/O
6	External power supply status (For future expansion)	1: External power supply is connected 0: External power supply is not connected	Fixed to 0
7	Fuse status	1: Blown fuse 0: Normal, no blown fuse	Fixed to 0
8	Online module replacement status/ execution from the standby system	1: Module information on the extension base unit is tried to be read during online module change or from the CPU module of standby system in the redundant system.[1)] 0: Other than above	Fixed to 0
9	Light/medium error status	1: Light/medium error has occurred	0: Normal
10 11	Module error status	00: No module error 10: Medium error	01: Light error 11: Serious error
12	Module standby status	1: Normal	0: Module error occurred
13	Vacant	Fixed to 0	
14	A-/Q-Module	1: The module is an A-series module 0: The module is a System Q module	Fixed to 0
15	Module installation status	1: Modules are installed	0: No modules are installed

[1] The Universal model QCPU used in the multiple CPU system is turned ON during the online module change of the module controlled by the other CPU.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- High Performance model QCPU, Process CPU, Redundant CPU and Universal model QCPU, L26CPU-BT:
  When n1 is other than 0 to FFH. (Error code 4100)
  When n2 is other than 0 to 256. (Error code 4100)
  When a total of n1 and n2 is greater than 256. (Error code 4100)
- Q00/Q01CPU/L02CPU:
  When n1 is other than 0 to 3FH. (Error code 4100)
  When n2 is other than 0 to 64. (Error code 4100)
  When a total of n1 and n2 is greater than 64. (Error code 4100)
- Q00JCPU:
  When n1 is other than 0 to FH. (Error code 4100)
  When n2 is other than 0 to 16. (Error code 4100)
  When n1 and n2 is greater than 16. (Error code 4100)
- MELSEC System Q CPU/LCPU:
  When the number of points specified by n2 for the devices specified in (d) and up is outside the range of that device.
  (Error code 4101)

Program Example

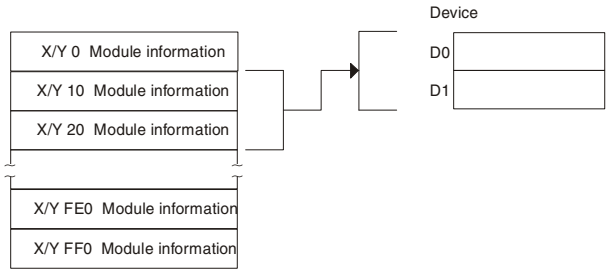
UNIRD

The following program stores the informations of the modules with the head I/O numbers 10H through 20H to D0 and D1, when X10 is turned ON.

Instruction List

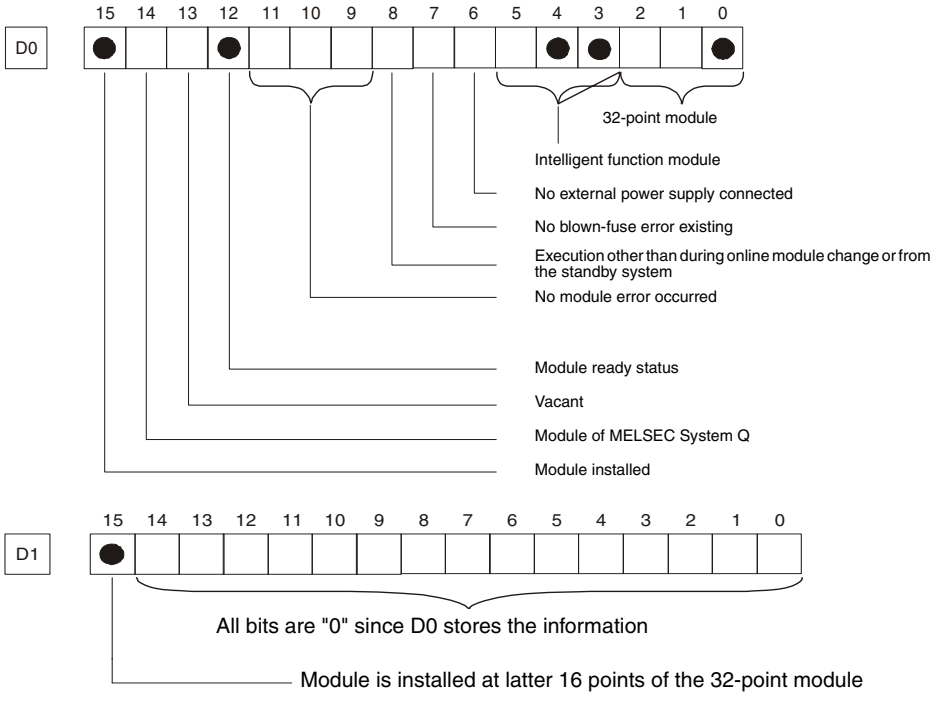
```
0  LD      X10
1  UNIRD   H1      D0      K2
```

Ladder Diagram

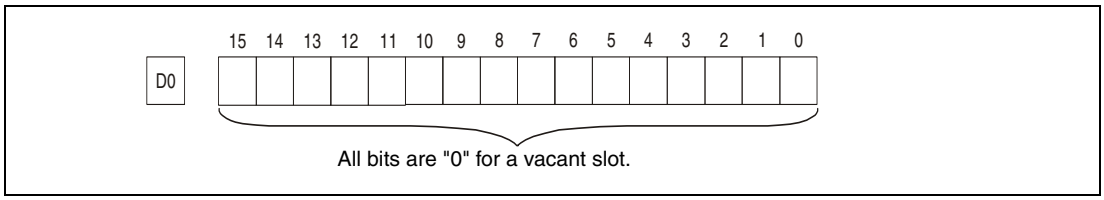


In this program example the module information is stored in D0 and D1. Readout results can be:

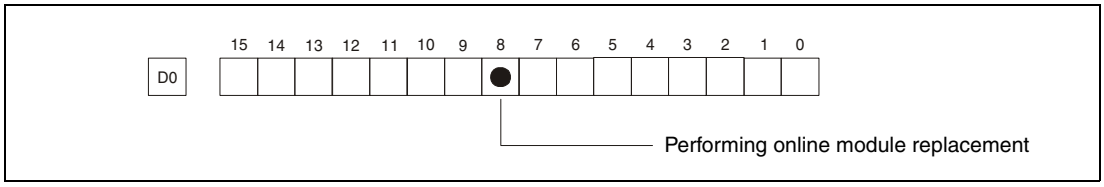
- For a 32-point intelligent function module of the System Q. With a 48- or 64-point module the same contents as stored in D1 is stored in D2 or D2 and D3 respectively.



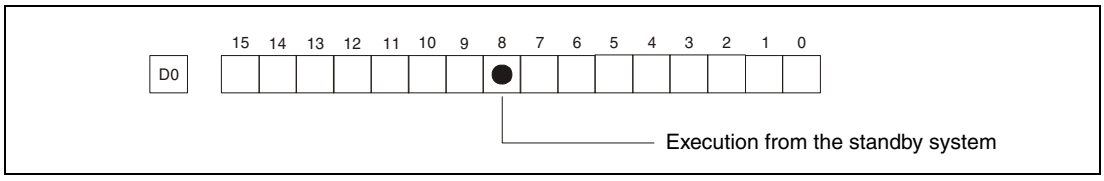
- Module information for a vacant slot



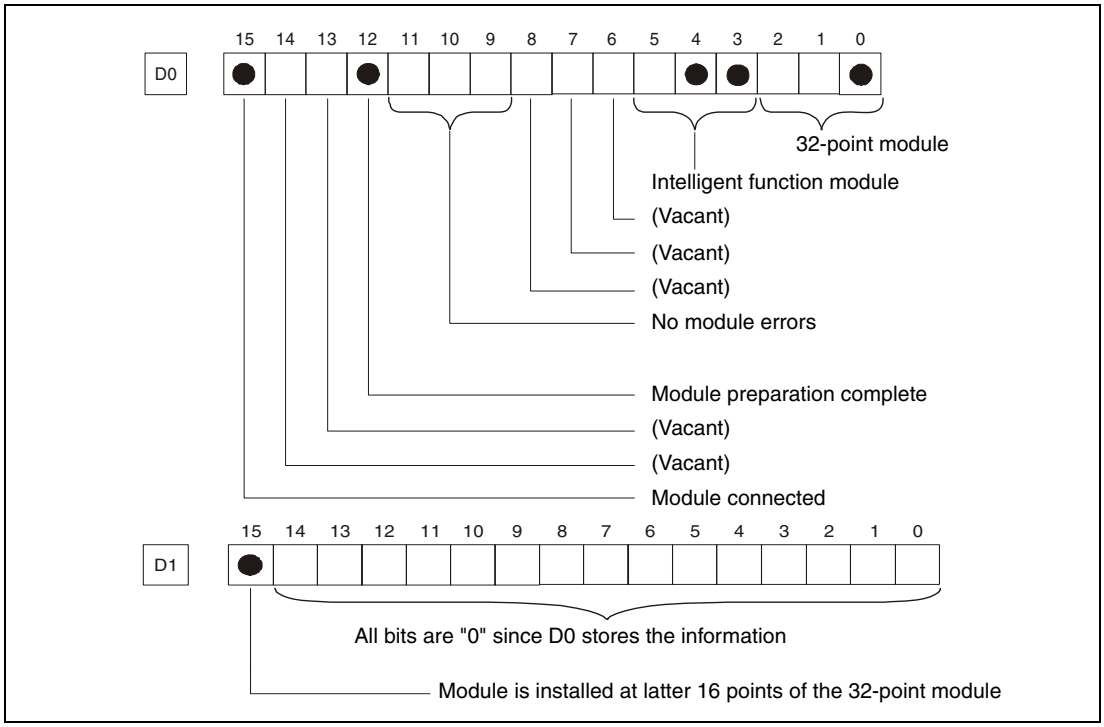
- Performing online module replacement



- Module information on the extension base unit is tried to be read from the standby system of the redundant system in separate mode:



- L series 32-point intelligent function module



7.18.10 TYPERD, TYPERDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				● [1)]	●

[1] Universal model QCPU: The serial number (first five digits) is "11043" or higher.

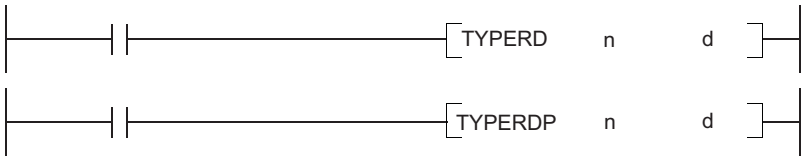
Devices

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
n	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Device	Meaning	Setting range	Data Type
n	Value obtained by dividing the head I/0 number of the module from which module information is read by 16. Set by user.	0 to FF$_H$, 3E0$_H$ to 3E3$_H$ (Universal model QCPU only)	BIN 16-Bit
d	d+0: Execution result of the instruction	Within each device range	BIN 16-Bit
	d+1 to d+9: Module model name		Character string

Functions **Reading module model name**

TYPERD **Read module model name**

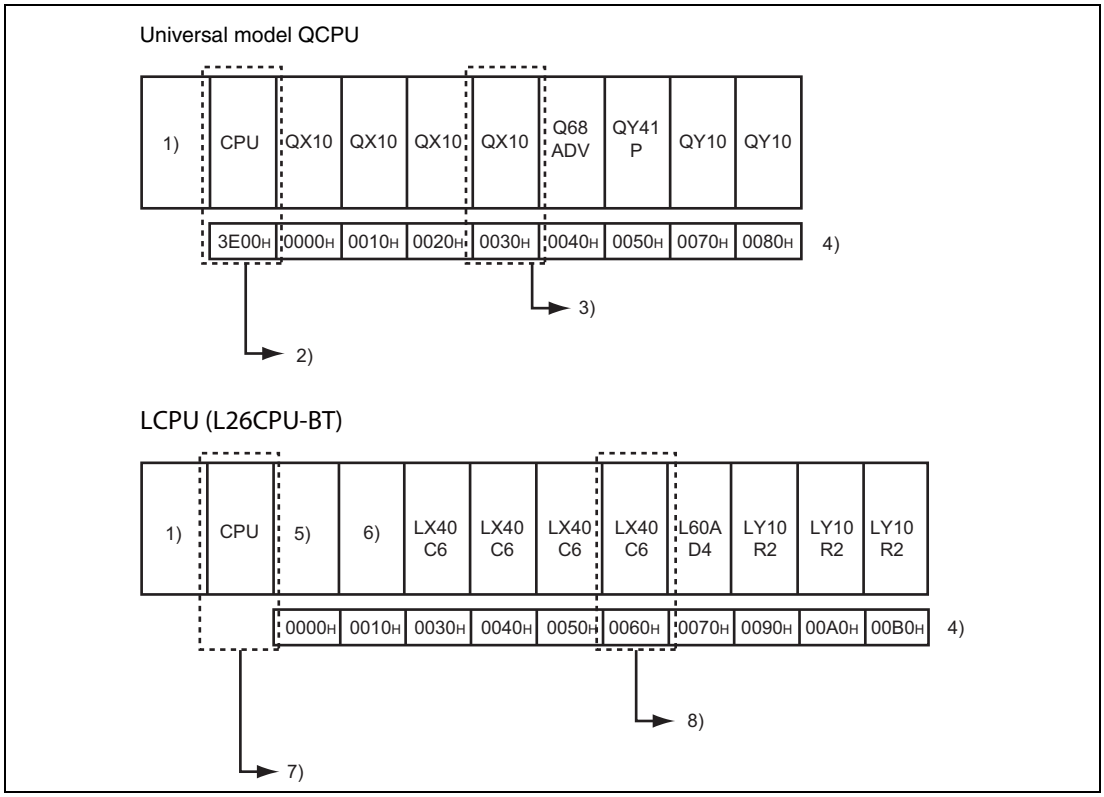
The TYPERD instruction reads the module information starting at the head I/O address, which is specified by n and stores the data at the address which is specified by d.

The following table shows which modules support the instruction:

Modules	Instruction Supported	
	QCPU	LCPU
CPU module	●	●
Input module	●	●
Output module	●	●
I/O combined module	●	○
Intelligent function module	●	●
GOT (bus connection)	●	○

●Supported
○Not supported

Specify the start I/O number of a module whose model name is to be read by "n" as follows: • Specify the value obtained by dividing the start I/O number of the target module by 16.



[1] Power supply module

[2] When the target module is a CPU module itself, specify the start I/O number by H3E0

[3] Specify the start I/O number by K3 or H3

[4] Head I/O number configured in the I/O assignment setting

[5] Built-in I/0

[6] Built-in CC-Link

[7] Specify H3E0 to read the module name of the CPU module

[8] Specify the start I/O number by K6 or H6

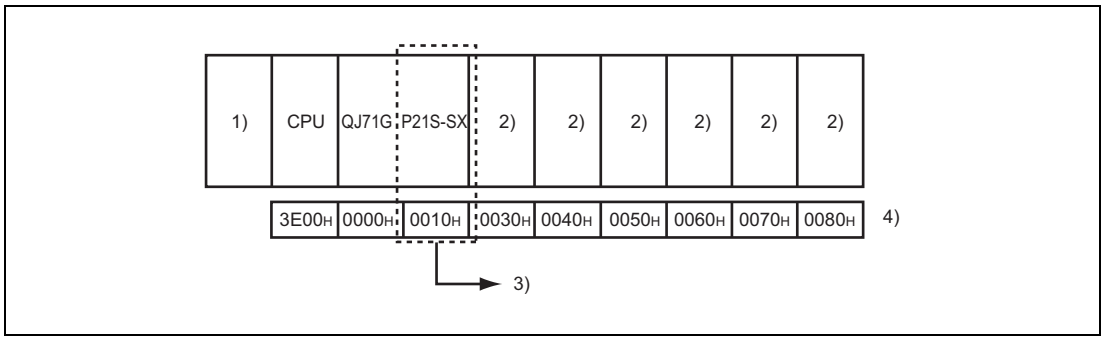
NOTE *On the LCPU, if the built-in I/O or first I/O on the built-in CC-Link is specified, then the model name of the CPU module is read.*

- When the target module occupies two slots

  The start I/O number to be specified may differ from that of the mounted module. For the start I/O number, refer to the manual of each module.

  Specify the value obtained by dividing the start I/O number of the target module by 16.

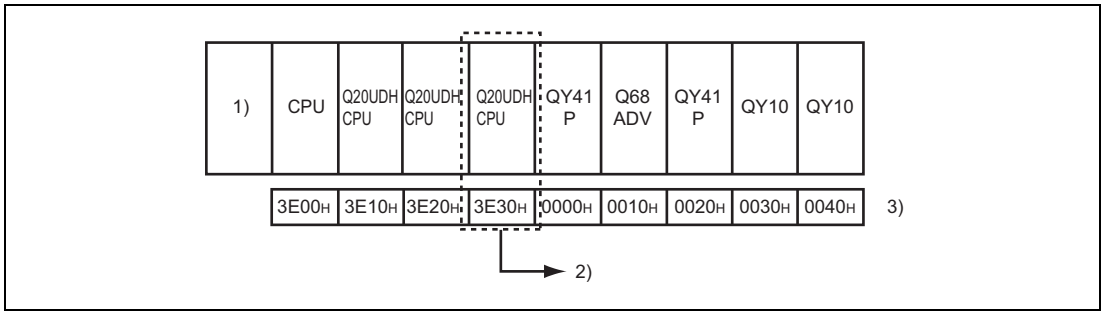
  Example: QJ71GP21S-SX
  Specify a value to which 0010H, start I/O number of the mounted module, is added.



[1] Power supply module
[2] Empty
[3] Specify the start I/O number by K1 or H1
[4] Start I/O number configured in the I/O assignment setting

- When the target module is a CPU module in multiple CPU systems

  Specify the value obtained by dividing the start I/O number of the target CPU module by 16.



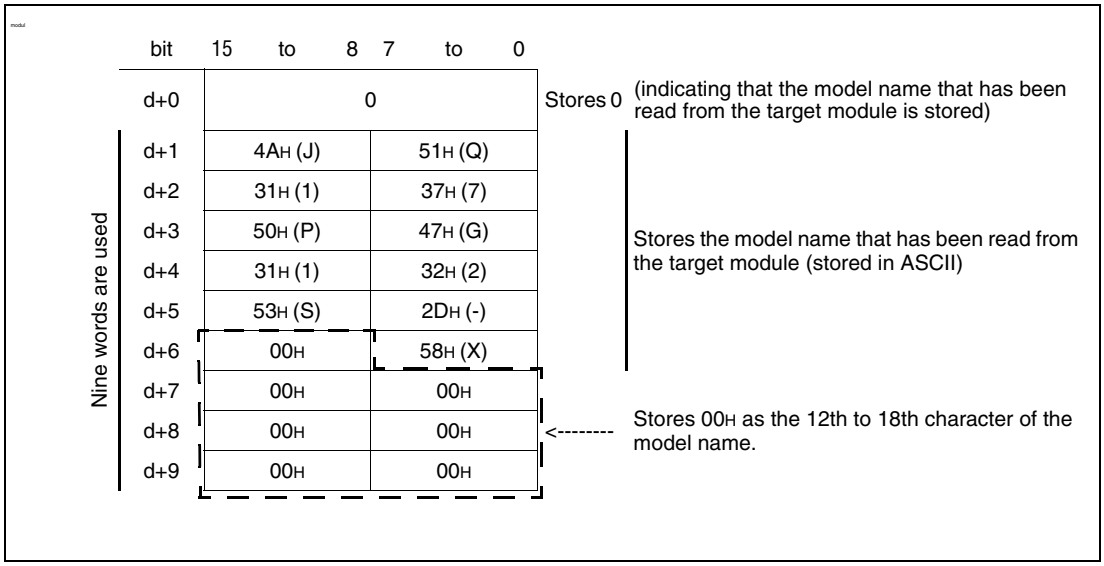
[1] Power supply module
[2] Specify the start I/O number by H3E3
[3] Start I/O number configured in the I/O assignment setting

  Or, the model name can be read by specifying the start I/O number of a module controlled by another CPU.

d+0 stores the execution result of the instruction and d+1 to d+9 store the module model name.

A value stored in d is as follows:

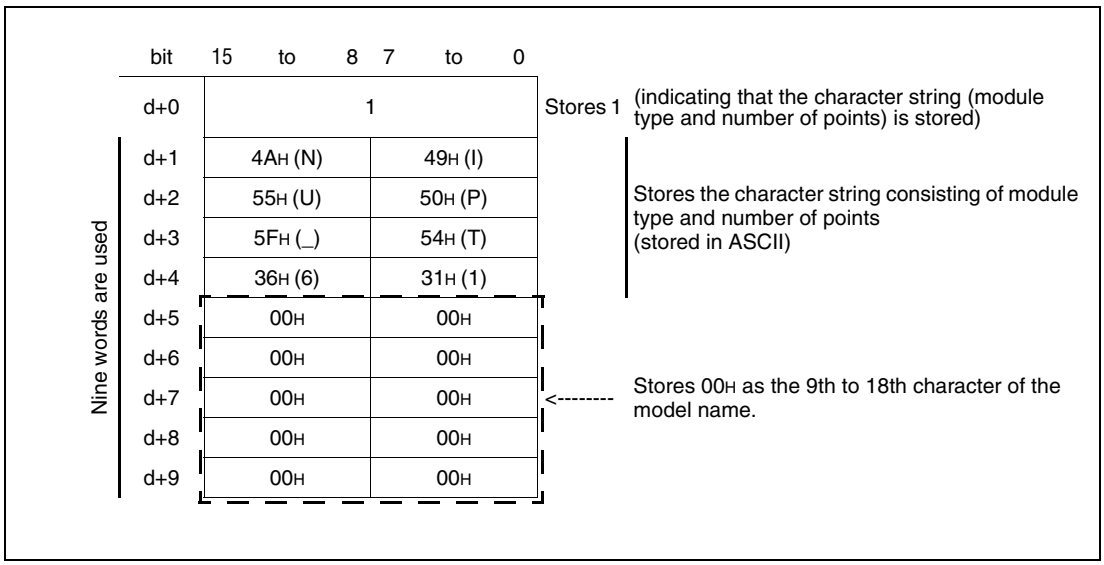
- When the model name has been read from the target module (example: QJ71GP21-SX)



The following table shows the examples of model names stored in d+1 to d+9.

Target Module	Stored Model Name
CPU module	Q06UDEHCPU
Intelligent function module	QJ71GP21-SX
GOT	GOT1000

- When the model name has not been written to the target module (example: QX40)

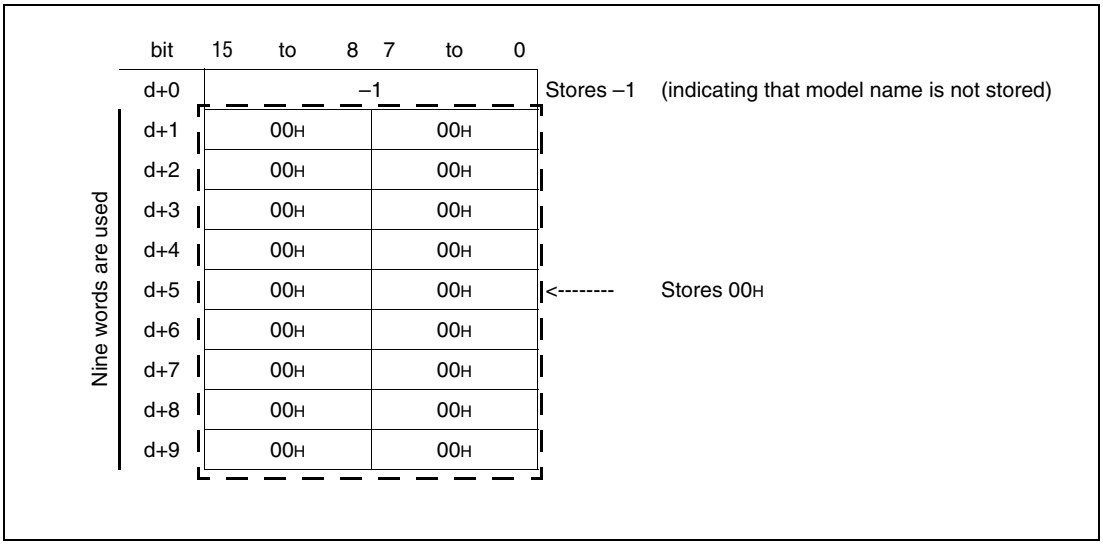


The following table shows the examples of character strings stored in d+1 to d+9.

Target Module	Stored character string [1)]
Input module	INPUT_16
Output module	OUTPUT_32
I/O combined module	MIXED_64
Intelligent function module (includes the QI60 and GOT)	INTELLIGENT_128

[1] The character strings consist of a string indicating the module type (for example: INPUT for the Input module) and a string indicating the number of I/O-points (16, 32, 48, 64, 128, 256, 512, 1024).

- Others
  - The specified slot is empty or the target module is during online module change.
  - The specified value (n) is not the start I/O number.
  - The specified value (n) is within the allowable setting range, but cannot be set in the I/O assignment setting screen of the PLC parameter dialog box.



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The target module cannot be communicated due to a failure.
  (Error code 2110)
- Devices by 10 words starting from the device specified by d exceed the device range.
  (Error code 4101)
- The specified value n is not within the range from 00H to FFH and from 3E0 to 3E3H.
  (Universal model QCPU)
  (Error code 4101)
- The specified value n is not within the range from 00H to FFH and is not 3E0H.
  (LCPU)
  (Error code 4101)

Program Example

TYPERD

The following program stores the model name of a module having the start I/O number 0020H in the area starting from the device specified by when X0 is turned on.

Instruction List

```
0   LD      X0
1   TYPERD  H2      D0
4   END
```

Ladder Diagram

7.18.11 TRACE, TRACER

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●[1)]	●

[1] Other than Q00UJCPU

Devices

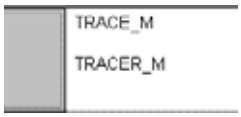
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2

Variables

Set data	Meaning	Data Type
—	—s	—

Functions

Trace set and trace reset

TRACE Trace set

The TRACE instruction stores the trace data designated by a peripheral device in the trace file in the memory card by the designated number when SM800, SM801, and SM802 turn ON. When the TRACE instruction is executed, SM803 turns ON. The sampling is repeated by the specified number of sampling trace after the TRACE instruction, then, data is latched and the trace is stopped.

The sampling is stopped if SM801 goes OFF during the trace execution.

After the TRACE instruction is executed and the trace is completed, SM805 turn ON.

During the execution of the TRACE instruction, other TRACE instructions are ignored. After the TRACE instruction is executed, the TRACE instruction is enabled again.

TRACER Trace reset

The TRACER instruction resets the TRACE instruction and the flags SM803 through SM805. After the TRACER instruction is executed, the TRACE instruction is enabled again.

NOTE

Please refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals) for more informations about trace.
Please refer to the operating manuals for the GX Works2 and GX IEC Developer for the execution of the trace with peripheral devices.

Program Example

TRACE, TRACER (GX Works2)

The following program executes the TRACE instruction when X0 is turned ON. When X1 is turned ON, the TRACE instruction is reset by the TRACER instruction.

Instruction List

```
0  LD      X0
1  TRACE
2  LD      X1
3  TRACER
```

Ladder diagram

7.18.12 SP.FWRITE

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●[1)]	●

[1] Other than Q00UJCPU/Q00UCPU/Q01UCPU

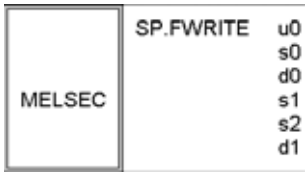
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s0	●	●	●	—	—	—	—	●	—
d0	—	●	●	—	—	—	—	—	—
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	●
d1	●[1)]	●[1)]	●[1)]	—	—	—	—	—	—

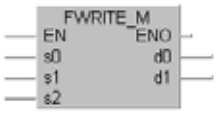
[1] Local devices and the devices designated for individual programs cannot be used.

GX IEC Developer

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

GX Works2

Variables

Set data	Meaning			Setting Range	Set By	Data Type
u0	Dummy (not used)			—	—	BIN 16-bit
s0	Drive designation.			2	User	
d0	Head number of the device storing the control data. The following control data is required.					BIN 16-bit
	Set data	**Item**	**Meaning/Set Data**	**Setting Range**	**Set By**	
	(d0)	Execution type	Specifies the execution type: 0000H: Write binary data 0100H: Write data after conversion into CSV format	0000H 0100H	User	
	(d0)+1	(Reserved)	Used by system	—	System	
	(d0)+2	Writing result (Number of written data)	Contains the number of actually written data. The unit for the value is determined by word/byte unit designation.	—	System	
	(d0)+3	Not used	—	—	—	
	(d0)+4 (d0)+5	Location in file	Sets the location in the file to start writing when **binary data** is selected (d0 = 0000H). 00000000H: From the beginning of the file 00000001H to FFFFFFFEH: From the specified address. The unit for the value is determined by word/byte unit designation. FFFFFFFFH: Add to the ending of the file. When data writing after **CSV format** conversion is selected (d0 = 0100H): For the High Performance model QCPU with serial number „01111“ or lower (first 5 digits), always set the beginning of the file (00000000H). For the High Performance model QCPU with serial number „01112“ or higher (first 5 digits)/Process CPU/Redundant CPU/Universal model QCPU set the file position. 00000000H to FFFFFFFEH: From the beginning of the file. FFFFFFFFH: Add to the ending of the file.	00000000H to FFFFFFFFH	User	
	(d0)+6	Number of columns	Sets the number of columns to write data in CSV format. 0: No column setting. Data is shown in a single row. > 0: Data is shown in the specified number of columns	0 to 65535	User	
	(d0)+7	Word/Byte designation	0: Word 1: Byte	0, 1	User	

Variables

<table>
<tr><th>Set data</th><th colspan="3">Meaning</th><th>Setting Range</th><th>Set By</th><th>Data Type</th></tr>
<tr><td rowspan="3">s1</td><td colspan="5">Head number of the device storing a file name.</td><td rowspan="3">BIN 16-bit</td></tr>
<tr><td>Set data</td><td>Item</td><td>Meaning/Set Data</td><td>Setting Range</td><td>Set By</td></tr>
<tr><td>(s1)+1 to (s1)+n</td><td>File name</td><td>The file name consists of up to 8 characters + period + extension (for example: ABD.BIN). The extension can be omitted. In this case, the period („.“) can also be omitted.
When more than 8 characters are used, the extension is ignored regardless of its presence. The Extension „BIN“ or „CSV“ is assigned automatically.</td><td>Character string</td><td>User</td></tr>
<tr><td rowspan="4">s2</td><td colspan="5">Head number of the device storing the data.</td><td rowspan="4">BIN 16-bit</td></tr>
<tr><td>Set data</td><td>letm</td><td>Meaning/Set Data</td><td>Setting Range</td><td>Set By</td></tr>
<tr><td>(s2)</td><td>Number of data to be written</td><td>Sets the number of data to be written (in units of words). This number should be designated in the unit of words even when byte is selected in (d0)+7.</td><td>1 to 480
1 to 32767 [1)]</td><td>User</td></tr>
<tr><td>(s2)+1 to (s2)+n</td><td>Data to be written</td><td>Data requested to be written.</td><td>0000H to FFFFH</td></tr>
<tr><td rowspan="4">d1</td><td colspan="5">Bit device that goes ON after the execution of the SP.FWRITE instruction.
When an error occurs, (d1)+1 goes ON.</td><td rowspan="4">Bit</td></tr>
<tr><td>Set data</td><td>Item</td><td>Meaning/Set Data</td><td>Setting Range</td><td>Set By</td></tr>
<tr><td>(d1)</td><td>Completion signal</td><td>Indicates the completion of the SP.FWRITE instruction.
ON: Completed
OFF: Not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d1)+1</td><td>Error completion signal</td><td>Indicates whether the SP.FWRITE instruction is normally completed or abnormally completed.
ON: Error completion
OFF: Normal completion</td><td>—</td></tr>
</table>

[1] Indicates the range applicable only for the Universal model QCPU and LCPU.

NOTES

- *For QCPU: Only the ATA card drive (2) can be set as s0 (drive designation).*
  Note that when the Flash card is loaded, the SP.FWRITE instruction cannot be used to perform writing. The SRAM card, standard RAM or standard ROM drive cannot be set.

  For LCPU: Only the SD memory card drive (2) can be set as s0 (drive designation).
- *The data written in CSV format is expressed as decimal value by the programming software. For example, the character „A“ (41H) is written as 65. The available range is from -32768 to 32767.*
- *For binary write, the word-specified file position setting range is 00000000H to 7FFFFFFFH and FFFFFFFFH.*
- *For the LCPU, this instruction cannot be executed while SM606 (SD memory card forced disable instruction) is ON. Even if the instruction is attempted to be executed, the command will be ignored.*

Functions **Writing data to a designated file**

SP.FWRITE Write data

The SP.WRITE instruction writes a specified number of data to the designated file. The user can select whether to write data as binary data without any conversion or to convert binary data into CSV-format data before writing it. (For QCPU, writing is only supported for ATA cards. For LCPU, it is only supported for SD memory cards.)

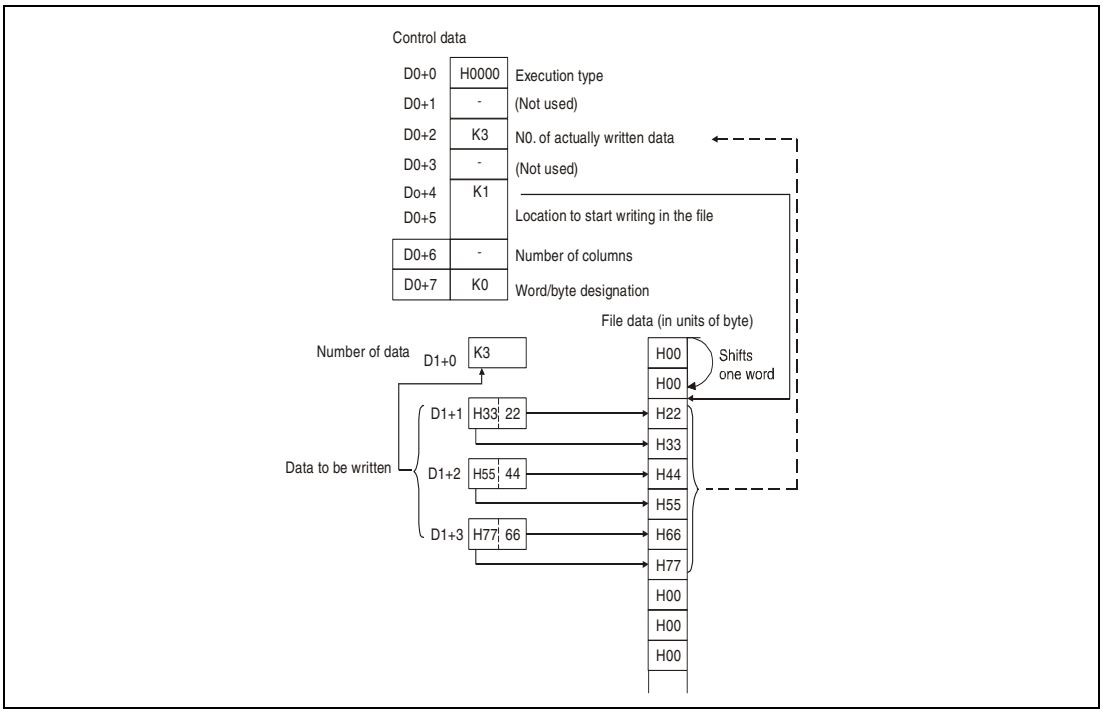
The completion signal bit device (d1)+0 automatically turns ON after the completion of the instruction is detected and the END instruction is executed. The bit device turns OFF at the execution of the END instruction in the next scan. This bit device can be used as the execution completion flag for the SP.FWRITE instruction.

When the SP.FWRITE instruction is completed abnormally, the error completion device (d1)+1 turns ON/OFF in synchronization with the execution completion flag (d1)+0. This bit device can be used as error completion flag for the SP.FWRITE instruction.

SM721 is on during the execution of the instruction. The SP.FWRITE instruction cannot be started while SM721 is ON. If an attempt is made, no processing is performed. When an error is detected prior to the execution of the instruction (before SM721 goes ON), the execution completion device [(d1)+0], the error completion device [(d1)+1] and SM721 do not turn ON.

The unit for the number of data to be written [(s2)+0] is „word“, regardless of the setting in (d0)+7 (word/byte designation).

The following shows the method for writing binary data when No. of request write data and file position are specified.



Writing of binary data

If the extension of the object file is omitted, „.BIN“ is added as an extension. When the designated file does not exist, a new file is created and the data is added and saved from the beginning of the file. The attributes of this new file are set using archive attributes.

When the size of the data exceeds that of the existing area in the file during the writing, the excess data is added at the end of the file.

An error occurs if the designated location in the file is larger than the file size:

- The High Performance model QCPU with the serial number "01111" or lower (first 5 digits) will issue an error code.
- The High Performance model QCPU with the serial number "01112" or higher (first 5 digits)/ Process CPU/Redundant CPU/Universal model QCPU/LCPU will not write any data and will complete the instruction without an error message.

When the medium runs out of free space when data is added/saved, an error occurs. In such a case, the data that is sucessfully added/saved remains in the medium. The error completion is indicated after as much data as possible is added/saved.

Writing of data after CSV format conversion

If the extension of the object file is omitted, „.CSV“ is added as an extension.

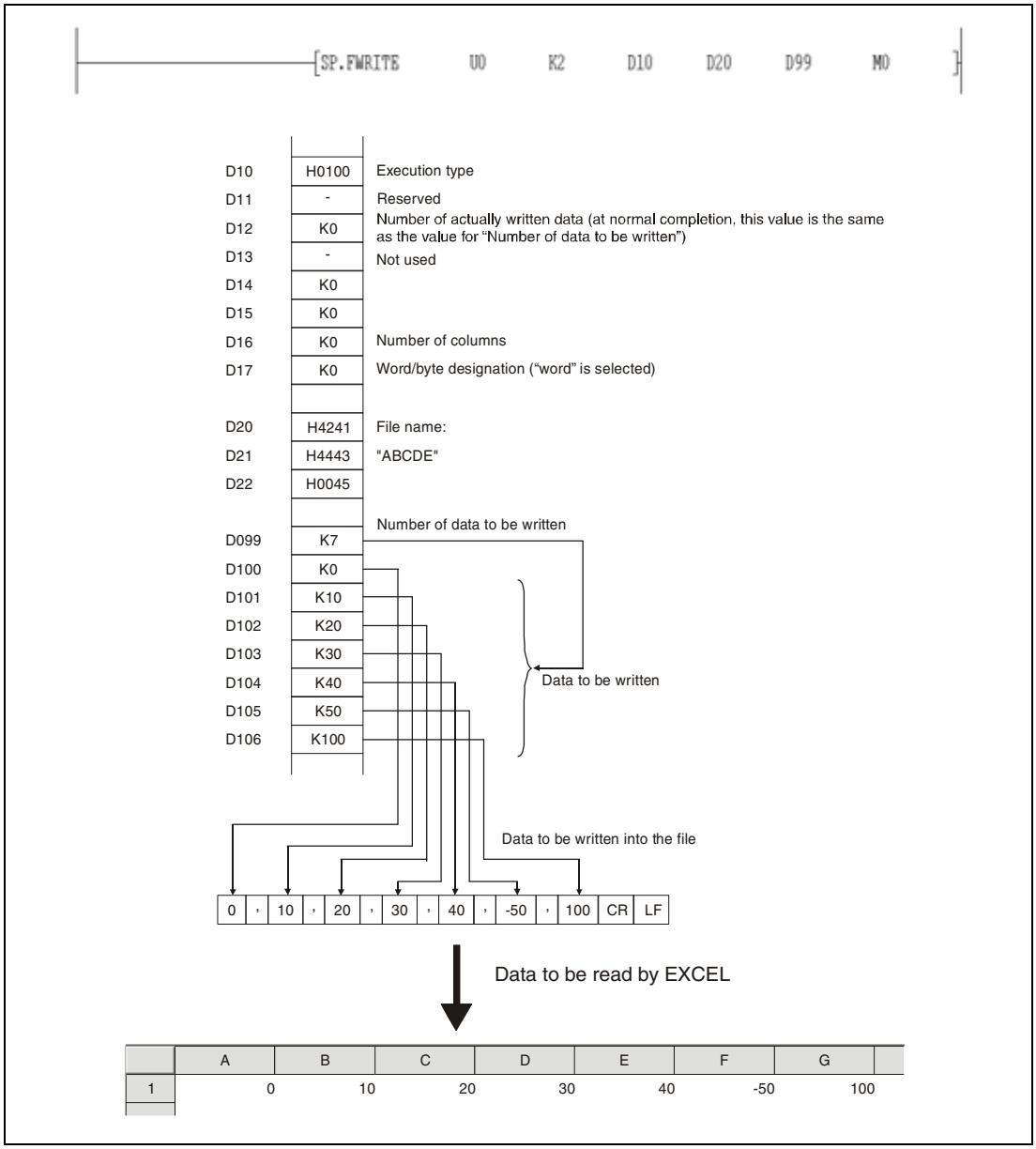
When an existing file is designated and:

– the High Performance model QCPU with the serial number "01111" or lower (first 5 digits) is used, all the contents of the file is deleted and the designated data is saved starting from the beginning of the file.

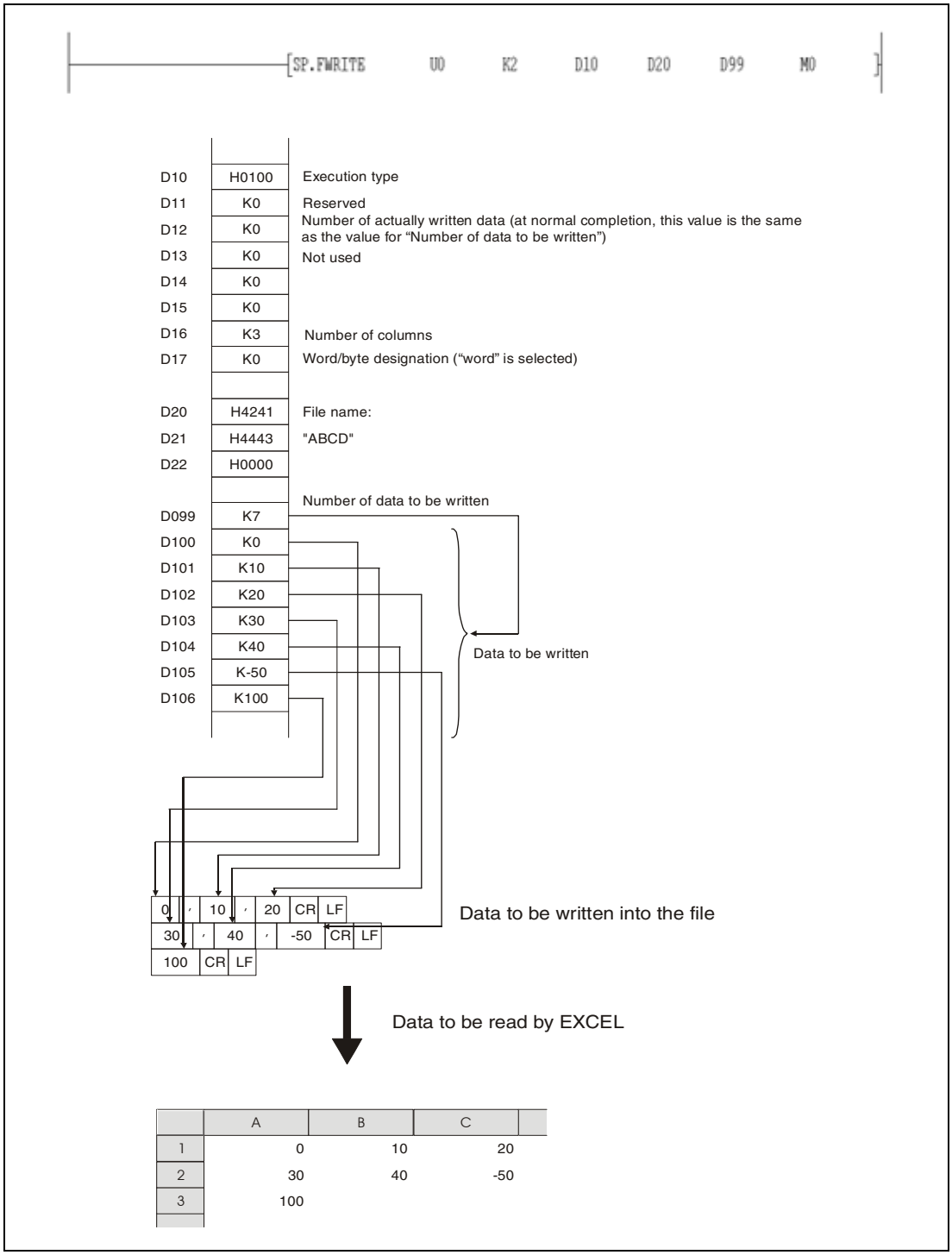
– the High Performance model QCPU with the serial number "01112" or higher (first 5 digits)/ Process CPU/Redundant CPU/Universal model QCPU or LCPU is used, the module will react depending of the value written in (d0)+4 and (d0)+5:
When other than FFFFFFFFH is specified in (d0)+4 and (d0)+5, the file contents will be deleted and the data will be stored from the beginning of the file.
When FFFFFFFFH is specified in (d0)+4 and (d0)+5, the data is added to the end of the file.

When the designated file does not exist, a new file is created and the data is added/saved from the beginning of the file. The attributes of this new file are set using archive attributes. An error occurs when the medium runs out of free space when data is added/saved. In such a case, the data that is sucessfully added/saved remains in the medium. The error completion is indicated after as much data as possible is added/saved.

When the designated number of columns is „0“, the data is stored as single-row data in a CSV-format file. The figure on the following page indicates such a case:


SP.FWRITE U0 K2 D10 D20 D99 M0
D10 H0100 Execution type
D11 - Reserved
D12 K0 Number of actually written data (at normal completion, this value is the same as the value for "Number of data to be written")
D13 - Not used
D14 K0
D15 K0
D16 K0 Number of columns
D17 K0 Word/byte designation ("word" is selected)
D20 H4241 File name:
D21 H4443 "ABCDE"
D22 H0045
Number of data to be written
D099 K7
D100 K0
D101 K10
D102 K20
D103 K30
D104 K40
D105 K50
D106 K100
Data to be written
Data to be written into the file
0 , 10 , 20 , 30 , 40 , -50 , 100 CR LF
Data to be read by EXCEL
A B C D E F G
1 0 10 20 30 40 -50 100

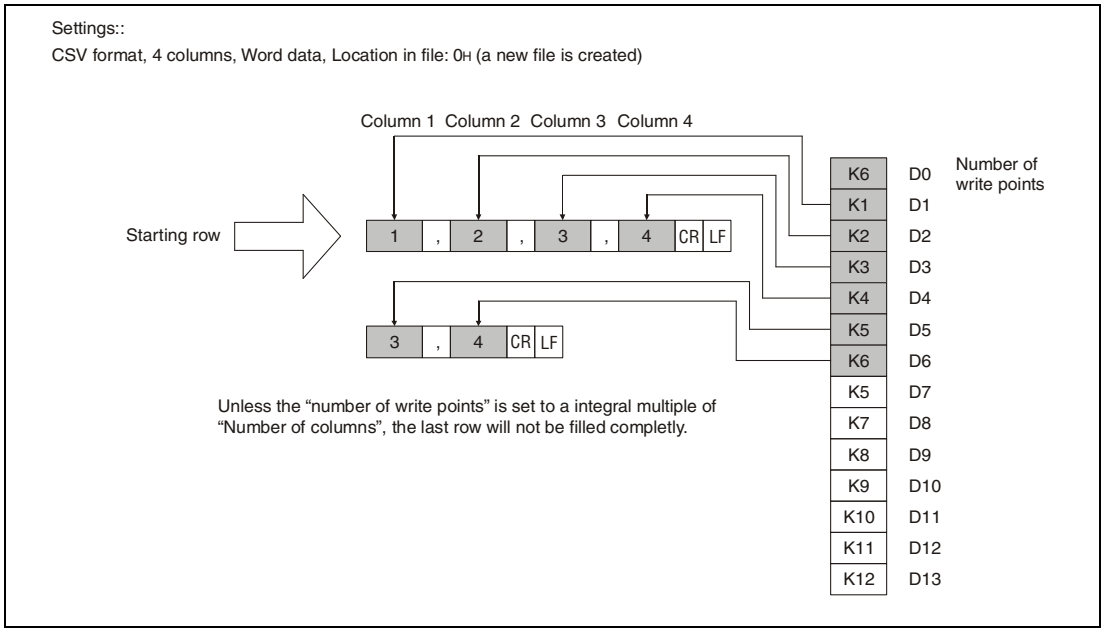
When data is written after CSV format conversion and the designated number of columns is other than „0“, the data is stored as table data with the specified number of columns in a CSV format file. The following figure shows an example:

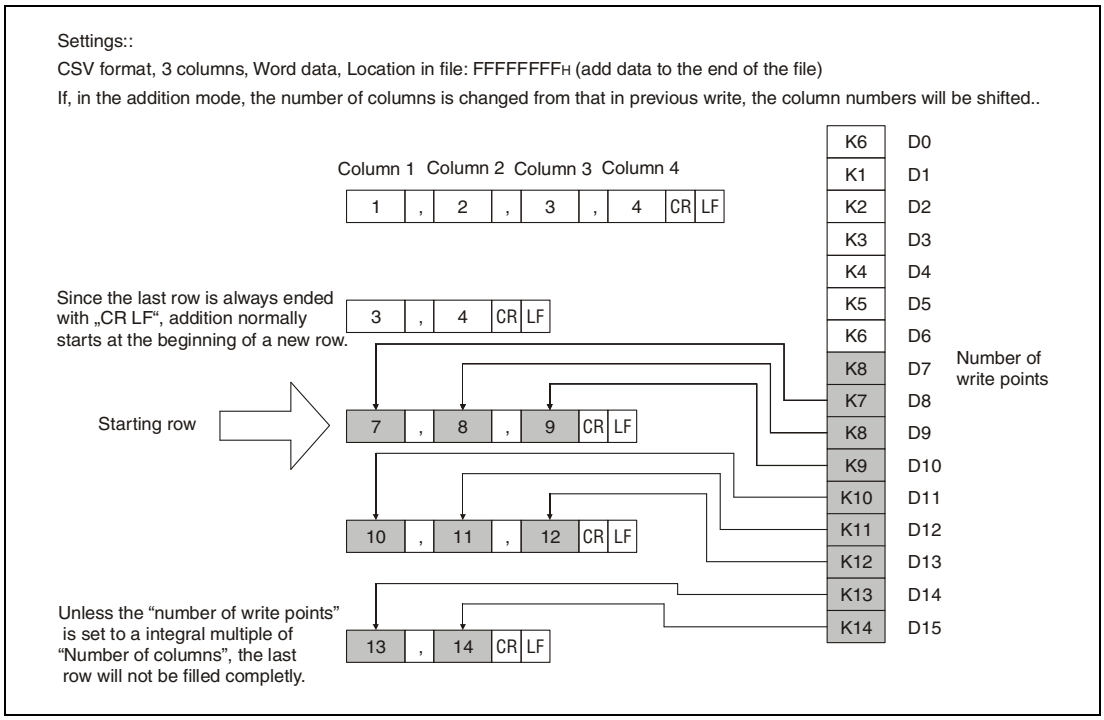


The following two figures are showing examples of writing data with the following CPU modules:

– High Performance model QCPU with serial number „01112“ and higher (first 5 digits)/ Process CPU/Redundant CPU/Universal model QCPU

– LCPU





NOTE *Do not execute the SP.FWRITE instruction in an interrupt program.*

Method for calculating the file size (total number of bytes) when a CSV format file is written to the ATA card

Total number of bytes = Total bytes excluding final line + bytes of final line

Number of bytes on a line = number of columns[1)] + 1 + total bytes of all data values on line[2)]

[1] For all lines but the final line, this is the specified number of columns. The number of columns on the final line depends on the number of columns specified via the amount of data written. It is calculated as follows.

(1) The number of lines excluding the final line is calculated.
Number of lines excluding final line = Amount of data in write request
+ number of columns (remainders discarded)

(2) The number of columns in the final line is calculated.
Number of columns in final line = Amount of data in write request
- number of lines excluding final line
* number of columns)

[2] The number of bytes for each data value is calculated as shown below.

Sign of Data Value	Bytes per Data Value	Byte Count Range	Examples
Positive	Number of digits	1 to 5 (word specified) 1 to 3 (byte specified)	12345: 5 bytes 67: 2 bytes
Negative	Number of digits + 1	2 to 6 (word specified) 2 to 4 (byte specified)	-12345: 6 bytes -67: 3 bytes

Operation Errors

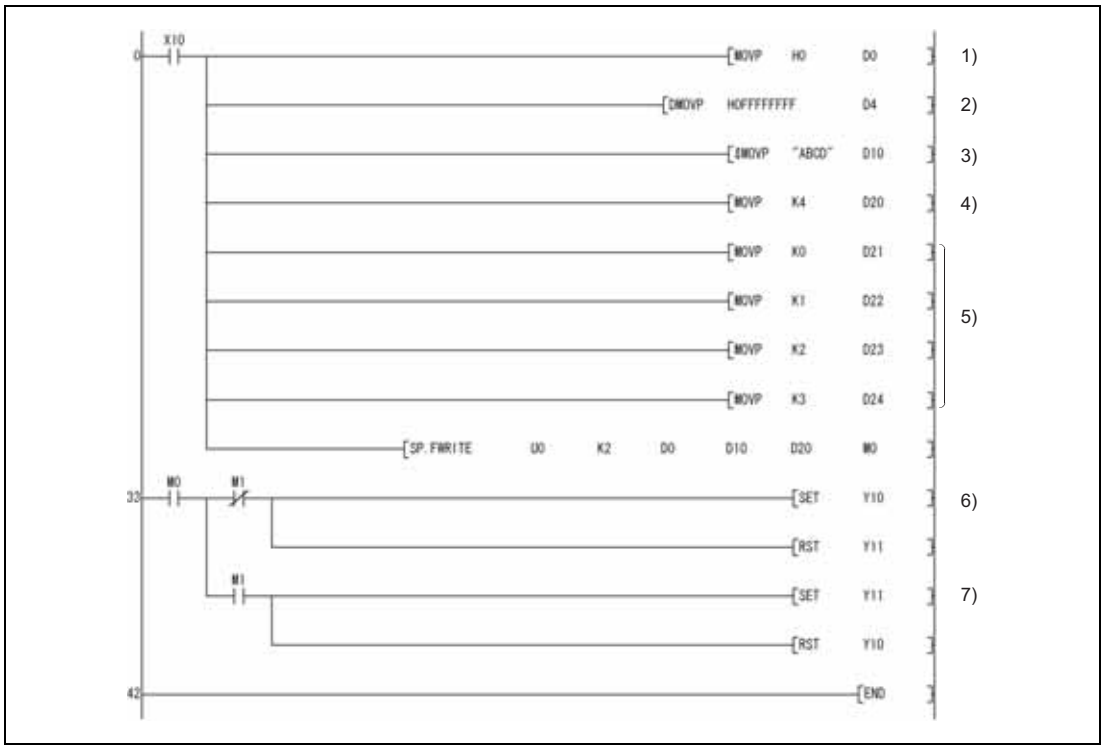
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The drive specified by s0 contains a medium other than an ATA card. (for QCPU) (Error code 4100)
- The drive specified by s0 contains a medium other than the SD Memory card. (for LCPU) (Error code 4100)
- Values specified in the areas for control data are out of the setting range. (Error code 4100)
- The value „number of data to be written“ in (s2)+0 is out of the setting range, or is larger than the data stored in the area beginning with (s2)+1. (Error code 4101)
- Free space in the ATA card is insufficient. (for QCPU) (Error code 4100)
- Free space in the SD Memory card is insufficient. (for LCPU) (Error code 4100)
- No free space is found when an attempt is made to create a new file. (Error code 4100)
- An invalid device is designated. (Error code 4004)
- Access error occurred in the ATA card. (for QCPU) (Error code 4100)
- Access error occurred in the SD Memory card. (for LCPU) (Error code 4100)
- An unusable value is set for a file name (s1). (Error code 4100)
- The attribute of a file name (s1) is "read only". (Error code 4100)
- The device specified by d0 or d1 exceeds the range of the corresponding device. (For the Universal model QCPU, LCPU) (Error code 4101)

Program Example 1

SP.FWRITE (GX Works2)

In the following program example, four bytes of binary data (00H, 01H, 02H, and 03H) are added to file „ABCD.BIN“ when X10 turn ON. The memory card is inserted in drive 2.
Beginning with D0, eight points are reserved for control data.

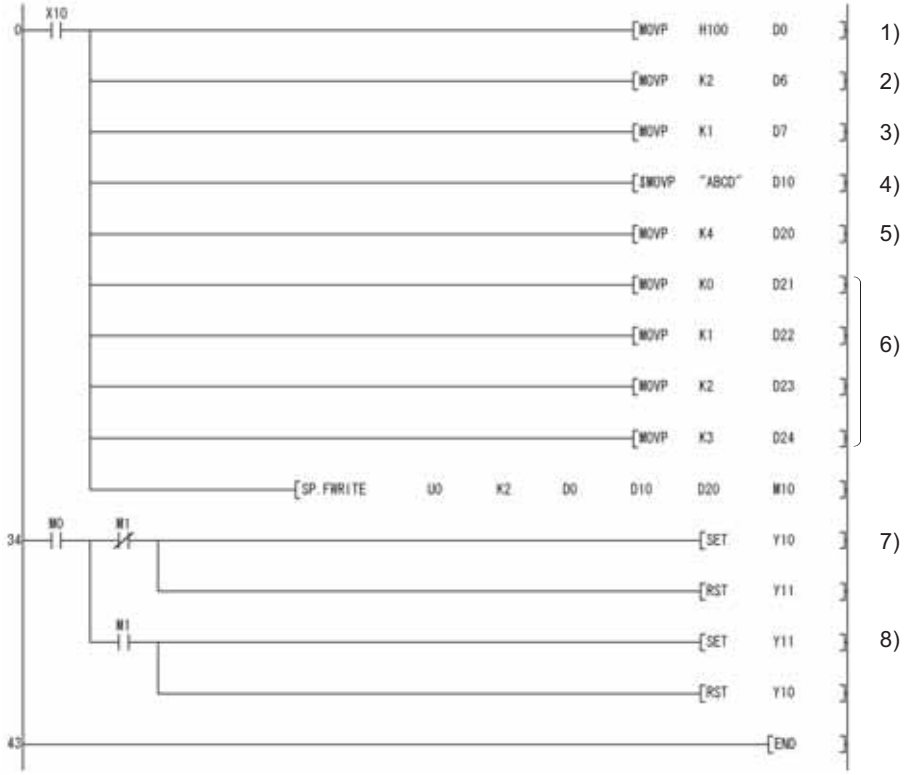


[1] Setting of the execution/completion type (In this example: binary data)
[2] Setting of the location in the file (In this example: data is added)
[3] Setting of the file name, the extension „.BIN“ is added automatically.
[4] Number of data to be written.
[5] The data (00H, 01H, 02H, and 03H) is moved to the control data area.
[6] Normal completion display
[7] Error completion display

Program Example 2

SP.FWRITE (GX Works2)

When X10 is turned ON, the following program creates a file named „ABCD.CSV“ in the memory card inserted to drive 2. Then, four bytes of data (00H, 01H, 02H and 03H) are written as two-column table data in CSV format. Control data is stored from D0 onward (8 points).



0	'	1	'	CR	LF
2	'	3	'	CR	LF

Contents of the file to be written

Data to be read by EXCEL

	A	B
1	0	1
2	2	3

[1] Setting of the execution/completion type (In this example: CSV format)
[2] Setting of the number of columns
[3] Sets the data type specified
[4] Setting of the file name, the extension „.CSV“ is added automatically.
[5] Number of data to be written.
[6] The data (00H, 01H, 02H, and 03H) is moved to the control data area.
[7] Normal completion display
[8] Error completion display

7.18.13 SP.FREAD

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●[1)]	●

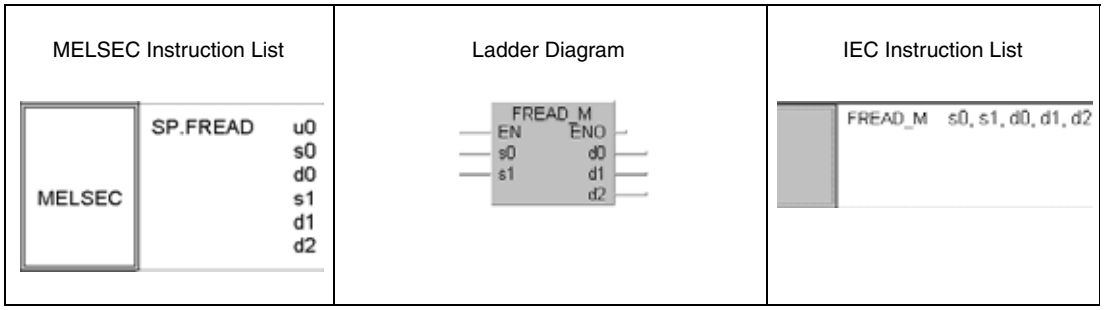
[1] Other than Q00UJCPU/Q00UCPU/Q01UCPU

Devices

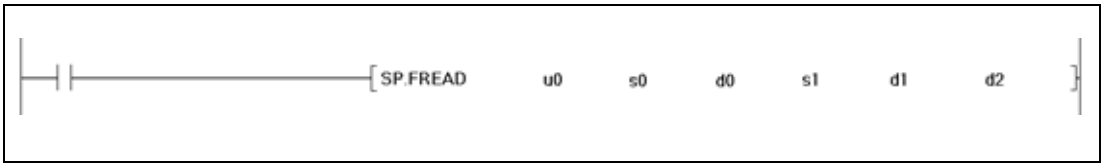
	Usable Devices									
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant		Other
	Bit	Word		Bit	Word			K, H (16#)	$	
s0	●	●	●	—	—	—	—	●	—	—
d0	—	●	●	—	—	—	—	—	—	—
s1	—	●	●	—	—	—	—	—	—	—
d1	—	●	●	—	—	—	—	—	●	—
d2	●[1)]	●[1)]	●[1)]	—	—	—	—	—	—	—

[1] Local devices and the devices designated for individual programs cannot be used.

GX IEC Developer



GX Works2



Variables

Set data	Meaning			Setting Range	Set By	Data Type
u0	Dummy (not used)			—	—	BIN 16-bit
s0	Drive designation			2	User	
d0	Head number of the device storing the control data. The following control data is required.					BIN 16-bit
	Set data	**Item**	**Meaning/Set Data**	**Setting Range**	**Set By**	
	(d0)	Execution type	Specifies the execution type: 0000H: Read binary data 0100H: Read data after conversion into CSV format	0000H 0100H	User	
	(d0)+1	(Reserved)	Used by system	—	System	
	(d0)+2	Number of data to be read	Sets the number of data to be read (in units of words). This number should be designated in the unit of words even when byte is selected in (d0)+7.	1 to 480 1 to 32767 1)	User	
	(d0)+3	Not used	—	—	—	
	(d0)+4 (d0)+5	Location in file	Sets the location in the file to start reading when **binary** data is selected (d0 = 0000H). 00000000H: From the beginning of the file 00000001H to FFFFFFFCH: From the specified address. The unit for the value is determined by word/byte unit designation. FFFFFFFDH: Setting disabled When data reading after **CSV format** conversion is selected (d0 = 0100H): For the High Performance model QCPU with serial number "01111" or lower (first 5 digits), always set the beginning of the file (00000000H). For the High Performance model QCPU with serial number "01112" or higher (first 5 digits) / Process CPU / Redundant CPU / Universal model QCPU/LCPU set the file position. 00000000H: From the beginning of the file. 00000001H: to FFFFFFFEH: From the specified address. FFFFFFFFH: Read continues, starting at the previous read position	00000000H to FFFFFFFFH	User	
	(d0)+6	Number of columns	Sets the number of columns for the data to be read. 0: No column setting. Data is considered to be in a single row. > 0: Data is considered to be a table with the specified number of columns	0 to 65535	User	
	(d0)+7	Word/Byte designation	0: Word 1: Byte	0, 1	User	

Variables

<table>
<tr><th>Set data</th><th colspan="4">Meaning</th><th>Setting Range</th><th>Set By</th><th>Data Type</th></tr>
<tr><td rowspan="3">s1</td><td colspan="5">Head number of the device storing a file name.</td><td rowspan="3">BIN 16-bit</td></tr>
<tr><td>Set data</td><td>Item</td><td colspan="2">Meaning/Set Data</td><td>Setting Range</td><td>Set By</td></tr>
<tr><td>(s1) to (s1)+n</td><td>File name</td><td colspan="2">The file name consists of up to 8 characters + period + extension (for example: ABD.BIN). The extension can be omitted. In this case, the period („.“) can also be omitted.
When more than 8 characters are used, the extension is ignored regardless of its presence. The Extension „BIN“ or „CSV“ is assigned automatically.</td><td>Character-string</td><td>User</td></tr>
<tr><td rowspan="4">d1</td><td colspan="5">Head number of the device storing the data.</td><td rowspan="4">BIN 16-bit</td></tr>
<tr><td>Set data</td><td>Item</td><td colspan="2">Meaning/Set Data</td><td>Setting Range</td><td>Set By</td></tr>
<tr><td>(d1)</td><td>Reading result (Number of read data)</td><td colspan="2">Contains the number of actually read data. The unit for the value is determined by word/byte unit designation.</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d1)+1 to (d1)+n</td><td>Data to be read</td><td colspan="2">Data requested to be read</td><td>—</td></tr>
<tr><td rowspan="4">d2</td><td colspan="5">Bit device that goes ON after the execution of the SP.FREAD instruction. When an error occurs, (d1)+1 goes ON.</td><td rowspan="4">Bit</td></tr>
<tr><td>Set data</td><td>Item</td><td colspan="2">Meaning/Set Data</td><td>Setting Range</td><td>Set By</td></tr>
<tr><td>(d2)</td><td>Completion signal</td><td colspan="2">Indicates the completion of the processing.
ON: Completed
OFF: Not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d2)+1</td><td>Error completion signal</td><td colspan="2">Indicates whether the processing is normally completed or abnormally completed.
ON: Error completion
OFF: Normal completion</td><td>—</td></tr>
</table>

[1] Indicates the range applicable only for the Universal model QCPU and LCPU.

NOTE

- *For QCPU: Only the ATA card drive (2) can be set as s0 (drive designation).*
  Note that when the Flash card is loaded, the SP.FREAD instruction cannot be used to perform reading. The SRAM card, standard RAM or standard ROM drive cannot be set.

  For LCPU: Only the SD memory card drive (2) can be set as s0 (drive designation).
- *The data written in CSV format is expressed as decimal value by the programming software. For example, the character „A“ (41H) is written as 65. The available range is from -32768 to 32767.*
- *For binary read, the word-specified file position setting range is 00000000H to 7FFFFFFFH.*
- *For the LCPU, this instruction cannot be executed while SM606 (SD memory card forced disable instruction) is ON. Even if the instruction is attempted to be executed, the command will be ignored.*

Functions

Reading data from a designated file

SP.FREAD Read data

The SP.FREAD instruction reads a specified number of data from a designated file. The user can select whether to read data as binary data without any conversion or to convert data from the CSV-format into binary data before reading it. (For QCPU, reading is only supported for ATA cards. For LCPU, it is only supported for SD memory cards.)

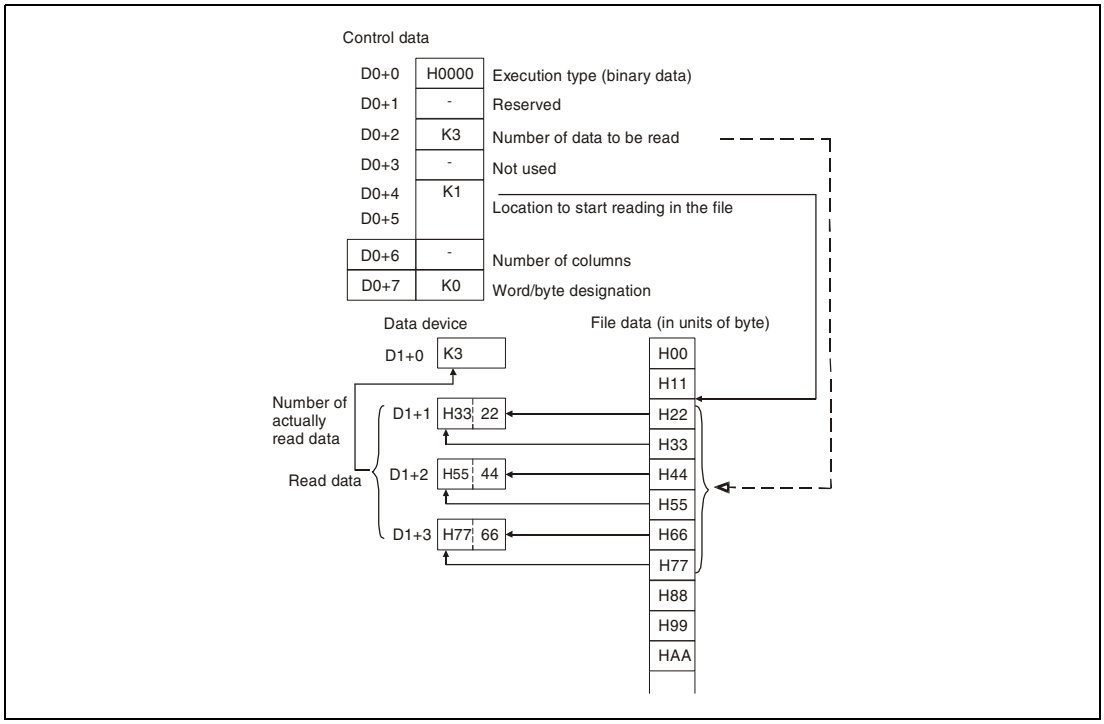
The completion signal bit device (d2)+0 automatically turns ON after the completion of the instruction is detected and the END instruction is executed. The bit device turns OFF at the execution of the END instruction in the next scan. This bit device can be used as the execution completion flag for the SP.FREAD instruction.

When the SP.FREAD instruction is completed abnormally, the error completion device (d2)+1 turns ON/OFF in synchronization with the execution completion flag (d2)+0. This bit device can be used as error completion flag for the SP.FREAD instruction.

SM721 is on during the execution of the SP.FREAD instruction. The SP.FREAD instruction cannot be started while SM721 is ON. If an attempt is made, no processing is performed. When an error is detected prior to the execution of the instruction (before SM721 goes ON), the execution completion device [(d2)+0], the error completion device [(d2)+1] and SM721 do not turn ON.

The unit for the number of data to be read [(d0)+0] is „word“, regardless of the setting in (d0)+7 (word/byte designation).

The following figure illustrates the reading of binary data:



Reading of binary data

If the extension of the object file is omitted, „.BIN“ is added as an extension. When the designated file does not exist, an error ocurs.

An error occurs if the designated location in the file is larger than the file size:

- The High Performance model QCPU with serial number "01111" or lower (first 5 digits) will issue an error code.
- The High Performance model QCPU with serial number "01112" or higher (first 5 digits), Process CPU, Redundant CPU, Universal model QCPU or LCPU will not read any data and will complete the instruction without an error message.

Reading of data after CSV format conversion

The elements in the CSV-format file (cells for EXCEL) are read row by row. The numerical values and character strings are converted into binary data and stored in the device. If the extension of the file is omitted, „.CSV“ is added as an extension.

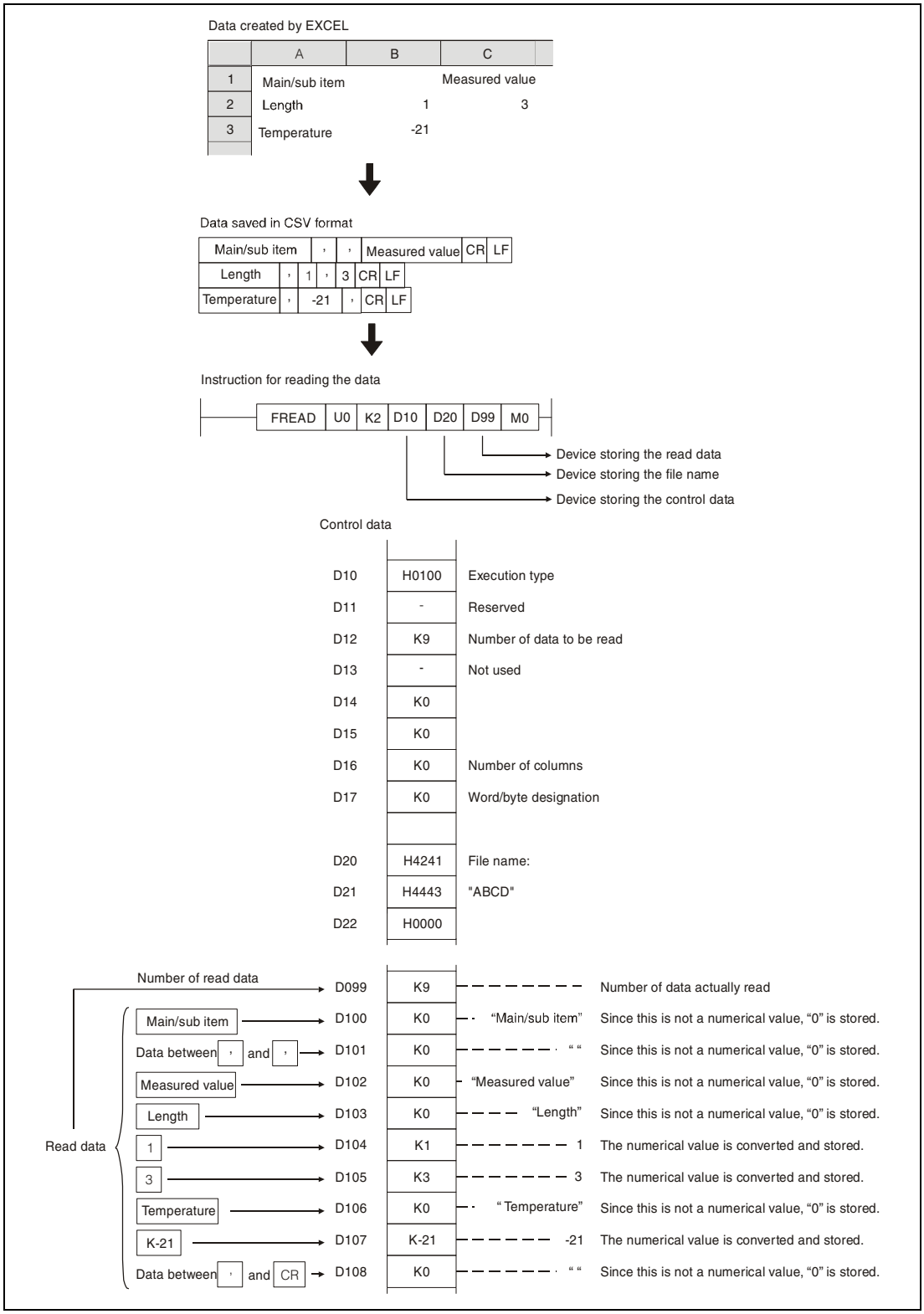
The reading starts at the specified position of the file. The number of elements to read is set in the control data with (d0)+2.

When the designated file does not exist, an error occurs.

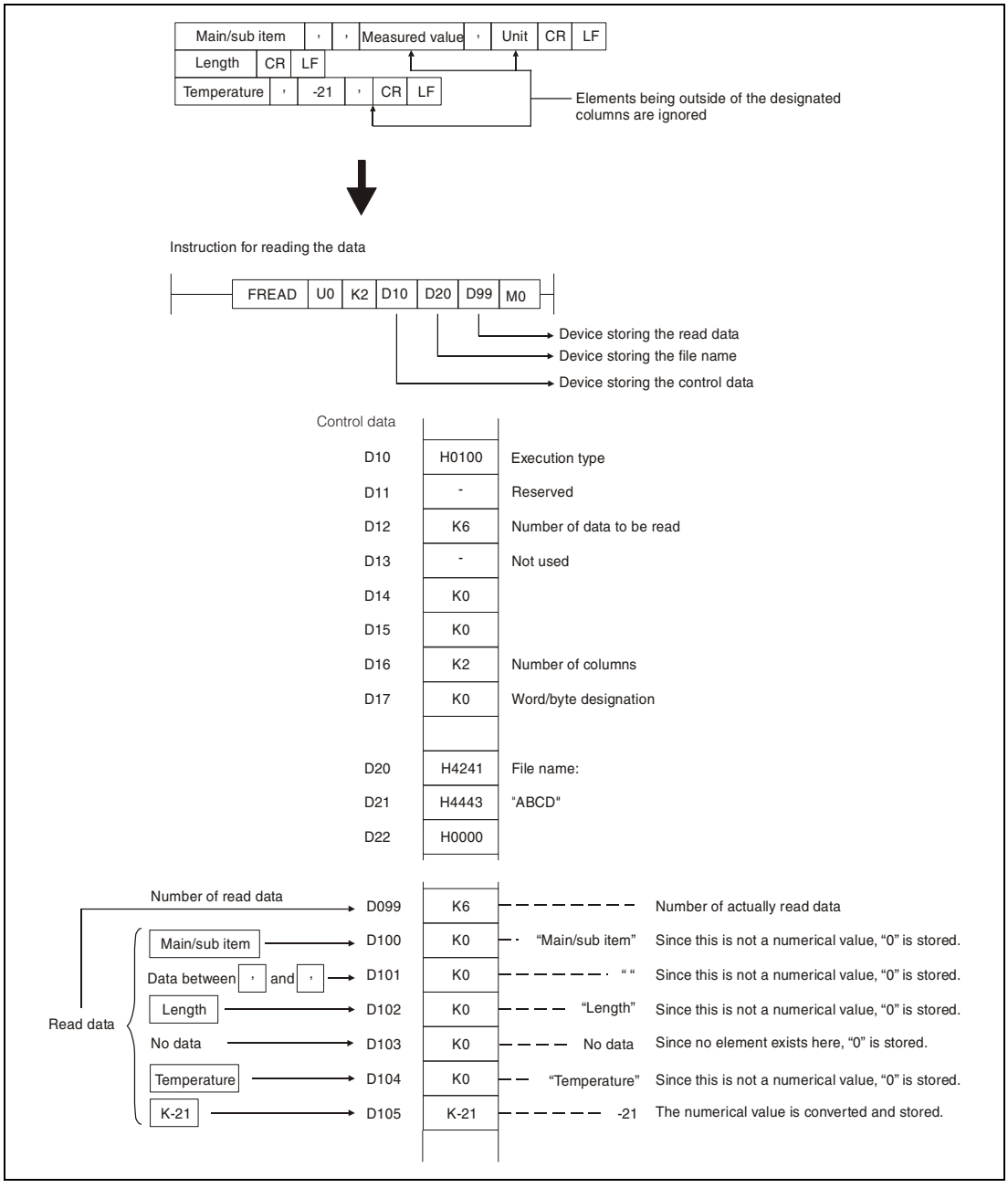
– When the last data of the file is reached before the specified number of data has been read, a High Performance model QCPU with serial number "01111" or lower (first 5 digits) will issue an error code.

– A High Performance model QCPU with serial number "01112" or higher (first 5 digits), Process CPU, Redundant CPU, Universal model QCPU or LCPU will read the data that can be read.

When the specified number of columns is „0“, the data is read by ignoring the rows in a CSV-format file. The figure on the following page shows the handling of data in such a case.

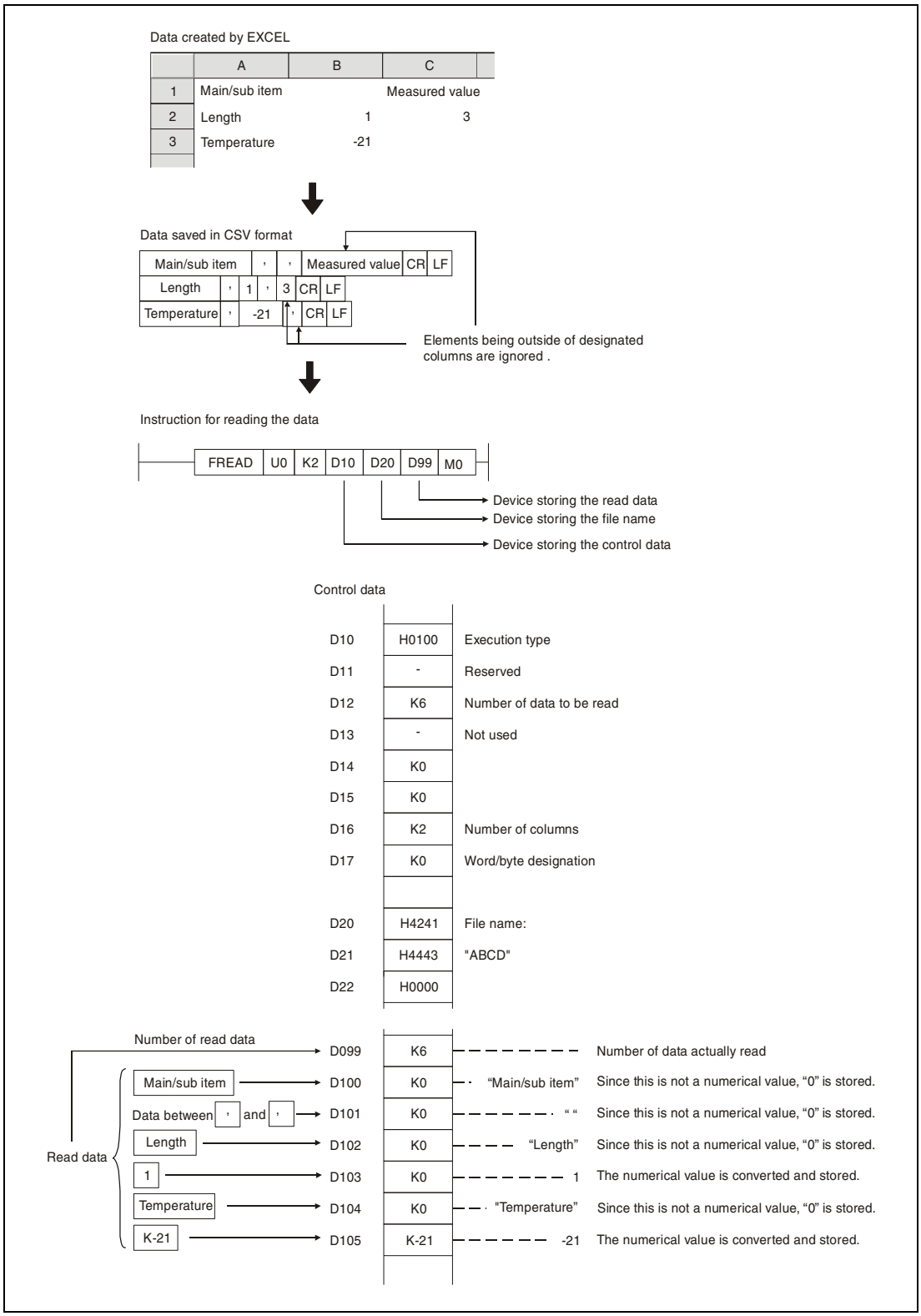

Data created by EXCEL
A
B
C
1
Main/sub item
Measured value
2
Length
1
3
3
Temperature
-21
Data saved in CSV format
Main/sub item , , Measured value CR LF
Length , 1 , 3 CR LF
Temperature , -21 , CR LF
Instruction for reading the data
FREAD U0 K2 D10 D20 D99 M0
Device storing the read data
Device storing the file name
Device storing the control data
Control data
D10 H0100 Execution type
D11 - Reserved
D12 K9 Number of data to be read
D13 - Not used
D14 K0
D15 K0
D16 K0 Number of columns
D17 K0 Word/byte designation
D20 H4241 File name:
D21 H4443 "ABCD"
D22 H0000
Number of read data
D099 K9 Number of data actually read
Read data
Main/sub item
D100 K0 "Main/sub item" Since this is not a numerical value, "0" is stored.
Data between , and ,
D101 K0 " " Since this is not a numerical value, "0" is stored.
Measured value
D102 K0 "Measured value" Since this is not a numerical value, "0" is stored.
Length
D103 K0 "Length" Since this is not a numerical value, "0" is stored.
1
D104 K1 1 The numerical value is converted and stored.
3
D105 K3 3 The numerical value is converted and stored.
Temperature
D106 K0 " Temperature" Since this is not a numerical value, "0" is stored.
K-21
D107 K-21 -21 The numerical value is converted and stored.
Data between , and CR
D108 K0 " " Since this is not a numerical value, "0" is stored.

If the number of columns varies in each row, the data is also read by ignoring the rows. (EXCEL does not create such files. This happens when a user modifies a CSV file.)



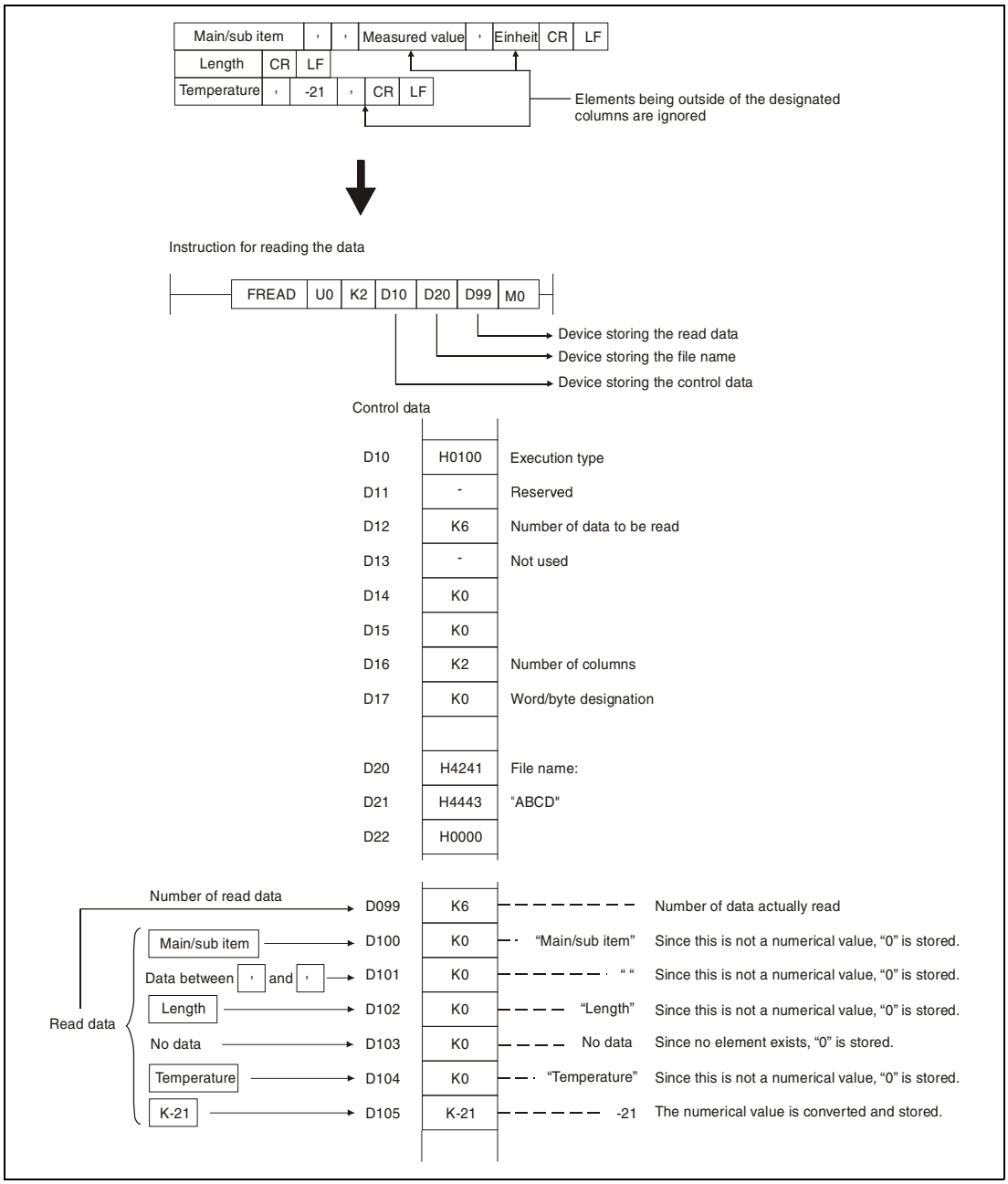
When data is read after CSV format conversion and the designated number of columns is other than „0“, the data is expected to be in a table with the specified number of columns. The elements being outside the specified columns are ignored.

The following figure illustrates such a case:



If the number of columns varies in each row, the elements ouside of the designated columns are ignored and „0“ is added to the places where elements do not exist.

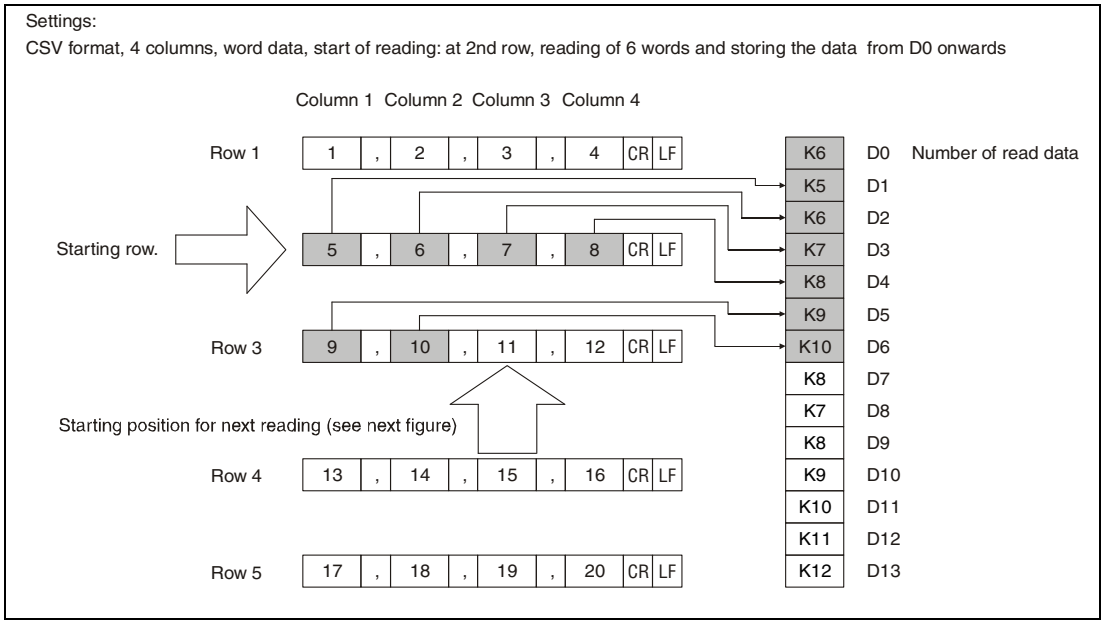
If the number of rows in the file is less than specified by (d0)+2 (Number of data to be read) „0“ is added to the places where rows do not exist.

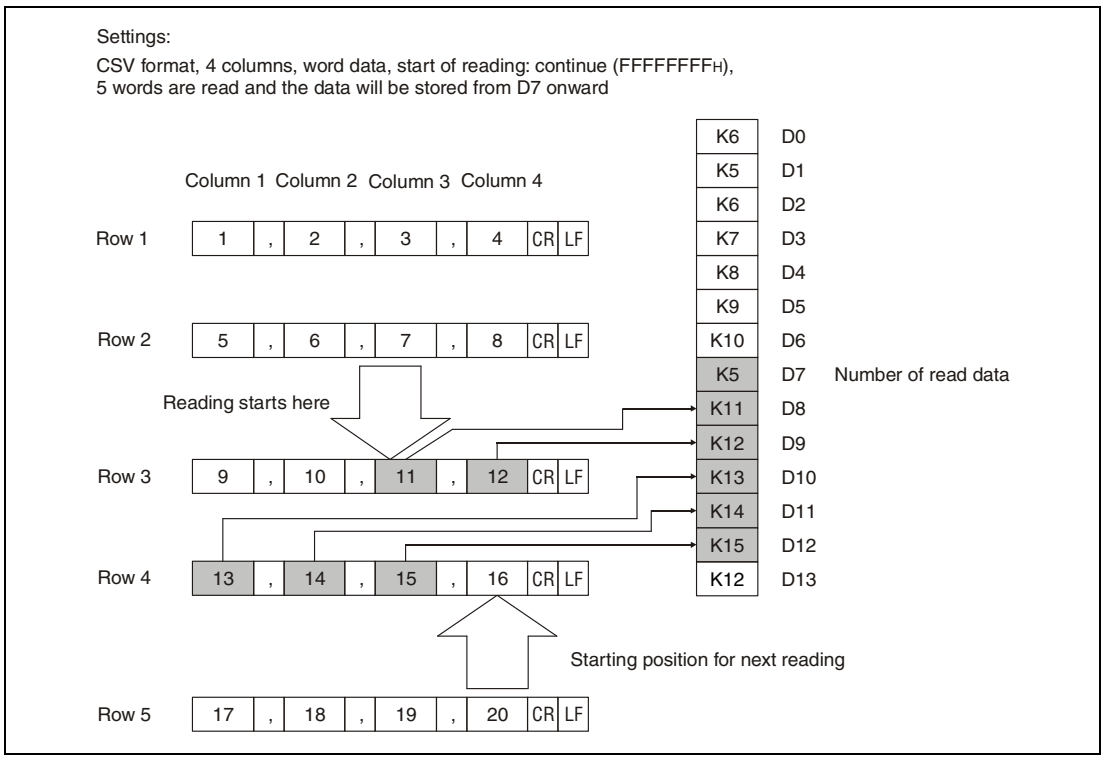


The following figures are to illustrate the case, when data is read separately several times from the same file (continuation mode) using following CPU modules:

– High Performance model QCPU with serial number "01112" or higher (first 5 digits), Process CPU, Redundant CPU or Universal model QCPU

– LCPU





When read is performed in the continuation mode, the settings for data format, number of columns and word/byte designation must not differ from the settings for the previous reading.

During reading in the continuation mode the execution of other SP.FREAD or SP.FWRITE instructions must be disabled.

When data is read after CSV format conversion, numerical values are read and converted as follows:

Numerical Values in CSV Format	Word Device	
	Without Sign	With Sign
-32768	32768	-32768
-1	65535	-1
0	0	0
1	1	1
32767	32767	32767
32768	32768	-32768
65535	65535	-1

Numerical values which are out of range and elements other than numerical values in the object CSV file are converted into „0“.

NOTE *Do not execute the SP.FREAD instruction in an interrupt program.*

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The drive specified by s0 contains a medium other than an ATA card. (for QCPU) (Error code 4100)
- The drive specified by s0 contains a medium other than the SD Memory card. (for LCPU) (Error code 4100)
- Values specified in the areas for control data are out of the setting range (excluding d0+2). (Error code 4100)
- The value „number of data to be read“ [(d0)+0] is out of the setting range. (Error code 4101)
- An invalid device is designated. (Error code 4004)
- The file name specified by s1 does not exist in the designated drive. (Error code 2410)
- Size of read data exceeds the size of the reading device. (Error code 4101)
- When binary data is read, the number of data in the file is less than the size designated by the number of data to read [(d0)+2]. (for High Performance model QCPU with serial number '01111' or lower (first 5 digits)) (Error code 4100)
- Access error occurred in the ATA card. (for QCPU) (Error code 4100)
- Access error occurred in the SD Memory card. (for LCPU) (Error code 4100)
- The device specified by d0 or d2 exceeds the range of the corresponding device. (for Universal model QCPU, LCPU) (Error code 4101)

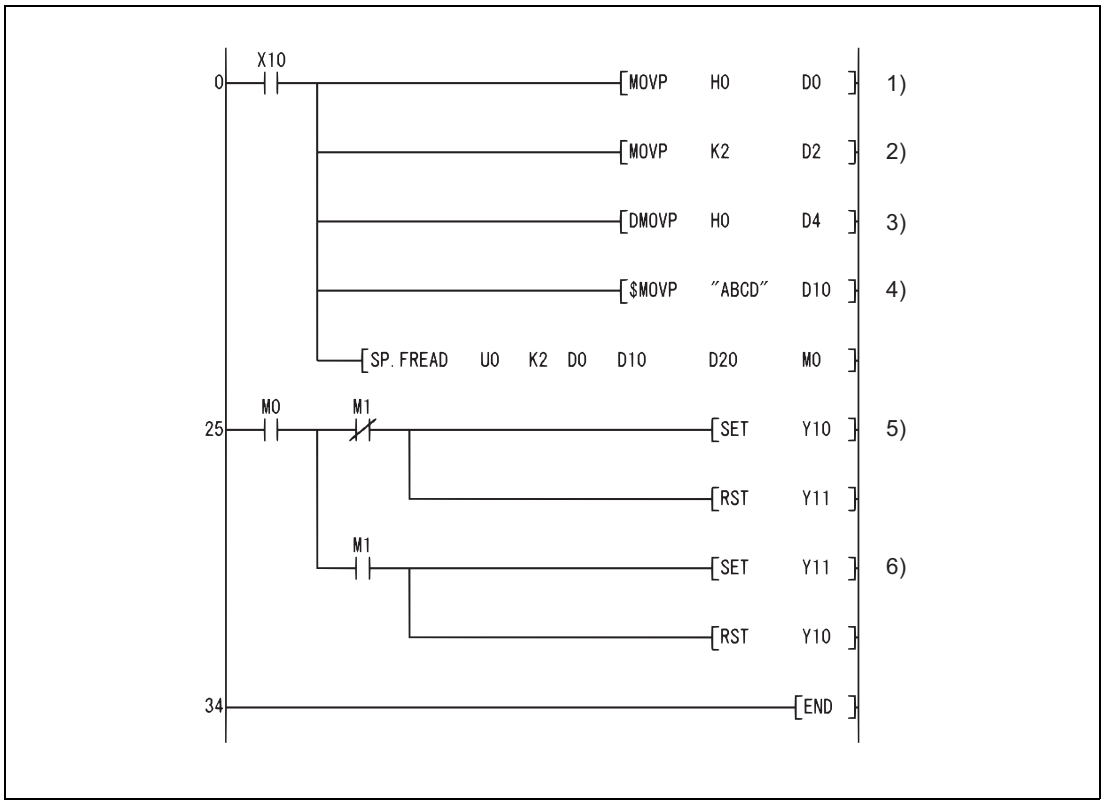
Program Example 1

SP.FREAD

When X10 is turned ON, four bytes of binary data are read from the beginning of the file „ABCD.BIN“. The file „ABCD.BIN“ is stored at a memory card which is inserted in drive 2.

From D0 onward, eight points are reserved for control data.

100 bytes are reserved from D20 for the read data.



[1] Setting of the execution/completion type
[2] Setting of the number of data to read
[3] Head address in the file (start reading at the beginning of the file)
[4] Transfer of the file name to the control data
[5] Normal completion display
[6] Error completion display

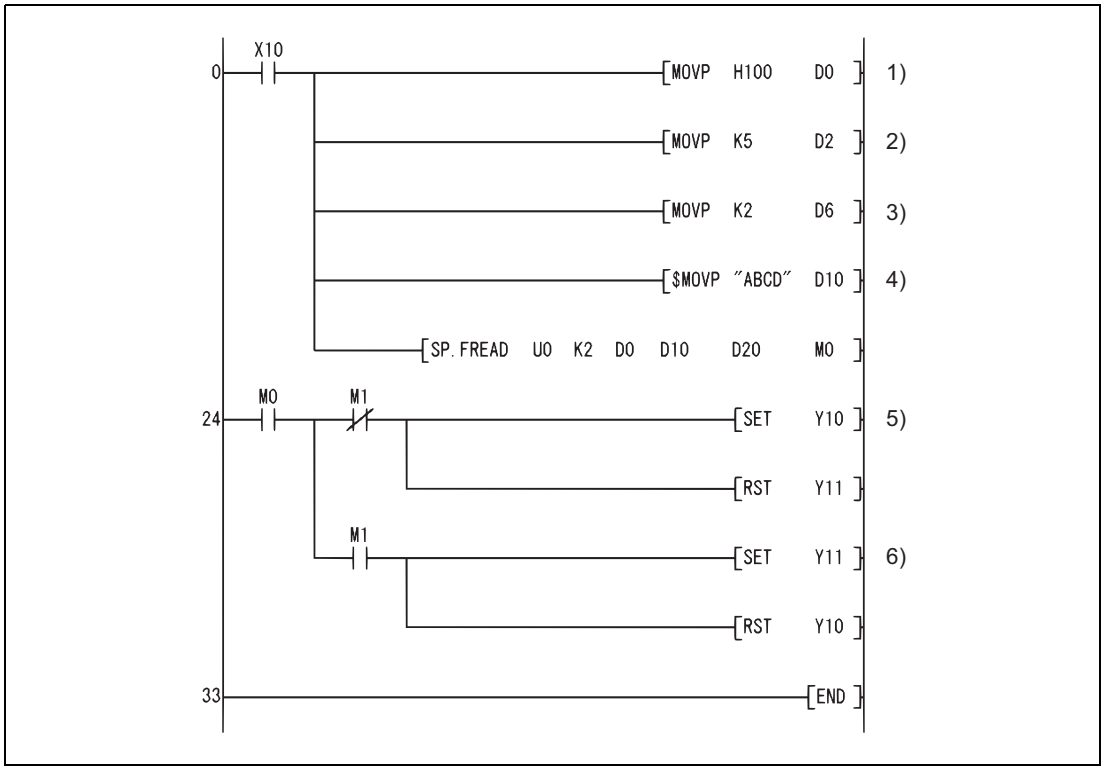
Program Example 2

SP.FREAD

The following program reads data from the file „ABCD.CSV“, which is stored at the memory card in drive 2 when X10 is turned ON. The contents of the file is two-column table data in CSV format. The file contains numerical values only.

From D0 onward, eight points are reserved for control data.

For the read data, 100 bytes are reserved from D20.



[1] Setting of the execution/completion type (CSV format for this example)

[2] Setting of the number of data to read

[3] Setting of the number of columns

[4] Transfer of the file name to the control data

[5] Normal completion display

[6] Error completion display

7.18.14 SP.DEVST

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
n1	—	●	●	—	—	—	—	●	—
s	—	●	●	—	—	—	—	—	—
n2	—	●	●	—	—	—	—	●	—
d	●[1)]	—	●[1)]	—	—	—	—	—	—

[1] Devices assigned as local devices can not be used.

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2

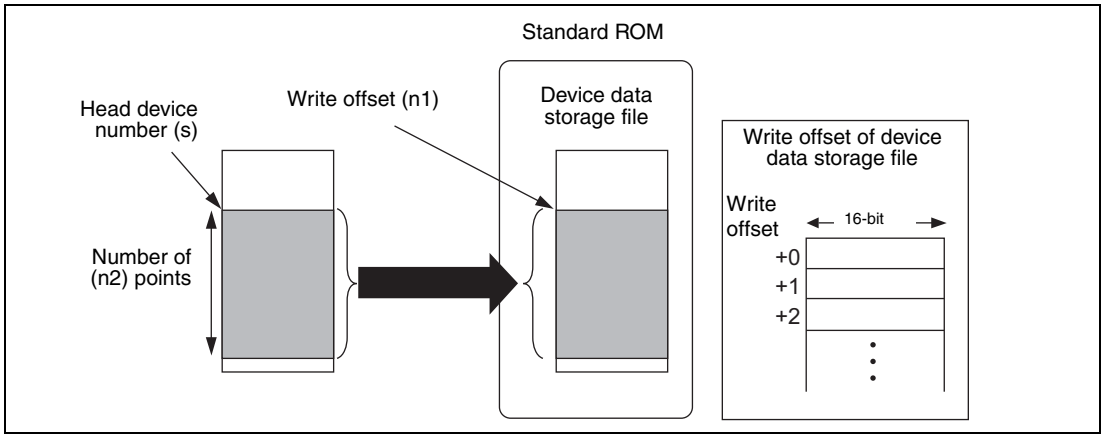
Variables

Set data	Meaning	Data Type
n1	Write offset of the device data storage file (specified in units of 16-bit words)	BIN 32-bit
s	Head device number written to the standard ROM	Device name
n2	The number of write points	BIN 16-bit
d	d+0: Completion device d+1: Error completion device	bits

Functions **Writing data to Standard ROM**

SP.DEVST **Write data**

Writes device data for the number of points specified at n2 of the device s to the write offset, which is specified for n1, of the device data storage file in the standard ROM. n1 is the offset from the head of device data storage file and specified by word offset (in units of 16-bit words).



Since the device data write position completion device (d+0) in the standard ROM automatically turns ON at execution of the END instruction, which detects the completion of this instruction, and turns OFF with the END instruction of next scan, it is used as an execution completion flag of this instruction.

When this instruction is completed in error, the error completion device (d+1) turns ON/ OFF at the same timing with the completion device (d+0). This device is used as an error completion flag of this instruction.

SM721 turns ON during execution of this instruction. When SM721 has already turned ON, this instruction can not be executed (if executed, no processing is performed). When an error is detected at execution of this instruction, the completion device (d+0), error completion device (d+1) and SM721 do not turn ON.

NOTE *The value written to the standard ROM is the value at execution of this instruction.*

The standard ROM write count index (SD687 and SD688) is increased by the execution of the SP.DEVST instruction. If the standard ROM write count index exceeds hundred thousand times, FLASH ROM ERROR (error code 1610) occurs.

To prevent the number of ROM writes from increasing due to executing instruction carelessly, set the specification of writing to standard ROM instruction count (SD695) to restrict the number of writes a day. Exceeding the number of writes (the default values are 36 times.) set causes OPERATION ERROR (error code 4113).

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The write offset specified at n1 is out of the device data storage file range.
  (Error code 4100)
- The number of n2 points from the write offset specified at n1 is out of the device data storage file range.
  (Error code 4100)
- The range for the number of n2 points from the device s exceeds the corresponding device.
  (Error code 4141)
- The device data storage file is not set at "PLC file" of PLC parameter.
  (Error code 2410)
- The device specified by d exceeds the range of the corresponding device.
  (Error code 4101)

Program Example

SP.DEVST

The following program writes the ten points of data from D100 to the device data storage file in the standard ROM when M0 turns ON.

Ladder diagram

7.18.15 S.DEVLD, SP.DEVLD

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●	●

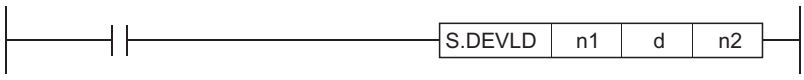
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants E	Other
	Bit	Word		Bit	Word				
n1	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n2	—	●	●	—	—	—	—	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

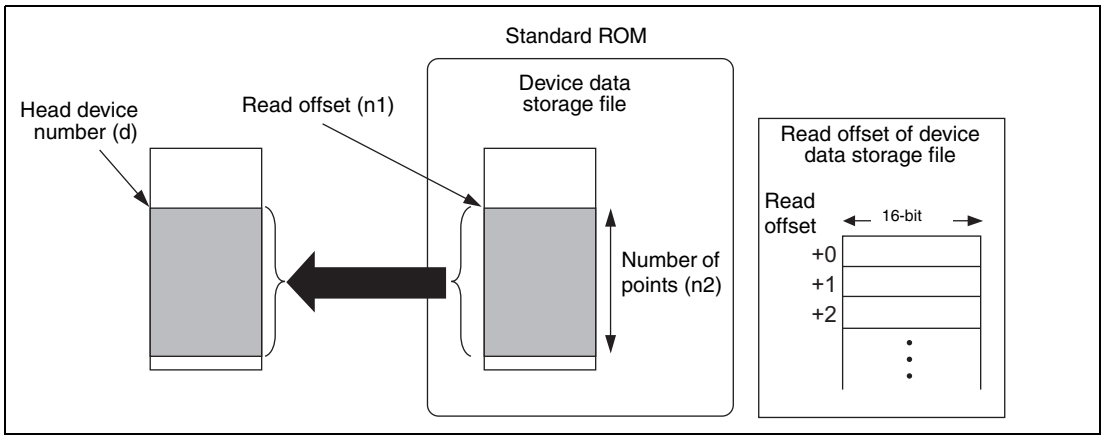
Set data	Meaning	Data Type
n1	Read offset of the device data storage file (specified in units of 16-bit words)	BIN 32-bit
d	Head device number read from the standard ROM	Device name
n2	Number of reading points.	BIN 16-bit

Functions

Read data from Standard ROM

SP.DEVLD Read data

Reads device data for the number of points specified at n2 from the read offset, which is specified for n1, of the device data storage file in the standard ROM, and stores the data to the device specified for d. n1 is the offset from the head of device data storage file and specified by word offset (in units of 16-bit words).



Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

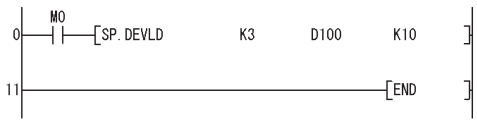
- The address specified at n1 is out of the standard ROM range. (Error code 4100)
- The number of n2 points from the address specified at n1 is out of the standard ROM range. (Error code 4100)
- The range for the number of n2 points from the device d exceeds the corresponding device. (Error code 4101)
- The device data storage file is not set at "PLC file" of PLC parameter. (Error code 2410)

Program Example

SP.DEVLD

The program which reads the ten points of data from D100 from the device data storage file in the standard ROM when M0 turns ON.

Instruction List	Ladder diagram



7.18.16 PLOADP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●			

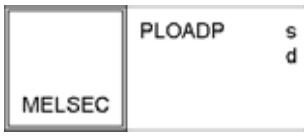
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	●[1)]	—	—	—	—	—	—	—	—

[1] Local devices cannot be used.

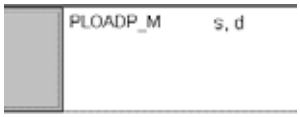
GX IEC Developer

MELSEC Instruction List

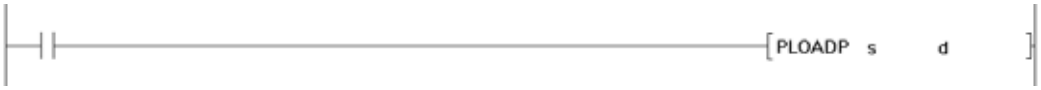


Ladder Diagram

IEC Instruction List



GX Works2



Variables

Set data	Meaning	Data Type
s	Drive number storing the program to be loaded, character string data of the file name, or head number of the device storing the character string data	BIN 16-bit
d	Device turned ON for 1 scan after completion of the instruction	bits

NOTE *The file system is not supported by the GX IEC Developer.*

Functions

Loading of a program from a memory card

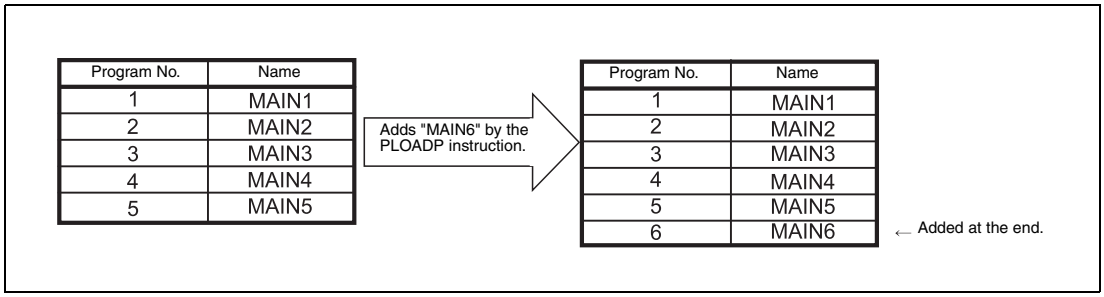
PLOADP Load program

The PLOADP instruction moves a program which is stored in a memory card or standard memory to the internal memory (drive 0) and places the program in the standby status. The memory card can be inserted in drive 1, 2 or 4. Drive 0 must have continuous free space.

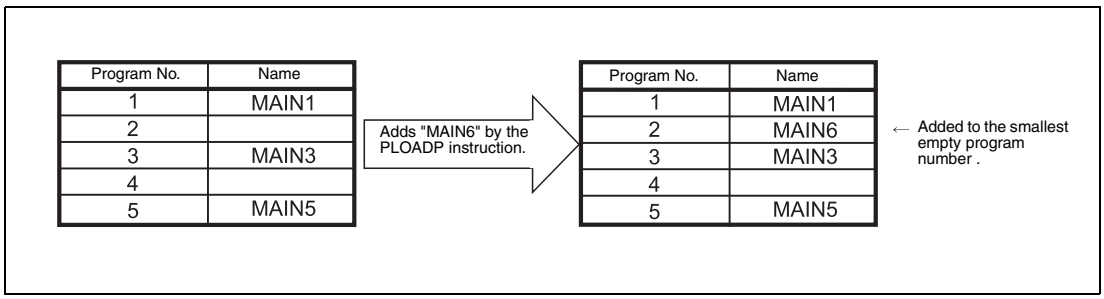
The lowest program number in the CPU which is vacant is used as the program number of the added program. The program numbers can be checked with the programming tool by reading the program list. A program number for the added program can be specified by storing a number in SD720.

The following example assumes that "MAIN6" is added by the PLOADP instruction.

When the program numbers have been set consecutively, the new program is added at the end of the preset program numbers. When programs No. 1 to 5 have been set, the new program is added as program No. 6.



When there are multiple open program numbers, the program designated by the PLOADP instruction is added to the lowest number among them to be added. (The open program numbers are made when programs are deleted by the PUNLOADP instruction.) When programs No. 2 and 4 are open, the new program is added as program No. 2.



It is unnecessary to designate the extension „.QPG“ to the file name.

The bit device specified by d goes ON during the END processing of the scan where the PLOADP instruction is completed. The bit device goes OFF during the next END processing.

The program instructions PLOADP, PUNLOADP and PSWAPP cannot be used simultaneously. If two or more instructions are executed, the instruction issued later will not be executed. Establish interlocks to avoid such a case.

The PLOADP instruction cannot be executed during an interrupt progam.

To execute the program that was transferred to the program memory with the PLOADP instruction, the PSCAN instruction must be executed.

The PLC file settings of the loaded program are set as follows:

- File usage for each program: All usage of the file register, device initial value, comment, and local device of the loaded program is set at „Use PLC file setting“.

  However, if „Use local device“ is designated in the PLC file setting and programs are loaded, an error occurs every time the number of executed programs exceeds the number of parameter-set programs.

  To use local devices in the loaded program, register a dummy file in the parameter, delete the dummy file with the PUNLOADP instruction, then load the program with the PLOADP instruction.

- I/O refresh setting:
  The I/O refresh setting for the loaded program is „Disabled“ for both input and output.

Writing during RUN is not executed during the execution of the PLOADP instruction, but executed after the instruction is completed. Conversely, the PUNLOADP instruction is not executed during the writing during RUN, but executed after the writing during RUN is completed.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The file name does not exist at the drive number specified by s.
  (Error code 2410)
- The drive number specified by s is invalid.
  (Error code 4100)
- There is not enough memory to load the specified program in drive 0.
  (Error code 2413)
- The number of programs shown below are already registered in the program memory.
  (Error code 4101)
- The program number stored in SD720 is already used, or larger than the largest program number shown below.
  (Error code 4101)

Type of CPU	Program Memory (Number of files)	Largest Program Number
Q02(H)	28	28
Q06H	60	60
Q12H Q25H Q12PH Q25PH	124	124

- A program file which has the same name as the program file to be loaded already exists.
  (Error code 2410)
- The file size of the local devices cannot be reserved.
  (Error code 2401)

Program Example

PLOADP

When M0 is ON in the following program, the program „ABCD.QPG“ is transferred from drive 4 to drive 0 and placed in standby status.

Instruction List

```
0   LD        M0
1   PLOADP    "4:ABCD" M10
7   END
```

Ladder Diagram

7.18.17 PUNLOADP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●			

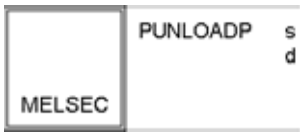
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	●[1)]	—	—	—	—	—	—	—	—

[1] Local devices cannot be used.

GX IEC Developer

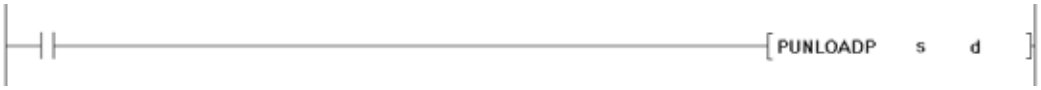
MELSEC Instruction List



Ladder Diagram

IEC Instruction List

GX Works2



Variables

Set data	Meaning	Data type
s	Character string data of the program file name to be unloaded, or head number of the device storing the character string data	BIN 16-bit
d	Device turned ON for 1 scan after completion of the instruction	Bit

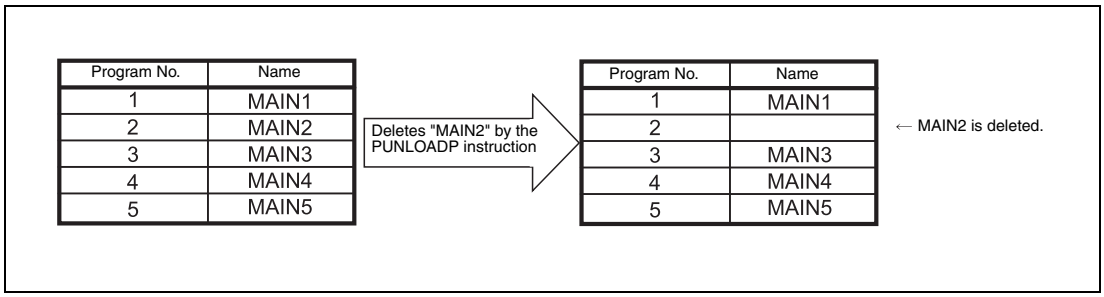
NOTE *The file system is not supported by the GX IEC Developer.*

Functions **Unloading of a program from program memory**

PUNLOADP Unload program

The PUNLOADP instruction is used to delete a standby program stored in the program memory (drive 0). The standby program being executed by the PSCAN instruction cannot be deleted.

The program No. deleted by the PUNLOADP instruction is made "Empty". In the following example program MAIN2 (program No. 2) is deleted.



It is unnecessary to designate the extension „.QPG“ to the file name.

The bit device specified by d goes ON during the END processing of the scan where the PUNLOADP instruction is completed. The bit device goes OFF during the next END processing.

The program instructions PLOADP, PUNLOADP and PSWAPP cannot be used simultaneously. If two or more instructions are executed, the instruction issued later will not be executed. Establish interlocks to avoid such a case.

If the power supply for the CPU is switched OFF and than turned ON again, or the CPU module is reset after the program deletion, the following operation is performed.

- When boot setting has been made in the PLC parameter dialog box, the program where the boot setting has been made is transferred to the program memory. When the program deleted by the PUNLOADP instruction is not to be executed, delete the corresponding program name from the boot setting and program setting of the PLC parameter dialog box.
- When boot setting has not been made in the PLC parameter dialog box, "FILE SET ERROR (error code 2400)" occurs.
  When the program deleted by the PUNLOADP instruction is not to be executed, delete the corresponding program name from the program setting of the PLC parameter dialog box.
  When the program deleted by the PUNLOADP instruction is to be executed again, write the corresponding program to the CPU module.

The PUNLOADP instruction cannot be executed during a interrupt progam.

The program to be deleted from the program memory with the PUNLOADP instruction should be placed in standby status with the PSTOP instruction before.

Writing during RUN is not executed during the execution of the PUNLOADP instruction, but ececuted after the instruction is completed. Conversely, the PUNLOADP instruction is not executed during the writing during RUN, but executed after the writing during RUN is completed.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The file name specified by s does not exist.
  (Error code 2410)
- The program designated by s is not in standby status or is being executed.
  (Error code 4101)
- The program specified by s is the only one in the program memory.
  (Error code 4101)

Program Example

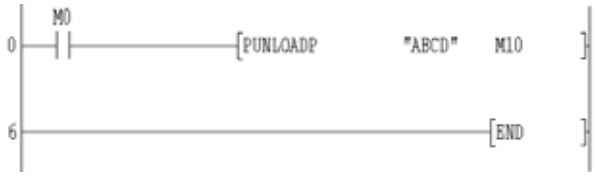
PUNLOADP (GX Works2)

The following program deletes the program „ABCD.QPG“ stored in drive 0 from the memory when M0 turns from OFF to ON.

Instruction List

```
0  LD       M0
1  PUNLOADP "ABCD"   M10
6  END
```

Ladder Diagram



7.18.18 PSWAPP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●			

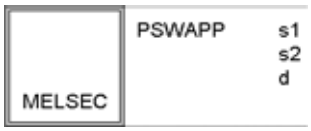
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant $	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	●	●
d	●[1)]	—	—	—	—	—	—	—	—

[1] Local devices cannot be used.

GX IEC Developer

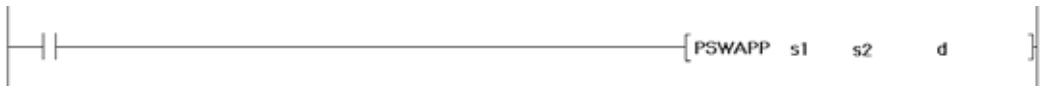
MELSEC Instruction List



Ladder Diagram

IEC Instruction List

GX Works2



Variables

Set data	Meaning	Data Type
s1	Character string data of the program file name to be unloaded, or head number of the device storing the character string data	BIN 16-bit
s2	Drive number storing the program to be loaded, character string data of the file name, or head number of the device storing the character string data	BIN 16-bit
d	Device turned ON for 1 scan after completion of the instruction	Bit

NOTE *The file system is not supported by the GX IEC Developer.*

Functions **Unloading of a program from program memory and loading of a program**

PSWAPP Unload program and load program

The PSWAPP instruction deletes (unloads) a standby program from the program memory (drive 0). The program to be deleted is specified by s1. The program set as the "scan execution type" with the PSCAN instruction or the program set as the "low speed execution type" with the PLOW instruction cannot be deleted. After the deletion, a program stored in drive 1, 2, or 4 is transferred ro the program memory and placed in standby status. This program is specified by s2. The program memory drive 0 must have continuous free space before loading the program.

The program number of the deleted program is used for the loaded program.

It is unnecessary to designate the extension „.QPG“ to the file name.

The bit device specified by d goes ON during the END processing of the scan where the PSWAPP instruction is completed. The bit device goes OFF during the next END processing.

The program instructions PLOADP, PUNLOADP and PSWAPP cannot be used simultaneously. If two or more instructions are executed, the instruction issued later will not be executed.
Establish interlocks to avoid such a case.

If the power supply for the CPU is switched OFF and than turned ON again, or the CPU module is reset after the program swap, the following operation is performed.

- When boot setting has been made in the PLC parameter dialog box, the program where the boot setting has been made is transferred to the program memory.
  When the program replaced by the PSWAPP instruction is to be executed, change the boot setting and program setting of the PLC parameter dialog box for the corresponding prgram name.
- When boot setting has not been made in the PLC parameter dialog box, "FILE SET ERROR (error code 2400)" occurs.
  When the program replaced by the PSWAPP instruction is to be executed, change the program setting of the PLC parameter dialog box for the corresponding program name.
  To execute the program set in the program setting of the PLC parameter dialog box, write the corresponding program to the CPU module again.

The PSWAPP instruction cannot be executed during a interrupt progam.

The PLC file settings of the loaded program are set as follows:

- All usage of the file register, device initial value, comment, and local device of the swapped program is set to „Use PLC file setting“.
- The I/O refresh setting for the swapped program is „Disabled“ for both input and output.

Writing during RUN is not executed during the execution of the PSWAPP instruction, but ececuted after the instruction is completed. Conversely, the PSWAPP instruction is not executed during the writing during RUN, but executed after the writing during RUN is completed.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The drive number or the file specified by s1 or s2 does not exist. (Error code 2410)
- The drive number specified by s1 is invalid. (Error code 4100)
- There is not enough capacity in the program memory (drive 0) to load the specified program. (Error code 2413)
- The program designated by s1 is not in standby status or is being executed. (Error code 4101)

Program Example

PSWAPP (GX Works2)

When M0 turns from OFF to ON in the following program example, the progam „EFGH.QPG“ is deleted from the program memory. Than the program „ABCD.QPG“ is loaded from drive 4, stored in the program memory, and placed in standby status.

Instruction List

```
  0   LD      M0
  1   PSWAPP  "EFGH"   "4:ABCD" M10
 10   END
```

Ladder Diagram

7.18.19 RBMOV, RBMOVP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●[1)]	

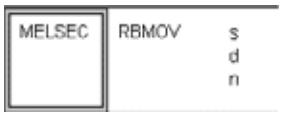
[1] Universal model QCPU: Other than Q00UJCPU

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
s	●	●	●	●	●	●	—	—	—
d	●	●	●	●	●	●	—	—	—
n	●	●	●	●	●	●	●	●	—

GX IEC Developer

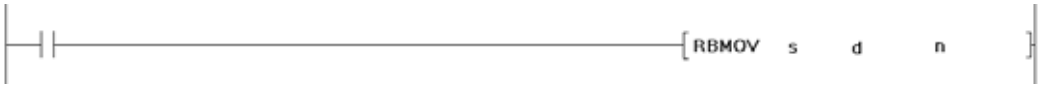
MELSEC Instruction List



Ladder Diagram

IEC Instruction List

GX Works2



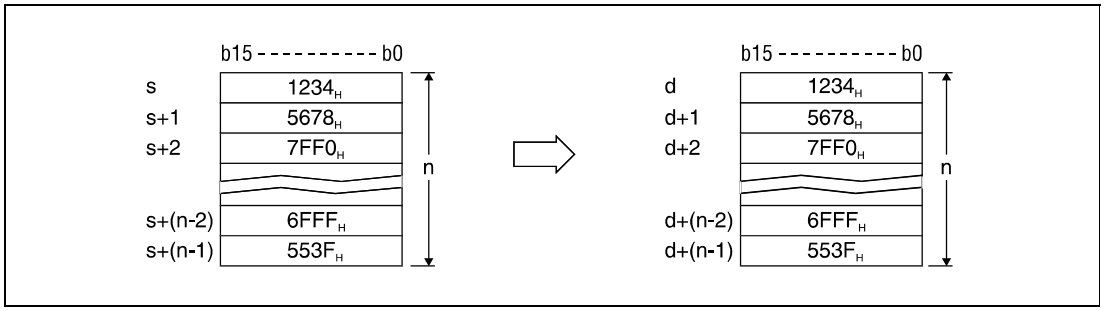
Variables

Set data	Meaning	Data type
s	Head number of the device storing the data to be transferred	BIN 16-bit
d	Head number of the destination device	
n	Number of data to be transferred	

Functions **High-speed block transfer of file register**

RBMOV/RBMOVP Block transfer

The RBMOV instruction batch transfers „n“ points of 16-bit data starting from the device specified by s to the area of „n“ points starting from the device specified by d.



The transfer is possible even if there is an overlap between the source and destination devices. For the transmission to the smaller devices, the data is transferred from s. For the transmission to the larger device number, the data is transferred from s+(n-1).

However, as shown in the example below, when transferring data from R to ZR, or from ZR to R, the range to be transferred (source) and the range of destination must not overlap.

– ZR transfer range:
((specified head No. of ZR) to
(specified head No. of ZR + the number of transfers -1))

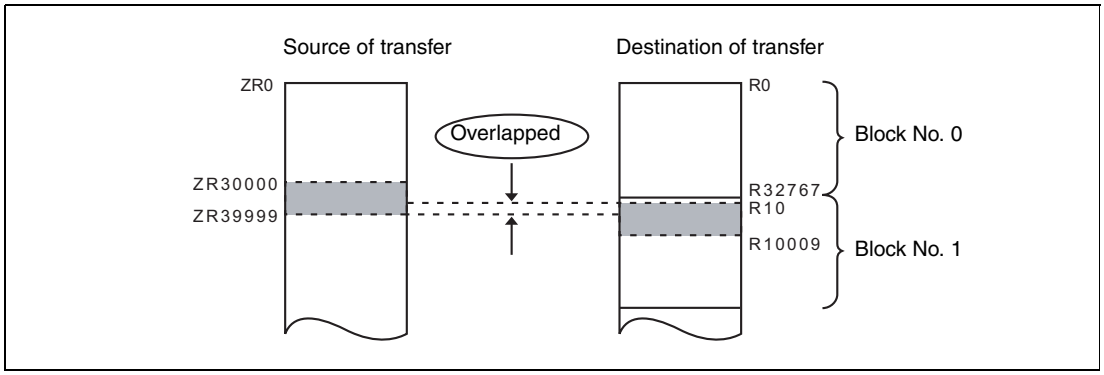
– R transfer range:
((specified head No. of R + file register block No. x 32768) to
(specified head No. of R + file register block No. x 32768 + the number of transfers -1))

Example: Transfer ranges of ZR and R overlap when transferring 10000 points of data from ZR30000 (source) to R10 (block No.1 of the destination).

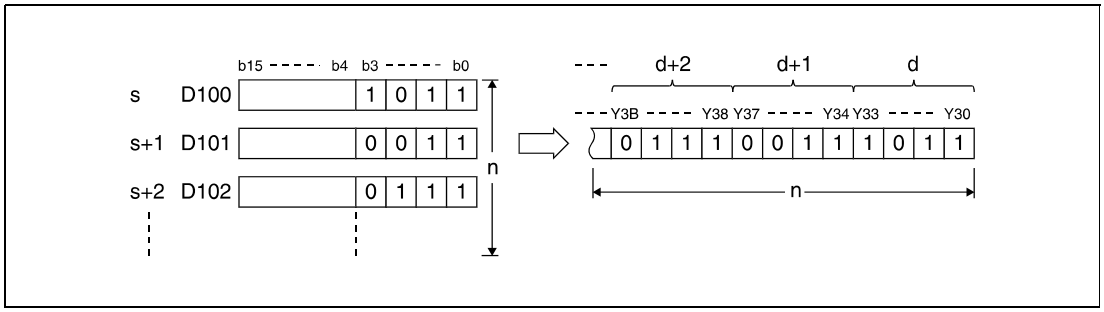
– ZR transfer range: (30000) to (30000+10000-1) = (30000) to (39999)

– R transfer range: (10+ (1 x 32768)) to (10+(1 x 32768) +10000-1)
= (32778) to (42777)

Therefore, the range 32778 to 39999 overlaps and data are not transferred correctly.



If s is a word device and d is a bit device, the object for the word device will be the number of bits designated by the bit devive digit designation. For example, when „K1Y30“ is specified by d, the lower four bits of the word device specified by s are the object.



If bit devices are specified by s and d, the number of digits must be the same for s and d.

NOTE

The RBMOV and the RBMOVP instructions are useful to batch transfer a large quantity of file register data with the QnHCPU/QnPHCPU/QnPRHCPU. With the QnUCPU, this instruction is similar to the BMOV instruction.
The comparision of processing speed between RBMOV and BMOV instructions is as follows:

- *Transfer from file registers to internal devices/internal devices to file registers*

CPU	Instruction	Target memory where File Register is stored	1 Word		1000 Words		10000 Words	
			Min.	Max.	Min.	Max.	Min.	Max.
QnHCPU QnPHCPU QnPRHCPU	RBMOV	Standard RAM	20.0 µs		91.0 µs		775.0 µs	
		SRAM card	22.0 µs		305.0 µs		2900.0 µs	
		Flash card [1]	22.5 µs		405.0 µs		3950.0 µs	
	BMOV	Standard RAM	7.5 µs		76.2 µs		720.0 µs	
		SRAM card	8.0 µs		384.0 µs		3900.0 µs	
		Flash card [1]			418.0 µs		4250.0 µs	
QnCPU	RBMOV	Standard RAM	45.5 µs		215.0 µs		1850.0 µs	
		SRAM card	49.5 µs		540.0 µs		5150.0 µs	
		Flash card [1]						
	BMOV	Standard RAM	17.5 µs		177.0 µs		1700.0 µs	
		SRAM card	18.0 µs		500.0 µs		5050.0 µs	
		Flash card [1]			572.0 µs		5800.0 µs	
Q00UCPU Q01UCPU	RBMOV	Standard RAM	12.2 µs	34.9 µs	121.5 µs	145.1 µs	1111.5 µs	1135.1 µs
		SRAM card [2]	—	—	—	—	—	—
		Flash card [2]	—	—	—	—	—	—
	BMOV	Standard RAM	7.3 µs	13.8 µs	116.5 µs	124.2 µs	1106.5 µs	1114.2 µs
		SRAM card [2]	—	—	—	—	—	—
		Flash card [2]	—	—	—	—	—	—
Q02UCPU	RBMOV	Standard RAM	9.4 µs	31.3 µs	118.5 µs	141.3 µs	1108.5 µs	1131.3 µs
		SRAM card	9.4 µs	31.4 µs	178.5 µs	201.3 µs	1708.5 µs	1731.3 µs
		Flash card [1]	9.4 µs	32.1 µs	278.5 µs	301.3 µs	2708.5 µs	2731.3 µs
	BMOV	Standard RAM	5.0 µs	11.6 µs	114.5 µs	122.3 µs	1104.5 µs	1112.3 µs
		SRAM card	5.1 µs	11.7 µs	174.5 µs	182.3 µs	1704.5 µs	1712.3 µs
		Flash card [1]	5.0 µs	11.6 µs	274.5 µs	282.3 µs	2704.5 µs	2712.3 µs
Q03UD(E)CPU	RBMOV	Standard RAM	11.3 µs	16.8 µs	120.7 µs	127.1 µs	1110.7 µs	1117.1 µs
		SRAM card	11.2 µs	16.7 µs	180.7 µs	187.1 µs	1710.7 µs	1717.1 µs
		Flash card [1]	11.3 µs	16.8 µs	280.7 µs	287.1 µs	2710.7 µs	2717.1 µs
	BMOV	Standard RAM	4.8 µs	6.6 µs	114.7 µs	117.1 µs	1104.7 µs	1107.1 µs
		SRAM card	4.8 µs	6.6 µs	147.7 µs	177.1 µs	1704.7 µs	1707.1 µs
		Flash card [1]	4.8 µs	6.5 µs	274.7 µs	277.1 µs	2704.7 µs	2707.1 µs
Q04UD(E)HCPU Q06UD(E)HCPU Q10UD(E)HCPU Q13UD(E)HCPU Q20UD(E)HCPU Q26UD(E)HCPU Q50UDEHCPU Q100UDEHCPU	RBMOV	Standard RAM	9.2 µs	15.1 µs	61.0 µs	68.6 µs	531.0 µs	538.6 µs
		SRAM card	9.4 µs	15.6 µs	165.0 µs	172.6 µs	1576.0 µs	1583.6 µs
		Flash card [1]	9.4 µs	15.7 µs	260.0 µs	267.6 µs	2526.0 µs	2533.6 µs
	BMOV	Standard RAM	4.1 µs	5.6 µs	56.0 µs	58.6 µs	526.0 µs	528.6 µs
		SRAM card	4.5 µs	6.1 µs	160.0 µs	162.6 µs	1571.0 µs	1573.6 µs
		Flash card [1]	4.3 µs	6.2 µs	255.0 µs	257.6 µs	2521.0 µs	2523.6 µs

[1] When file registers are stored in the Flash card, no processing is performed for transfer from internal devices to file registers.

[2] Unusable for the Q00UCPU and Q01UCPU.

- *Transfer from file registers to file registers*

CPU	Instruction	Target memory where File Register is stored	1 Word		1000 Words		10000 Words	
			Min.	Max.	Min.	Max.	Min.	Max.
QnHCPU QnPHCPU QnPRHCPU	RBMOV	Standard RAM	20.0 µs		91.0 µs		775.0 µs	
		SRAM card	22.5 µs		545.0 µs		5300.0 µs	
	BMOV	Standard RAM	7.5 µs		77.0 µs		720.0 µs	
		SRAM card	8.5 µs		692.0 µs		7050.0 µs	
QnCPU	RBMOV	Standard RAM	45.5 µs		215.0 µs		1850.0 µs	
		SRAM card	50.0 µs		870.0 µs		8350.0 µs	
	BMOV	Standard RAM	17.5 µs		179.0 µs		1700.0 µs	
		SRAM card	18.5 µs		839.0 µs		8600.0 µs	
Q00UCPU Q01UCPU	RBMOV	Standard RAM	12.6 µs	35.3 µs	232.5 µs	256.1 µs	2211.5 µs	2235.1 µs
		SRAM card [1)]	—	—	—	—	—	—
	BMOV	Standard RAM	7.7 µs	14.2 µs	227.5 µs	234.2 µs	2206.5 µs	2214.2 µs
		SRAM card [1)]	—	—	—	—	—	—
Q02UCPU	RBMOV	Standard RAM	9.6 µs	31.5 µs	228.5 µs	252.3 µs	2208.5 µs	2231.3 µs
		SRAM card	9.6 µs	31.5 µs	378.5 µs	401.3 µs	3708.5 µs	3731.3 µs
	BMOV	Standard RAM	5.2 µs	11.8 µs	224.5 µs	232.3 µs	2204.5 µs	2212.3 µs
		SRAM card	5.2 µs	11.8 µs	374.5 µs	382.3 µs	3704.5 µs	3712.3 µs
Q03UD(E)CPU	RBMOV	Standard RAM	11.2 µs	16.7 µs	230.7 µs	237.1 µs	2210.7 µs	2217.1 µs
		SRAM card	11.6 µs	16.7 µs	380.7 µs	387.1 µs	3710.7 µs	3717.1 µs
	BMOV	Standard RAM	4.9 µs	6.7 µs	224.7 µs	227.1 µs	2204.7 µs	2207.1 µs
		SRAM card	5.2 µs	6.7 µs	374.7 µs	377.1 µs	3704.7 µs	3707.1 µs
Q04UD(E)HCPU Q06UD(E)HCPU Q10UD(E)HCPU Q13UD(E)HCPU Q20UD(E)HCPU Q26UD(E)HCPU Q50UDEHCPU Q100UDEHCPU	RBMOV	Standard RAM	9.3 µs	15.5 µs	118.0 µs	124.6 µs	1102.0 µs	1107.6 µs
		SRAM card	9.7 µs	15.5 µs	365.0 µs	371.6 µs	3571.0 µs	3578.6 µs
	BMOV	Standard RAM	4.3 µs	6.2 µs	113.0 µs	115.6 µs	1096.0 µs	1098.6 µs
		SRAM card	4.5 µs	6.1 µs	360.0 µs	362.6 µs	3566.0 µs	3568.6 µs

[1] Unusable for the Q00UCPU and Q01UCPU.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device range of „n“ points starting from s or d exceeds the available device. (Error code 4101)
- The file register is not designated for both s and d. (Error code 4101)

Program Example 1

RBMOVP

The following program transfers the lower four bits (b0 through b3) of data in D66 through D69 to the outputs Y30 through Y3F with the rising edge of SM402. The number of data (4 blocks) is specified by n.

The bit patterns show the structure of bits before and after the transfer.

Instruction List

```
0   LD      SM402
1   RBMOVP  D66     K1Y30   K4
5   END
```

Ladder Diagram

	b15 ----- b4	b3 ----- b0								
D66		1	1	1	0	1	1	1	0	1	Y33 - Y30
D67		0	0	0	0	0	0	0	0	0	Y37 - Y34
D68		1	0	0	1	1	0	0	1	1	Y3B - Y38
D69		0	1	1	0	1	1	1	0	1	Y3F - Y3C
		1								

[1] These bits are ignored.

Program Example 2

RBMOVP

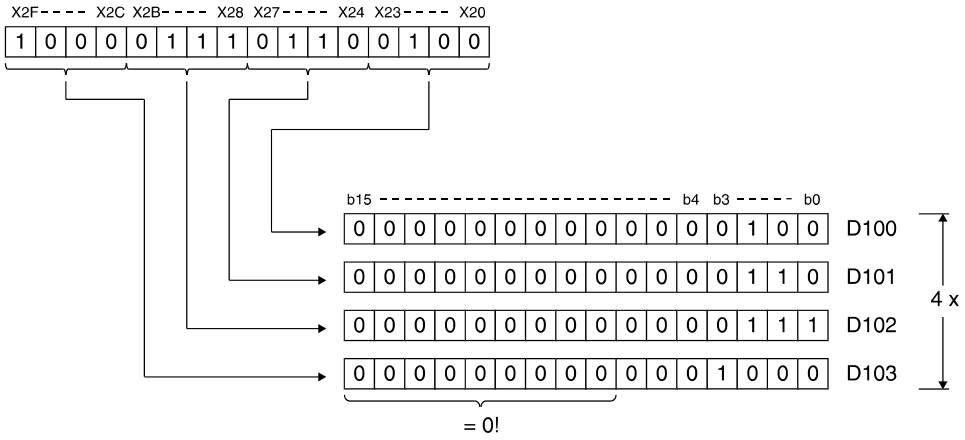
With leading edge from SM402, the following program transfers data at X20 through X2F to D100 through D103. The number of blocks (4) to be transferred is determined by the constant K4.

The bit patterns show the structure of bits before and after the transfer.

Instruction List

```
0  LD      SM402
1  RBMOVP  K1X20   D100    K4
5  END
```

Ladder Diagram



7.18.20 UMSG

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
					●

Devices

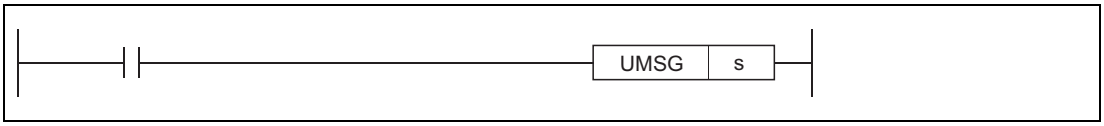
	Usable Devices									
	Internal Devices (System, User)		File-Register	Indirect specifications	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants $	Other
	Bit	Word			Bit	Word				
s	—	●	●	●	—	—	—	—	●[1)]	—

[1] Only strings can be used

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



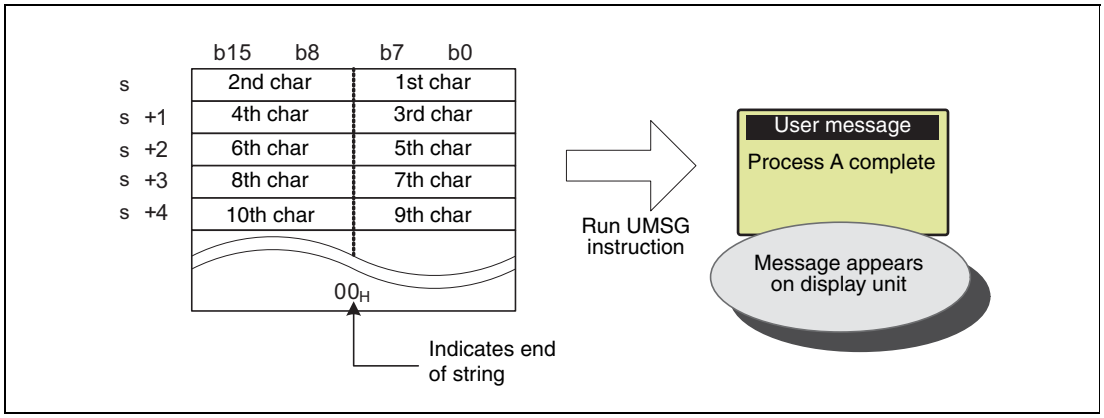
Variables

Set data	Meaning	Data Type
s	String to display on display unit, or lead number (string) of device storing string to display	—

Functions **User message**

UMSG User message

The string data specified by s is displayed as a user message in the display unit.
The string specified directly by s (surrounded by double quotation marks (")) or the string from the device number specified by s until the device number storing "00H" is displayed.



Strings of up to 128 single-byte characters can be displayed in the display unit.

The user message is displayed when the UMSG instruction command is rising.

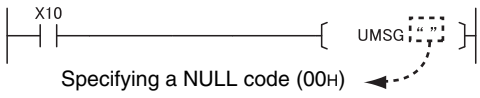
If the string is changed while the command is on, then the modified user message will appear in the display unit.

The string specified by the UMSG instruction is displayed upon END processing. If two or more UMSG instructions are executed, then the last UMSG instruction executed before the END is valid. If two or more programs are running, then the last UMSG instruction to be executed is valid.

This instruction is not processed if it is run when no display unit is mounted.

If the "ESC" key on the display unit is pressed while a user message is being displayed, the displayed message will disappear. To display the message again, execute "User Message" from the menu screen on the display unit.

If a NULL code (00H) is specified as the argument to this instruction, then any message currently being displayed will disappear. The procedure for specifying a NULL code (00H) in the instruction parameter is as follows:



See the MELSEC-L CPU Module User's Manual (Function Explanation, Program Fundamentals) for details about the display unit.

Operation Errors

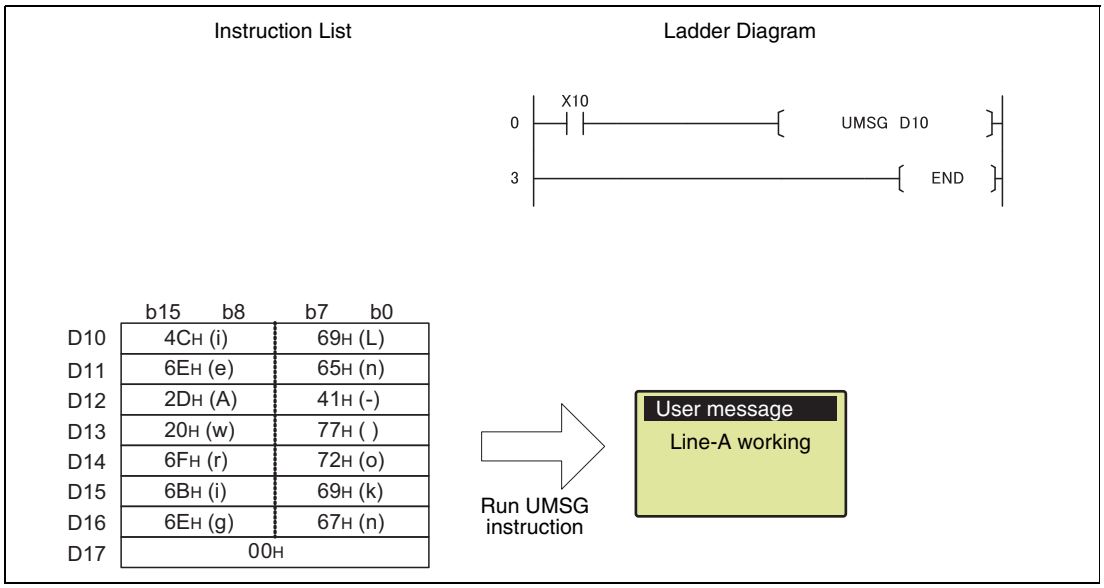
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- When there is no NULL code (00H) within the range of the target device following the device number specified by s.
  (Error code 4101)
- When more than 128 single-byte characters are specified in the s string.
  (Error code 4100)

Program Example 1

UMSG

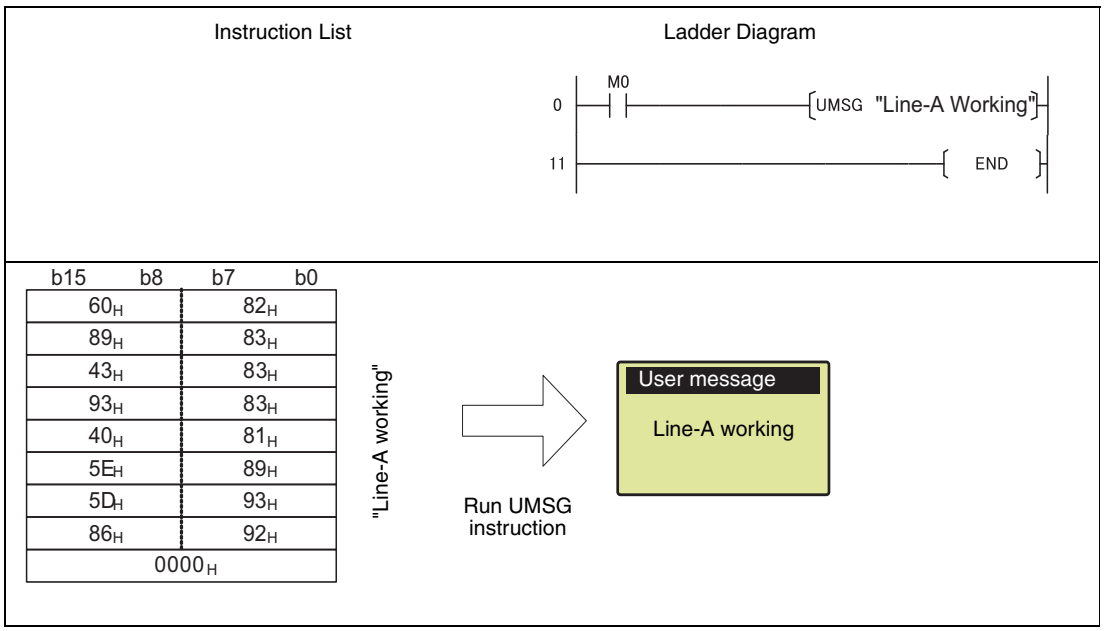
This program displays the string stored after D10 on the display unit, when X10 is set to "on".



Program Example 2

UMSG

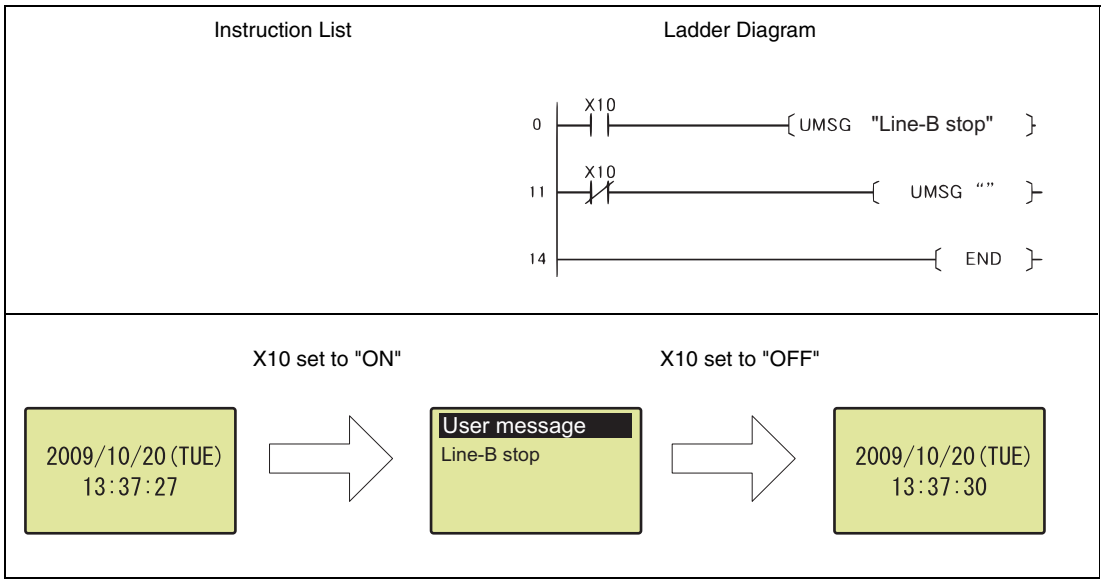
This program displays "Line-A Working" on the display unit when M0 is set to "on".



Program Example 3

UMSG

This program displays "Line-B stop" on the display unit when X10 is set to "on", and clears the message when X10 is set to "off".



8 Data Link Instructions

8.1 Categories of instructions

The following table gives an overview of the data link instructions:

Category	Meaning
Network refresh instructions	Instructions for data refresh operations in network modules.
Read/Write routing information	Read and write routing parameters (network number and station number of relay station, station number of routing station).

8.2 Data refresh instructions

The following instructions refresh data in network modules. The following table gives an overview of the instructions:

Function	MELSEC-Instruction in MELSEC-Editor	MELSEC-Instruction in IEC-Editor
Instructions for Network refresh	S.ZCOM	
	SP.ZCOM	
	S.ZCOM	
	SP.ZCOM	

The ZCOM instruction is used to perform refresh at any timing during execution of a sequence program.

The targets of refresh performed by the ZCOM instruction are indicated below.

- Refresh of CC-Link IE (when refresh parameters are set) (QCPU only)
- Refresh of MELSECNET/H (when refresh parameters are set) (QCPU only)
- Auto refresh of CC-Link (when refresh device is set)
- Auto refresh of intelligent function module (when auto refresh is set)

NOTE *In this section, instruction names are abbreviated as follows if not specified particularly: S(P).ZCOM ==> ZCOM.*

8.2.1 S.ZCOM, SP.ZCOM

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	●

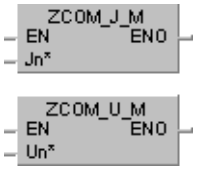
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
—	—	—	—	—	—	—	—	—	—

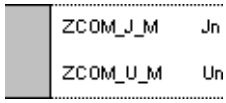
GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List



GX Works2

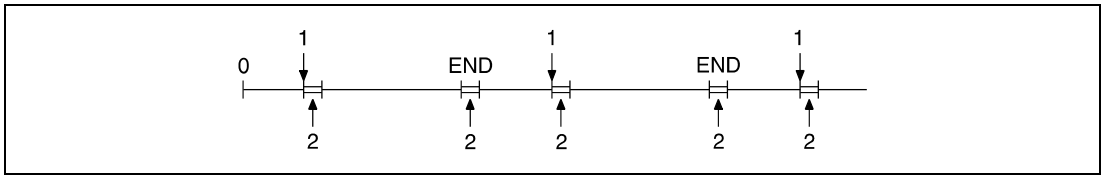
Variables

Set Data	Meaning	Data Type
Jn	Network number for host station (QCPU only).	BIN 16-bit
Un	Head I/O number of host station network.	

Functions **Network data refresh**

S.ZCOM **Data refresh in network modules**

On execution of the ZCOM instruction the CPU suspends processing the sequence program and refreshes the data in the network modules specified by Jn and Un.
(Specification cannot be made via Jn for LCPU.)



[1] Execution of the ZCOM instruction

[2] Data refresh

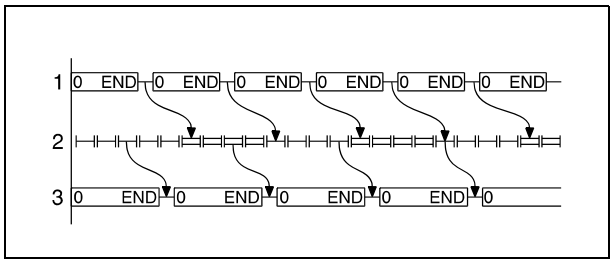
The ZCOM instruction does not perform the following processing.

- Communication processing between CPU module and programming tool
- Monitor processing of other station
- Read processing of buffer memory of other intelligent function module by serial communication module.
- Low-speed cyclic data transmission of MELSECNET/H

PLC to PLC network (Controller network in CC-Link IE controller network) (QCPU only)

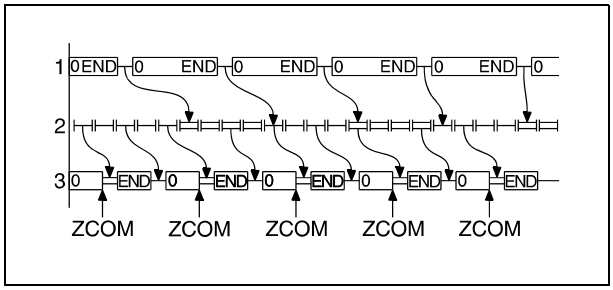
In cases where the scan time of the sequence program of the host station exceeds the scan time of the other stations, the ZCOM instruction ensures that the data from the other station is incorporated properly.

The following figure shows an example for data communication without applying the ZCOM instruction:



[1] Program of the control station
[2] Program scan of the linked station
[3] Program of the normal station

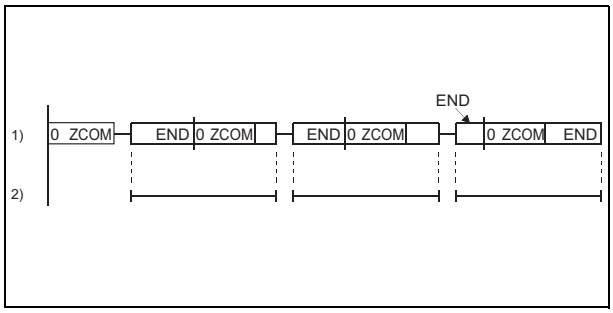
The following figure shows an example for data communication applying the ZCOM instruction:



[1] Program of the control station
[2] Program scan of the linked station
[3] Program of the normal station

For details of the transmission delay time on the PLC to PLC network (Controller network in CC-Link IE controller network), refer to the corresponding manuals of the network modules.

In cases where the scan time of the object station exceeds the scan time of the sequence program, the ZCOM instruction does not improve data communication.



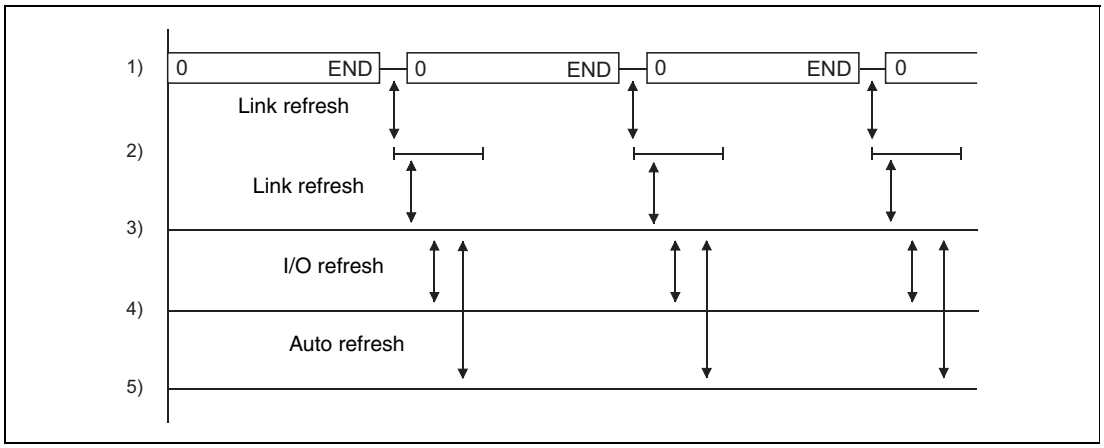
[1] Sequence program
[2] Scan time of the object station

Remote I/O network (QCPU only)

The link refresh of the remote master station is performed by the "END processing" of the CPU module. Since link scan is performed at completion of link refresh, link scan 'synchronizes' with the program of the CPU module.

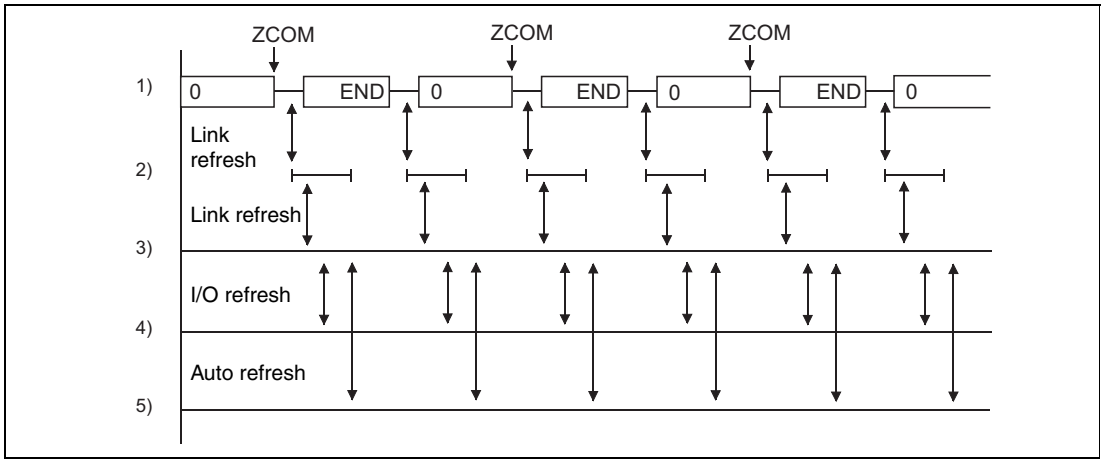
When the ZCOM instruction is used at the remote master station, link refresh is performed at the point of ZCOM instruction execution, and link scan is performed at completion of link refresh. Hence, use of the ZCOM instruction at the remote master station speeds up send/receive processing to/from the remote I/O station.

The following figure shows an example for data communication without applying the ZCOM instruction:



[1] Program of the remote master station
[2] Link scan
[3] Remote I/O station network refresh
[4] I/O module
[5] Intelligent function module

The following figure shows an example for data communication applying the ZCOM instruction:



[1] Program of the remote master station

[2] Link scan

[3] Remote I/O station network refresh

[4] I/O module

[5] Intelligent function module

The ZCOM instruction may be executed any times within a sequence program. However, note that each execution increases the scan time of the sequence program by the execution time of the data refresh.

Designating "Un" in the argument enables the access not only to network modules but also to intelligent function modules. In this case, the automatic refresh is performed for the buffer memory of the intelligent function module (replaces the FROM/TO instructions).

Only with the Universal model QCPU and LCPU, interruption of processing is enabled during the execution of the ZCOM instruction. However, when refresh data are used in an interrupted program, the data can split.

NOTES

The ZCOM instruction cannot be used in a fixed cycle execution type program or interrupt program.

The Redundant CPU has restrictions on use of the ZCOM instruction. Refer to the manual of the redundant system for details.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

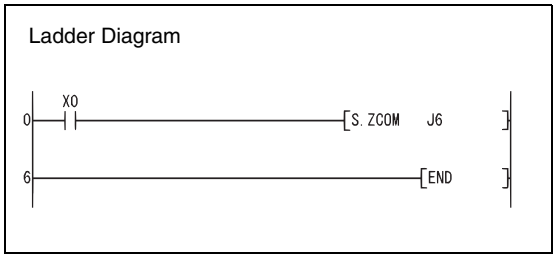
- The specified network number is not connected to the host station.
  (Error code 4102)
- The module for the specified I/O number is not a network unit or link unit.
  (Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU)
  (Error code 2111)
- The module for the specified I/O number is not a network unit or link unit.
  (Universal model QCPU, LCPU)
  (Error code 4102)

NOTE *To conduct only communication with peripheral device, use the COM instruction.*

Program Example 1

S.ZCOM

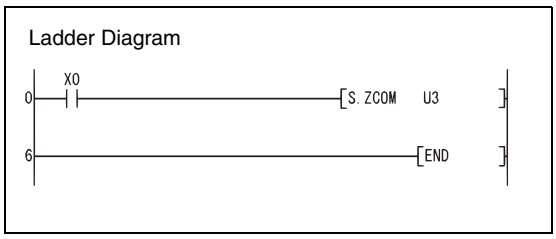
While X0 is set, the following program refreshes data in the network module with the network number 6.



Program Example 2

S.ZCOM

While X0 is set, the following program refreshes data in the network module at the I/O numbers X/Y30 through X/Y4F.



8.3 Reading and writing routing information

These instructions read and write routing information. The routing parameters comprise network and station number of the relay station and the station number of the routing station.

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Read routing information	S.RTREAD	RTREAD_M
	SP.RTREAD	RTREADP_M
Write routing information	S.RTWRITE	RTWRITE_M
	SP.RTWRITE	RTWRITEP_M

NOTE *In this section, instruction names are abbreviated as follows if not specified particularly:*

- *S(P).RTREAD ==> RTREAD*
- *S(P).RTWRITE ==> RTWRITE*

8.3.1 S.RTREAD, SP.RTREAD

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	

Devices

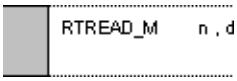
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
n	●	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List

Ladder Diagram

IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
n	Destination network of transmission (1 to 239).	BIN 16-bit	ANY16
d	First number of device storing read routing information.	Device number	Array [0..2] of ANY16

Functions

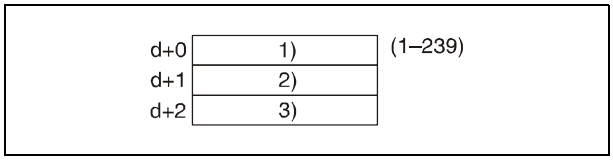
Reading routing information

S.RTREAD Read instruction

The S.RTREAD instruction reads the routing information from the destination network specified by n. The routing information is stored in routing parameters. The read routing information is stored from d+0 (Array_d[0]) onwards.

If no data is specified for the transmission the value 0 is written to the devices specified from d on (Array_d[0] through Array_d[2]).

The figure below shows the contents specified from d+0 (Array_d[0]) on:



[1] Network number of relay station
[2] Station number of relay station (see table below for range)
[3] Dummy

Network type	Specification range for relay station number
MELSECNET/H	1 to 64
CC-Link IE controller network	1 to 120
CC-Link IE field network	• Master station: Fixed at 125. (The fixed value is stored.) • Local station: 1 to 120 (A station number is stored.)

Operation Errors

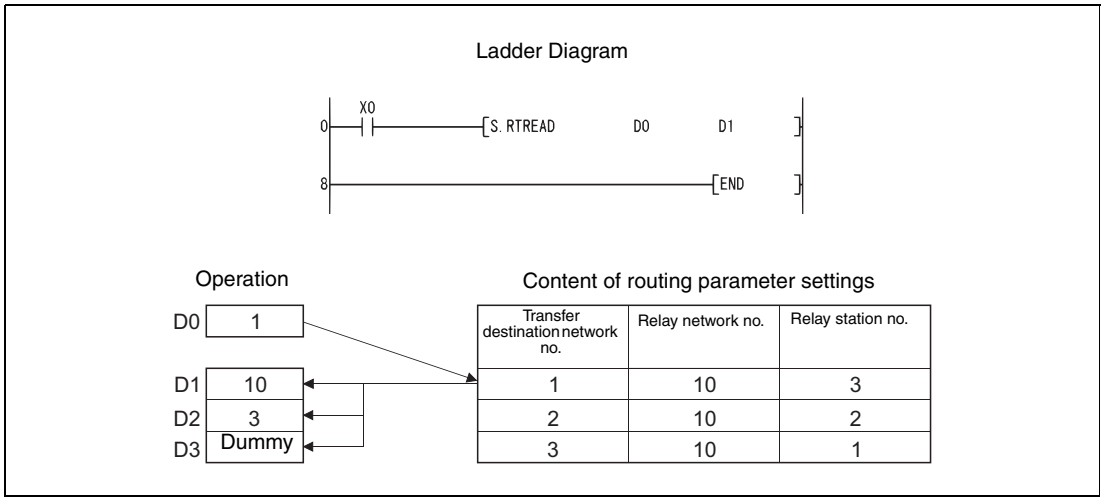
In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data value specified for n does not range within 1 and 239. (Error code 4100)
- The device specified by d exceeds the range of the corresponding device. (For the Universal model QCPU only) (Error code 4101)

Program Example

S.RTREAD

While X0 is set, the following program reads the routing information from the network specified by D0.



NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

8.3.2 S.RTWRITE, SP.RTWRITE

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
	●	●	●	●	

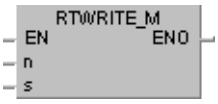
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
n	●	●	●	—	—	—	—	●	—
s	—	●	●	—	—	—	—	—	—

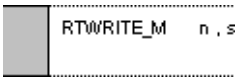
GX IEC Developer

MELSEC Instruction List

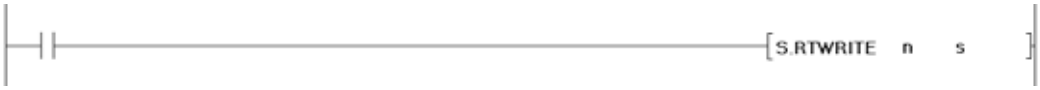
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Data Type	
		MELSEC	IEC
n	Destination network of transmission (1 to 239).	BIN 16-bit	ANY16
s	First number of device storing routing information to be written.	Device number	Array [0...2] of ANY16

Functions

Writing routing information

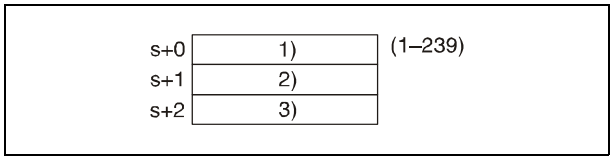
S.RTWRITE Write instruction

The S.RTWRITE instruction writes the routing information to the destination network specified by n. The routing information is stored in routing parameters. The read routing information is stored from s+0 (Array_s[0]) onwards.

If data for the destination network is set in the routing parameters, it is used to refresh the data stored from s+0 (Array_s[0]) on.

If all data in s or later (s+0 to s+2) is 0, the data for the transfer destination network number specified by n is deleted from the routing parameters.

The figure below shows the contents specified from s+0 (Array_d[0]) on:



[1] Network number of relay station
[2] Station number of relay station (see table below for range)
[3] DUMMY

Network type	Specification range for relay station number
MELSECNET/H	1 to 64
CC-Link IE controller network	1 to 120
CC-Link IE field network	• Master station: Fixed at 125. (The fixed value is stored.) • Local station: 1 to 120 (A station number is stored.)

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data value specified for n does not range within 1 and 239. (Error code 4100)
- The data specified by s exceed the relevant ranges. (Error code 4100)
- When the total number of routing information registered in the routing parameter of the network parameters and routing information registered with the RTWRITE instruction exceeds 64. (Error code 4100)
- The device specified by s exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code 4101)

Program Example

S.RTWRITE

While X0 is set, the following program writes the routing information stored in D1 through D3 as routing parameters to the network specified by D0.

Ladder Diagram

Operation

D0	1
D1	20
D2	3
D3	Dummy

Content of routing parameter settings

Transfer destination network no.	Relay network no.	Relay station no.
1	20	1
2	10	2
3	10	1

NOTE

This program example will not run without variable definition in the header of the program organization unit (POU). It would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

9 Multiple CPU Dedicated Instructions

Following instructions are available for use in a multi-CPU system:

Function	MELSEC-Instruction in MELSEC-Editor	MELSEC-Instruction in IEC-Editor
Write to CPU shared memory of host station	S.TO	TO_S_M
	SP.TO	TO_SP_M
	TO	
	TOP	
	DTO	
	DTOP	
Read from CPU shared memory of another station	FROM	FROM_M
	FROMP	FROMP_M
	DFRO	
	DFROP	

9.1 Writing to the CPU shared memory of host CPU

The S.TO or TO instruction is used to write to the CPU shared memory of the host station in the multiple CPU system.

The following table indicates the usability of the S.TO and TO instructions.

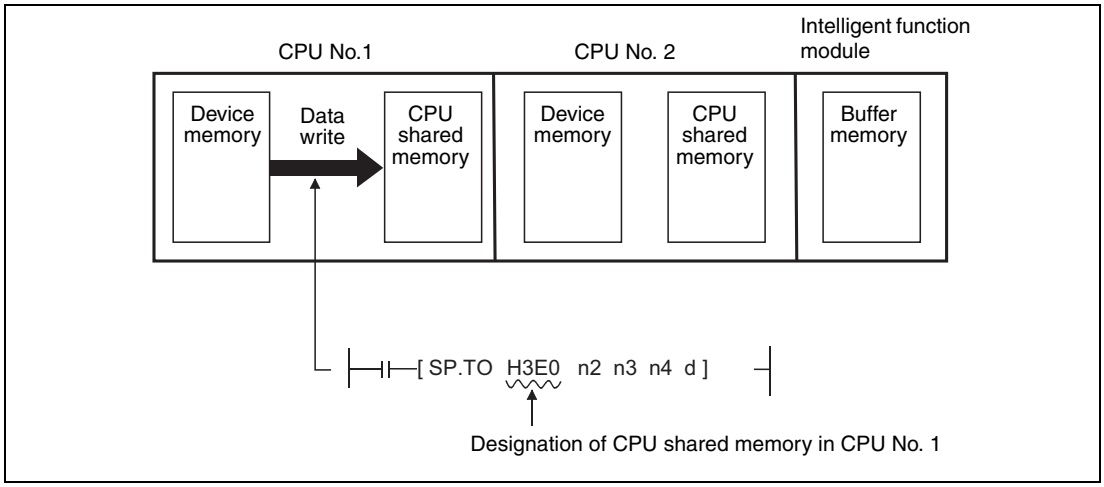
CPU Module Type Name		S.TO Instruction	TO Instruction
Basic model QCPU	Q00JCPU	—	—
	Q00CPU, Q01CPU	●	●
High Performance model QCPU	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	●	—
Process CPU	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	●	—
Redundant CPU	Q12PRHCPU, Q25PRHCPU	—	—
Universal model QCPU	Q00UJCPU	—	—
	Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU	●	●
LCPU	L02CPU, L26CPU-BT	—	—

● Usable

— Not usable

Operation of S.TO instruction

The S.TO instruction can write data to the CPU shared memory of the host CPU module. The following figure shows the processing performed when the S.TO instruction is executed in CPU No. 1.

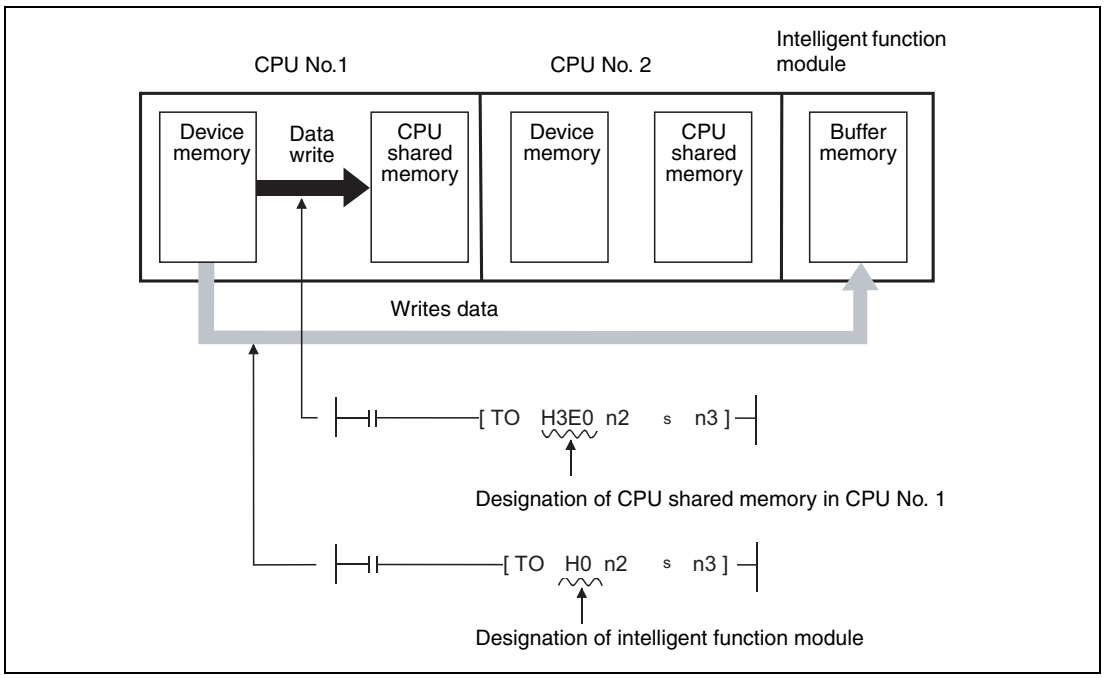


Operation of the TO instruction

The TO instruction can write device memory data to the following memories.

- CPU shared memory of host CPU module
- Buffer memory of intelligent function module

The following figure shows the processing performed when the TO instruction is executed in CPU No. 1.



NOTE *Both of the S.TO and TO instructions can be used for the Basic model QCPU (Q00CPU or Q01CPU) and Universal model QCPU to write data to the CPU shared memory. However, use of the TO instruction is recommended, since use of S.TO instruction increases the number of steps and processing time.*

Refer to section 7.8.2 when writing to the buffer memory of the intelligent function module by the TO instruction.

9.1.1 S.TO, SP.TO

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●[2)]	●		●	

[1] Basic model QCPU:The first 5 digits of serial No is "04122" or higher.

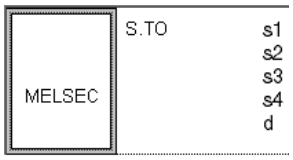
[2] High performance model QCPU: Function version B or later.

Devices

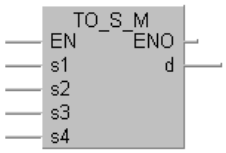
	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				
n1	—	●	●	—	—	—	—	●	—
n2	—	●	●	—	—	—	—	●	—
n3	—	●	●	—	—	—	—	—	—
n4	—	●	●	—	—	—	—	●	—
d	●	●	●	—	—	—	—	—	—

GX IEC Developer

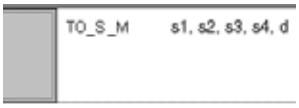
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

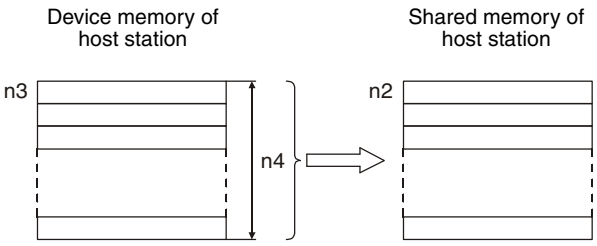
Set Data	Meaning	Data Type
n1	Head I/O number of the host CPU	BIN16-bit
n2	First number of CPU shared memory address area to be written to • Basic model QCPU: 0 to 511 • High Performance model QCPU, Process CPU, Universal model QCPU: 0 to 4095	
n3	First number of device area storing data to be written.	
n4	Number of data to be written • Basic model QCPU: 1 to 320 • High Performance model QCPU, Process CPU: 1 to 256 • Universal model QCPU: 1 to 2048	
d	Bit device which is turned ON for one scan after the instruction is executed	Bit

Functions **Writing data to the CPU shared memory**

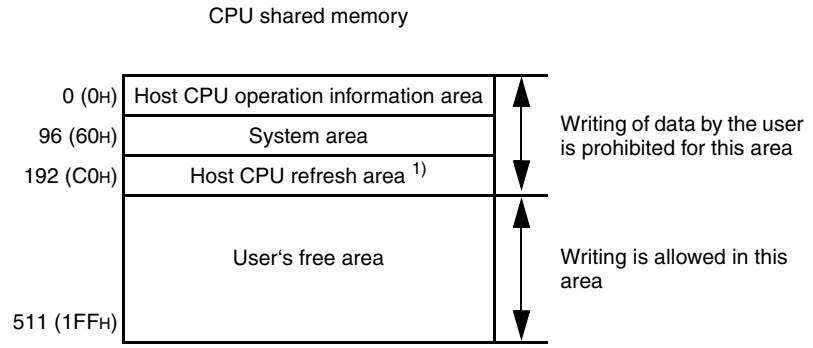
S.TO/SP.TO **Write data**

The S.TO instruction writes data to the user's area in the shared memory of the CPU which is executing the S.TO instruction (host station). The destination adress in the shared memory is entered in n2. The data is taken from a device area in the same CPU, starting from the number specified in n3. The number of data words is specified in n4.

The S.TO instruction cannot be used for writing data directly to another CPU in a multi-CPU system.

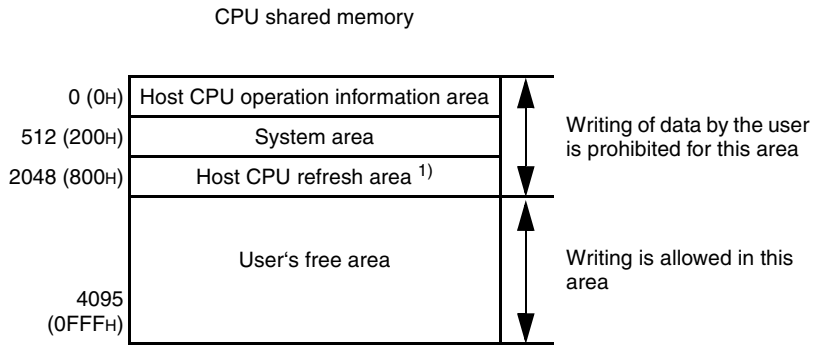


- CPU shared memory address of the Basic model QCPU



1 Usable as a user free area when auto refresh setting is not made. In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.

- CPU shared memory address of the High Performance model QCPU, Process CPU and Universal model QCPU (Data cannot be written to the multiple CPU high speed transmission area of the Universal model QCPU with the S(P).TO instruction)



1 Usable as a user free area when auto refresh setting is not made. In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.

When the number of write points is entered in n4 as „0“, processing of the instruction is not performed and the completion device, specified in d, does not turn on, either.

Only one S.TO instruction may be executed in one scan by each CPU. However, automatic handshaking makes sure that only the instruction called first will be processed, if two or more S.TO instructions are enabled simultaneously.

The number of data that can be written varies depending on the target CPU module.

CPU Module	Number of Write Points
Basic model CPU	1–320
High Performance model QCPU, Process CPU	1–256
Universal model QCPU	1–2048

The head I/O number of the CPU is determined by the slot in which the CPU module is loaded. Only the first 3 digits of the head I/O number are entered in n1.

Slot of the base unit	CPU	0	1	2
Number of the CPU in multi-CPU system	1	2	3	4
Head I/O number	3E00	3E10	3E20	3E30
Contents of n1	3E0	3E1	3E2	3E3

NOTE *Writing data to CPU shared memory can be performed using the intelligent function module device.*
For intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

Operation Errors

In the following cases an operation error occurs, the error flag is set, and the corresponding error code is stored in SD0:

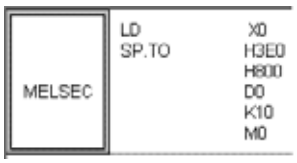
- When the specified data is outside the following range. (Error code 4101)
  - The number of write points specified in n4 is outside the specified range of the setting data.
  - The beginning of the CPU shared memory specified in n2 is larger than the CPU shared memory adress range.
  - The beginning of the CPU shared memory specified in n2 plus the number of write points specified in n4 exceeds the CPU shared memory adress range.
  - The first device number (n3) where the data to be written is stored plus the number of write points specified in n4 exceeds the device range.
- When the host CPU operation information area, system area or host CPU refresh area is specified to the CPU shared memory address (n2) of the write destination
  (High Performance model QCPU, Process CPU) (Error code 4101)
  (Basic model QCPU, Universal model QCPU) (Error code 4111)
- The value stored in n1 is not the head I/O-number of the CPU performing the S.TO instruction.
  (High Performance model QCPU, Process CPU) (Error code 2107)
  (Basic model QCPU, Universal model QCPU) (Error code 4112)
- No CPU module is installed at the position specified by the head I/O number of the CPU module. (Error code 2110)
- The number stored in n1 is other than a correct head I/O number (3E0H, 3E1H, 3E2H or 3E3H). (Error code 4100)
- The specified instruction is improper. (Error code 4002)
- The specified number of devices is wrong. (Error code 4003)
- An unusable device was specified. (Error code 4002)

Program Example

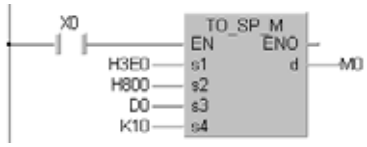
SP.TO

The data stored in CPU1 in the data registers D0 to D9 is written into the shared memory of the same CPU, beginning at adress Adresse 800H when X0 turns ON.

MELSEC Instruction List



Ladder Diagram



IEC Instruction List

9.1.2 TO, TOP, DTO, DTOP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]				●	

[1] Q00CPU/Q01CPU whose first 5 digits of the serial No. is "04122" or higher

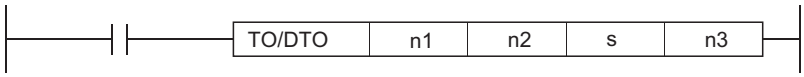
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				U
n1	●	●	●	●	●	●	●	●	●
n2	●	●	●	●	●	●	●	●	—
s	●	●	●	—	—	—	—	●	—
n3	●	●	●	●	●	●	●	●	—

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

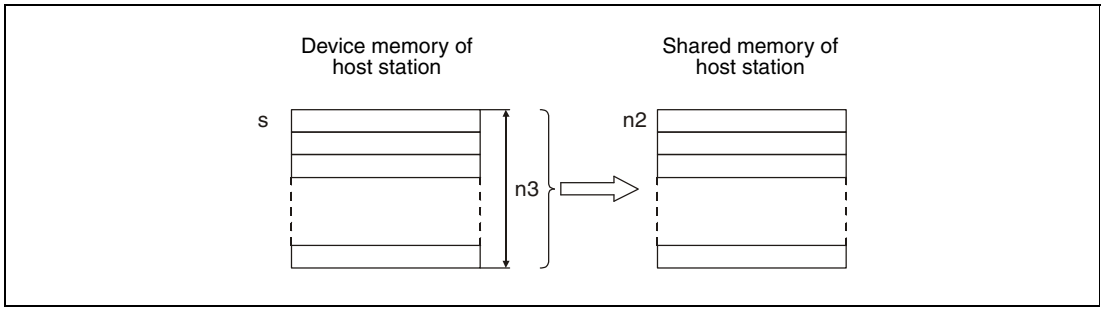
Set Data	Meaning	Data Type
n1	Head I/O number of the host CPU • Basic model QCPU: 3E0H • Universal model QCPU: 3E0H to 3E3H	BIN16-bit
n2	First number of CPU shared memory address area to be written to • Basic model QCPU: 192 to 511 • Universal model QCPU: 2048 to 4095, 10000 to 24335[1)]	
s	Data to be written or first number of device area storing data to be written.	
n3	Number of data to be written • Basic model QCPU: TO(P): 1 to 320, DTO(P) : 1 to 160 • Universal model QCPU: TO(P): 1 to 14336[1)], DTO(P) : 1 to 7168[1)]	

[1] The setting range varies depending on the auto refresh setting range of the multiple CPU high speed transmission function.

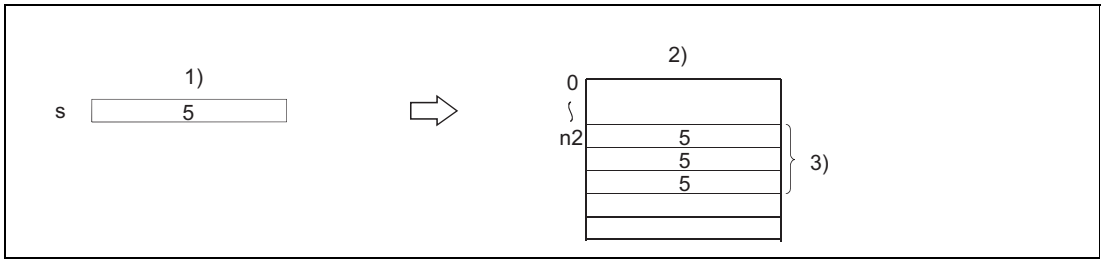
Functions **Writing data to the host station CPU shared memory**

TO/TOP **Write data**

Writes device data of words s to n3 to the CPU shared memory address specified by n2 of the host CPU module or later address.



When a constant is designated to s, the instruction writes the same data (value designated to s) to the area of n3 words starting from the specified CPU shared memory.
Following figure shows an example when the constant 5 is designated to s.



[1] Constant

[2] CPU shared memory of host CPU (n1)

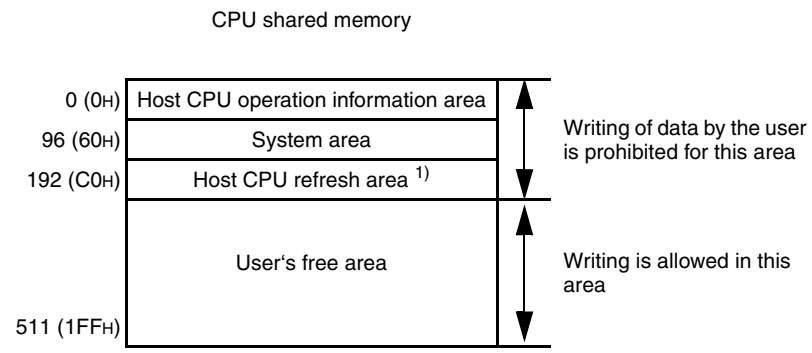
[3] n3 words (same data is written)

The head I/O number of the CPU is determined by the slot in which the CPU module is loaded. Only the first 3 digits of the head I/O number are entered in n1.

Slot of the base unit	CPU	0	1	2
Number of the CPU in multi-CPU system	1	2	3	4
Head I/O number	3E00	3E10	3E20	3E30
Contents of n1	3E0	3E1	3E2	3E3

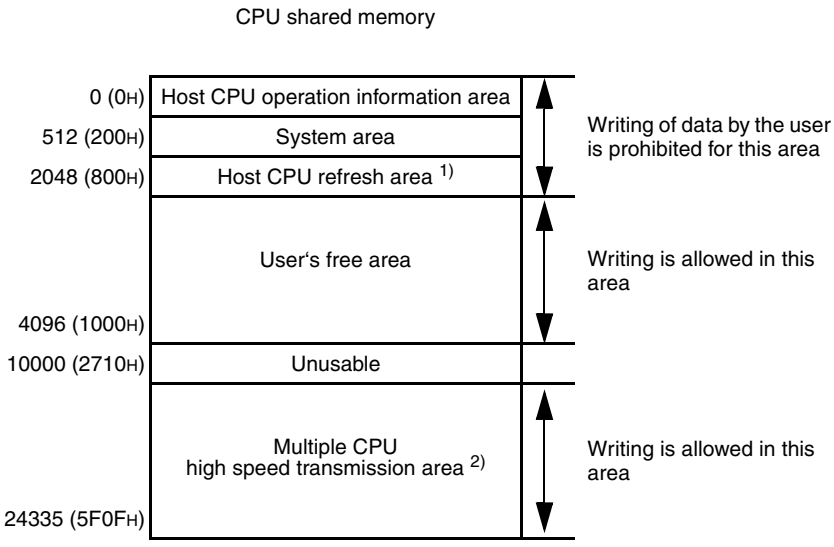
NOTE *Writing data to CPU shared memory can be performed using the intelligent function module device.*
For intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

● CPU shared memory address of the Basic model QCPU



[1] Usable as a user free area when auto refresh setting is not made. In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.

● CPU shared memory address of the Universal model QCPU



[1] Usable as a user free area when auto refresh setting is not made. In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.
[2] Data cannot be written to the multiple CPU high speed transmission area with the Q02UCPU

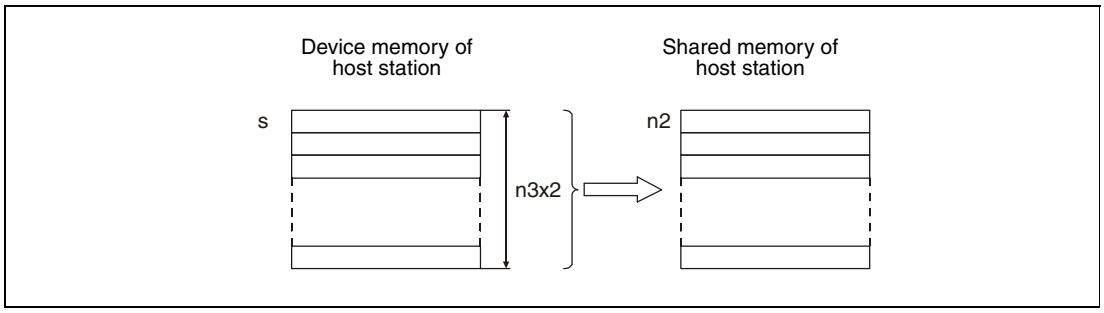
When the number of write points in n3 is „0“, processing of the instruction is not performed.

The number of data that can be written varies depending on the target CPU module.

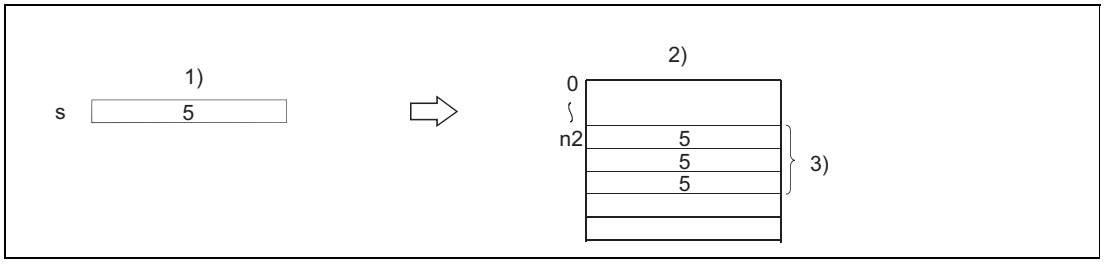
CPU Module	Number of Write Points
Basic model CPU	1–320
Universal model QCPU	1–14336

DTO/DTOP Write data

Writes device data of words s to (n3x2) to the CPU shared memory address specified by n2 of the host CPU module or later address.



When a constant is designated to s, the instruction writes the same data (value designated to s) to the area of n3x2 words starting from the specified CPU shared memory.
Following figure shows an example when the constant 5 is designated to s.



[1] Constant

[2] CPU shared memory of host CPU (n1)

[3] n3x2 words (same data is written)

When the number of write points in n3 is „0“, processing of the instruction is not performed.

The number of data that can be written varies depending on the target CPU module.

CPU Module	Number of Write Points
Basic model CPU	1–160
Universal model QCPU	1–7168

NOTE *Writing data to CPU shared memory can be performed using the intelligent function module device.*
For intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

Operation Errors

In the following cases an operation error occurs, the error flag is set, and the corresponding error code is stored in SD0:

- When the specified data is outside the following range.
  (Error code 4101)
  - The number of write points specified in n3 is outside the specified range of the setting data.
  - The beginning of the CPU shared memory specified in n2 plus the number of write points specified in n3 exceeds the CPU shared memory adress range.
  - The first device number (s) where the data to be written is stored plus the number of write points specified in n3 exceeds the device range.
  - When the head of CPU shared memory address (n2) of the write destination host CPU is outside the write permitted area.
- When the head of CPU shared memory address (n2) of the write destination host CPU is an invalid value. (Error code 4111)
- The value stored in n1 is not the head I/O-number of the host CPU. (Exclude the case when the multiple CPU high speed transmisson area of other CPU is used.)
  (Error code 4112)
- No CPU module is installed at the position specified by the head I/O number of the CPU module.
  (Error code 2110)

Program Example 1

TOP

The following program stores 10 points of data from D0 into address 10000 of the CPU shared memory of CPU No. 1 when X0 is turned ON.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

Program Example 2

DTOP

The following program stores 20 points of data from D0 into address 10000 of the CPU shared memory of CPU No. 4 when X0 is turned ON.

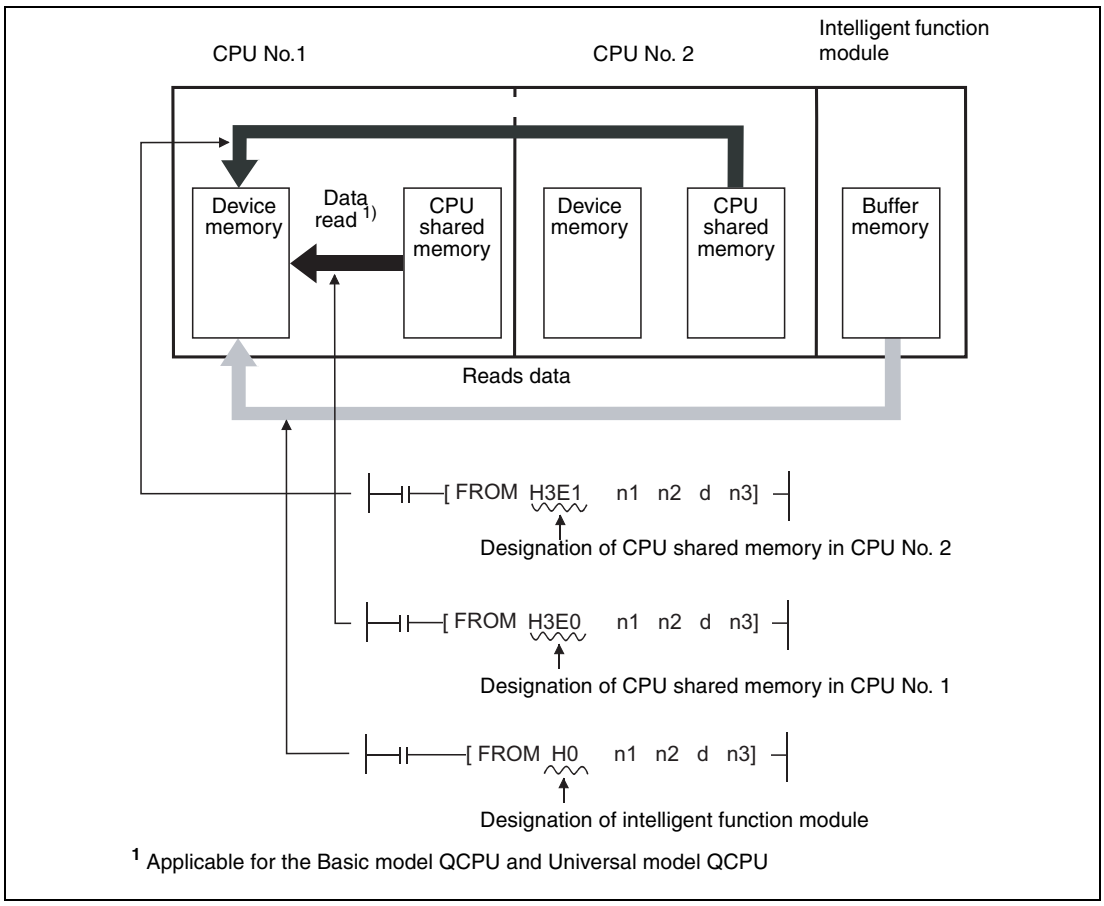
MELSEC Instruction List	Ladder Diagram	IEC Instruction List

9.2 Read from CPU shared memory of another station

The FROM/FROMP/DFRO/DFROP instructions of Multiple CPU system can read from the following memories:

- Buffer memory of intelligent function module
- CPU shared memory of other CPU module
- CPU shared memory of host CPU module (applicable for the Basic model QCPU and Universal model QCPU)

The following figure shows the processing performed when the FROM(P) instruction is executed in CPU No. 1.



NOTE *Refer to section 7.8.1 for reading the buffer memory of the intelligent function module with the FROM/DFRO instruction.*

9.2.1 FROM, FROMP, DFRO, DFROP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●[1)]	●[2)]	●		●	

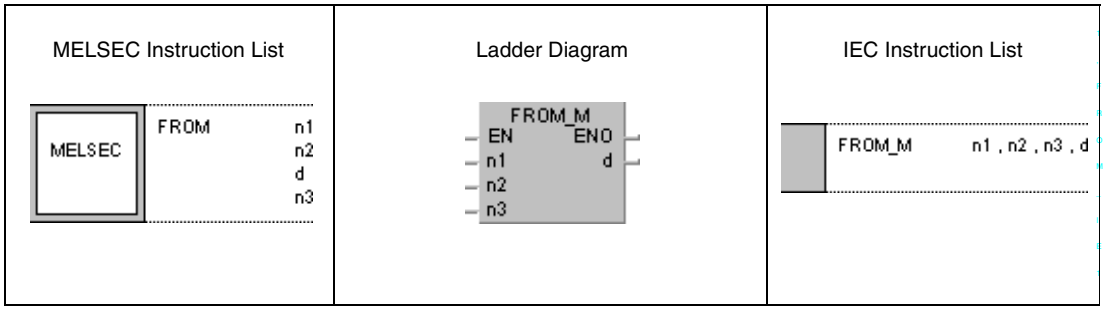
[1] Basic model QCPU:The first 5 digits of serial No is "04122" or higher.

[2] High performance model QCPU: Function version B or later.

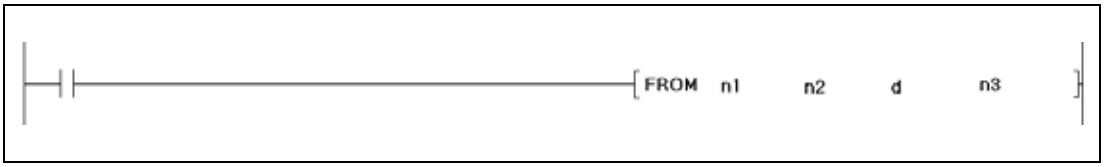
Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direkt J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word		Bit	Word				U
n1	—	●	●	●	●	●	●	●	●
n2	—	●	●	●	●	●	●	●	—
d	—	●	●	—	—	—	—	—	—
n3	—	●	●	●	●	●	●	●	—

GX IEC Developer



GX Works2



Variables

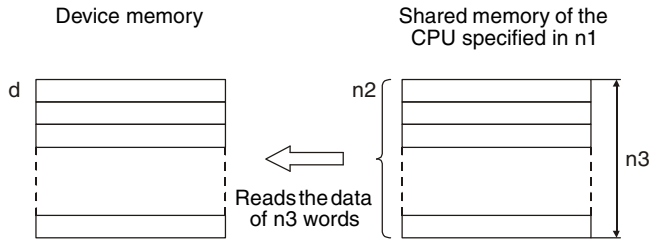
Set Data	Meaning	Data type
n1	Head I/O adress of the CPU which stores the data to be read • Basic model QCPU: 3E0H to 3E2H • High Performance model QCPU, Process CPU, Universal model QCPU: 3E0H to 3E3H	BIN 16 bit
n2	First address of data to be read in CPU shared memory • Basic model QCPU: 0 to 512 • High Performance model QCPU, Process CPU: 0 to 4095 • Universal model QCPU: 0 to 4095, 10000 to 24335[1)]	
d	First number of memory address area where the read data will be stored	
n3	Number of data words to read • Basic model QCPU: FROM(P): 1 to 512, DFRO(P): 1 to 256 • High Performance model QCPU, Process CPU: FROM(P): 1 to 4096 • Universal model QCPU: FROM(P): 1 to 14336[1)], DFRO(P) : 1 to 7168[1)]	

[1] The setting range varies depending on the auto refresh setting range of the multiple CPU high speed transmission function.

Functions **Reading from shared memory of another CPU**

FROM/FROMP **Read word data**

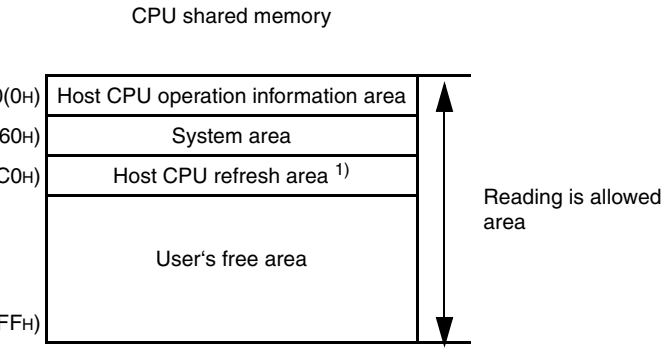
In a multi-CPU system the FROM instruction is used to read word data from the user's free area of the shared memory of another CPU. The head adress of this CPU is specified in n1. Enter the number of words to be read in n3. The starting adress in the shared memory of the other CPU is specified in n2. The data will be stored in the CPU which executes the FROM instruction starting from the device specified in d.



The head I/O number of the CPU is determined by the slot in which the CPU module is loaded. Only the first 3 digits of the head I/O number are entered in n1.

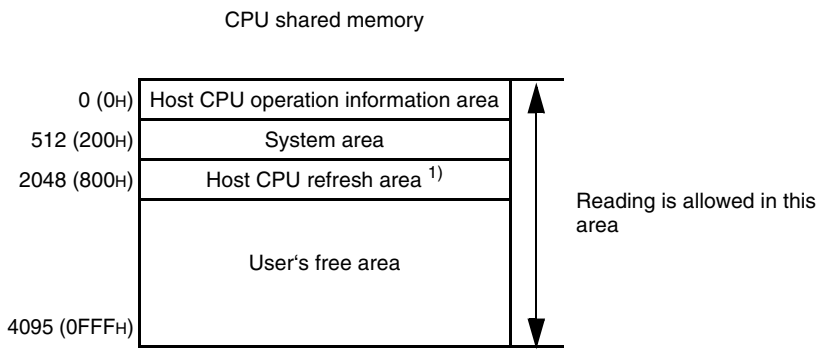
Slot of the base unit	CPU	0	1	2
Number of the CPU in multi-CPU system	1	2	3	4
Head I/O number	3E00	3E10	3E20	3E30
Contents of n1	3E0	3E1	3E2	3E3

- CPU shared memory address of the Basic model QCPU



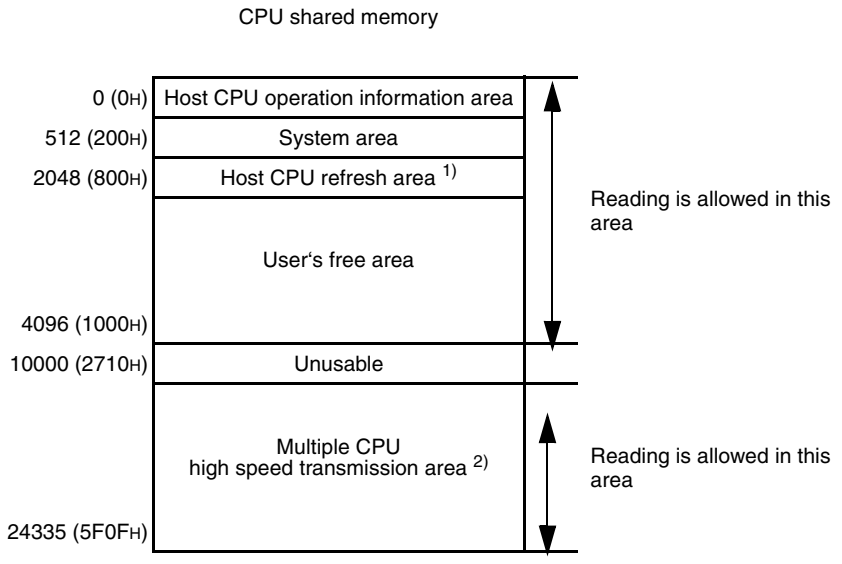
1 Usable as a user free area when auto refresh setting is not made. In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.

● CPU shared memory address of the High Performance model QCPU and Process CPU



1 Usable as a user free area when auto refresh setting is not made. In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.

● CPU shared memory address of the Universal model QCPU



1 Usable as a user free area when auto refresh setting is not made. In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.
2 Data cannot be read from the multiple CPU high speed transmission area with the Q02UCPU

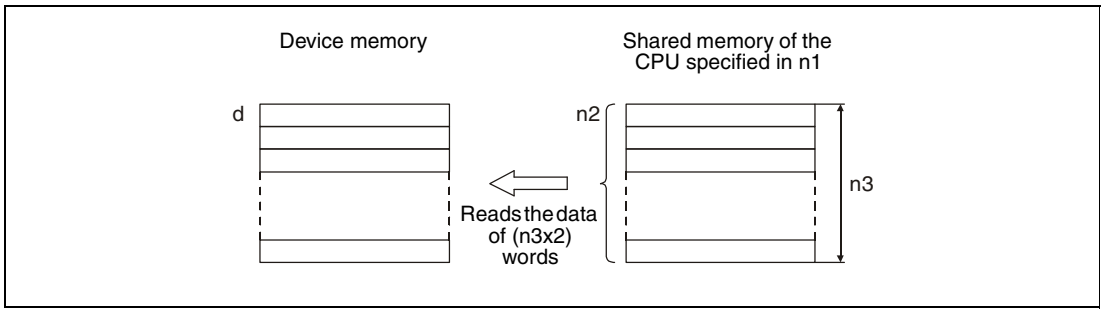
Processing of the instruction is not performed when the number of read data is entered in n3 as „0“.

The number of data that can be read varies depending on the target CPU module.

CPU Module	Number of Read Points
Basic model CPU	1–512
Universal model QCPU	1–14336

DFROM/DFROMP Read word data

Reads the data of (n3×2) words from the CPU shared memory address designated by n2 of the CPU module designated by n1, and stores that data into the area starting from the device designated by d.



Processing of the instruction is not performed when the number of read data is entered in n3 as „0“.

The number of data that can be read varies depending on the target CPU module.

CPU Module	Number of Read Points
Basic model CPU	1–256
Universal model QCPU	1–7168

NOTES *Reading data from CPU shared memory can be performed using the intelligent function module device. (For intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).)*

The QCPU provides automatic interlocks for the FROM and TO instructions.

Operation Errors

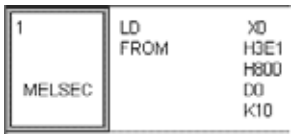
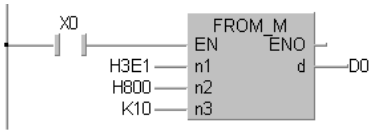
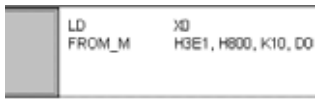
In the following cases an operation error occurs, the error flag (SM0) is set, and the corresponding error code is stored in SD0:

- When the specified data is outside the following range.
  (Error code 4101)
  - The beginning of the CPU shared memory adress (n2) from where read will be performed is greater than the CPU shared memory range.
  - The in n2 specified beginning of the CPU shared memory plus the number of read points (n3) exceeds the CPU shared memory range.
  - The read data storage device number (d) plus the number of read points (n3) is greater than the specified device range.
- No CPU module exists in the position specified with the head I/O number in n1.
  (Error code 2110)
- When the head of CPU shared memory address (n2) which performs reading is an invalid value. (Error code 4101)

Program Example 1

FROM

When XO is set, 10 datawords are read from the shared memory of CPU No. 2, starting from address 800H. The data is stored in the data registers D0 to D9 of the CPU processing the FROM instruction.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List
		

Program Example 2

DFROP

When XO is set, 20 datawords are read from the shared memory of CPU No. 4, starting from address 10000. The data is stored in the area starting from D0 of the CPU processing the instruction.

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

10 Multiple CPU Device Write/Read Instructions

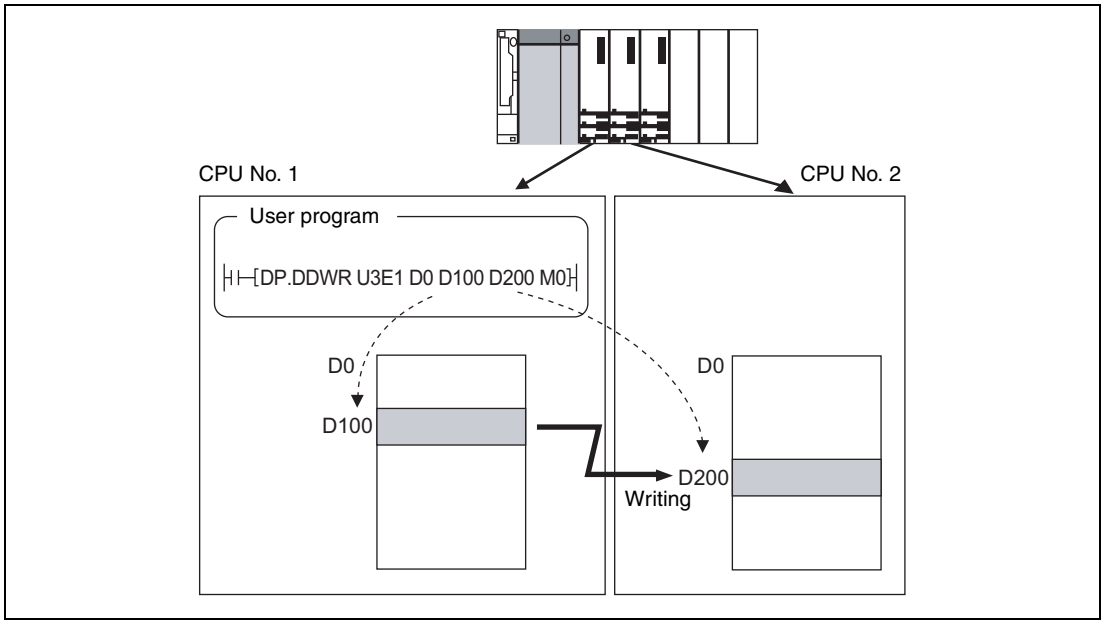
Dedicated Instructions for Multiple CPU high-speed transmission

Function	MELSEC-Instruction in MELSEC-Editor	MELSEC-Instruction in IEC-Editor
Writing devices to another CPU	D.DDWR	
	DP.DDWR	
Reading devices from another CPU	D.DDRD	
	DP.DDRD	

10.1 Overview

The multiple CPU high-speed transmission dedicated instruction directs the Universal model QCPU to write/read device data to/from another Universal model QCPU.

The following shows an operation when CPU No. 1 writes device data to CPU No. 2 with the multiple CPU high-speed transmission dedicated instruction.



NOTE *The multiple CPU high-speed transmission dedicated instruction in either host CPU or another CPU (target CPU module of instruction) is available only for the following CPU modules.*

- *Q03UDCPU, Q04UDHCPU, Q06UDHCPU*
  The first five digits of serial number is 10012 or higher.
- *Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU*
- *QnUDE(H)CPU*

Parameter setting and system configuration to execute the multiple CPU high-speed transmission dedicated instruction

The multiple CPU high-speed transmission dedicated instruction can be executed in the following parameter setting and system configuration.

- CPU No. 1 is a QnUD(H)CPU or QnUDE(H)CPU.
- The multiple CPU high speed main base unit (Q3□DB) is used.
- "Use multiple CPU high speed transmission" is selected in the Multiple CPU settings screen of PLC parameter.

Writable/readable devices

The following table shows the devices that can be written to/read from the Universal model QCPU in another CPU with the multiple CPU high-speed transmission dedicated instruction.

Category	Type	Device Name	Setting of Target Device	Remarks
Internal user device	Bit device	X, Y, M, L, B, F, SB	○	Requirements for the setting • Digits are specified by 16 bits (4 digits). • The start bit device is multiples of 16(10H).
	Word device	T, ST, C, D, W, SW	●	—
Internal system device	Bit device	SM	○	Requirements for the setting • Digits are specified by 16 bits (4 digits). • The start bit device is multiples of 16(10H).
	Word device	SD	●	—
File register	Word device	R, ZR	●	—

● Settable
○ Settable with conditions

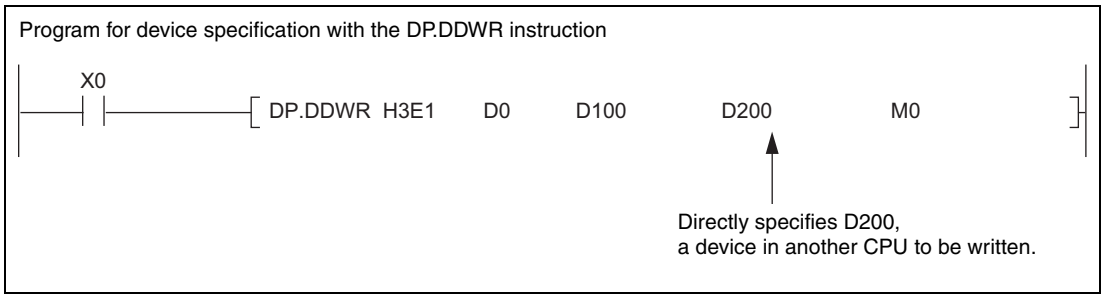
NOTE *SB, SW, SM, and SD include system information area. Take care not to destroy the system information when writing data to the devices above with the D.DDWR/DP.DDWR instruction of the multiple CPU high-speed transmission dedicated instruction.*

Specification method of a device and writable/readable device range

There are two methods for specifying a device in another CPU: device specification and string specification. They differ in writable/readable device range to another CPU.

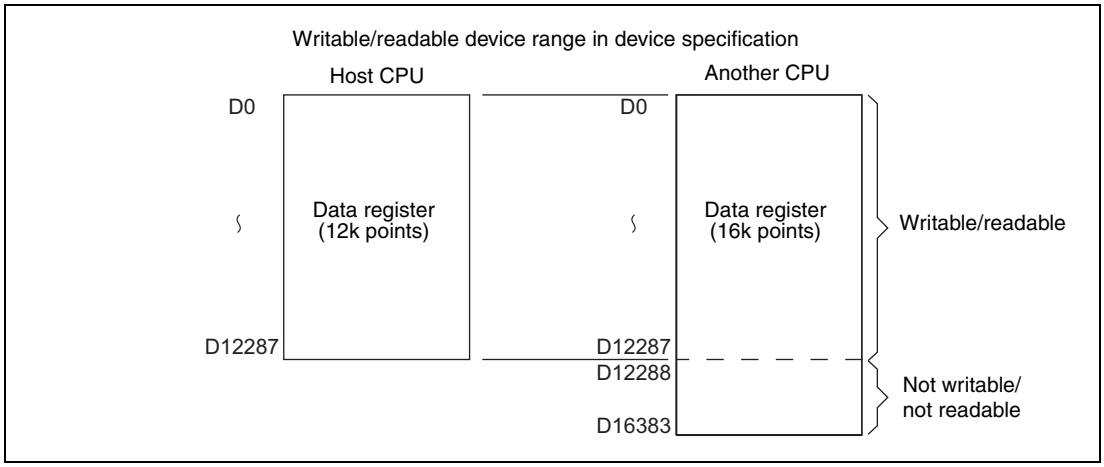
- Device specification

  The device specification is a method to directly specify a device in another CPU to be written/read.



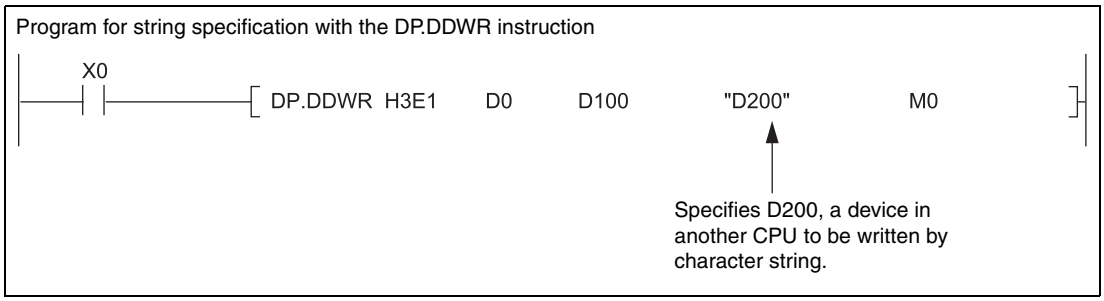
In the device specification, data can be written/read within the device range of host CPU.

For example, when data register in host CPU is 12k points and data register in another CPU is 16k points, data can be written/read by 12k points from the start of the data register in another CPU.



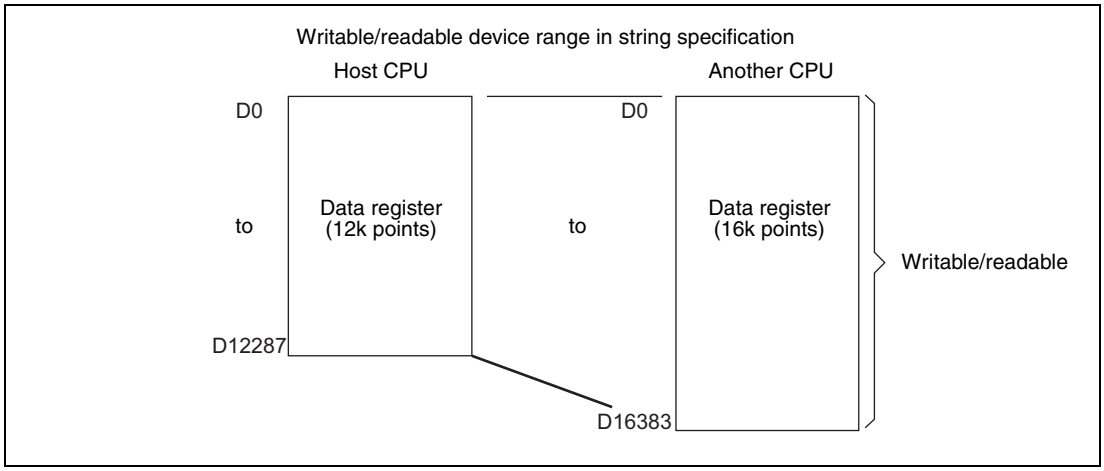
● String specification

The string specification is a method to specify a device in another CPU to be written/ read by character string.



In the string specification, data can be written to/read from all device ranges of another CPU.

For example, when data register in host CPU is 12k points and data register in another CPU is 16k points, data can be written/read by 16k points from the start of the data register in another CPU.



NOTE *The following explains precautions for string specification.*

- *The number of characters that can be specified is 32.*
- *Whether "0" is appended at the start of the device number or not, the devices are processed as the same.*
  For example, both "D1" and "D0001" are processed as "D1".
- *Whether a device is specified by upper case character or lower-case character, they are processed as the same.*
  For example, both "D1" and "d1" are processed as "D1".
- *If a device not existing in another CPU is specified by a character string, the instruction will be completed abnormally.*

Managing the multiple CPU high speed transmission area

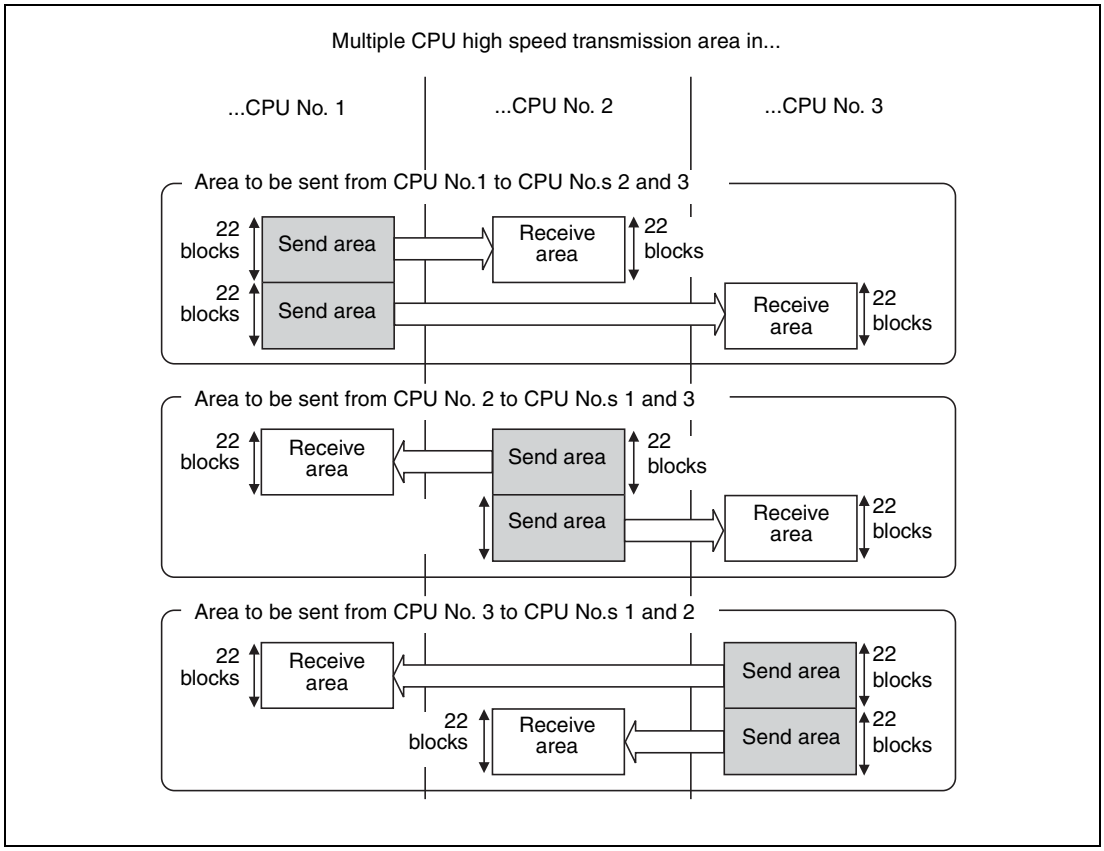
The multiple CPU high speed transmission area is managed by blocks in units of 16 words.

The following table shows the number of blocks that can be used in each CPU.

Number of CPU Modules	System Area [1)]	
	1k Points	2k Points
2	46	110
3	22	54
4	14	35

[1] For setting of the system area, refer to the QCPU User's Manual (Multiple CPU System).

The following shows configuration of the multiple CPU high speed transmission area when the multiple CPU system is configured with three CPU modules and the system area size is 1k word.



The number of blocks used for the instruction

The number of blocks used for the instruction depends on the number of write points. The following table shows the number of blocks used for the instruction.

Number of Write/Read Points Specified by the Instruction	D.DDWR/DP.DDWR Instruction	D.DDRD/DP.DDRD Instruction
1–4	1	1
5–20	2	
21–36	3	
37–52	4	
53–68	5	
69–84	6	
85–100	7	

Concurrent execution of multiple CPU high-speed transmission dedicated instructions

For the Universal model QCPU, the multiple CPU high-speed transmission dedicated instructions can be concurrently executed within the range satisfying the following formula.

Number of blocks that can be used in each CPU

>=

Total number of blocks used for the instructions concurrently executed

When the number of blocks used for the multiple CPU high-speed transmission dedicated instructions exceeds the total number of blocks in the multiple CPU high speed transmission area, the instruction will not be executed in the scan (no processing) but executed at the next scan.

Note that the instruction will be completed abnormally when the number of empty blocks in the multiple CPU high speed transmission area is less than the setting values of SD796 to SD799 (maximum number of used blocks for multiple CPU high-speed transmission dedicated instruction setting) at the execution of the instruction.

The following table shows execution possibility of the multiple CPU high-speed transmission dedicated instructions when the number of empty blocks in the multiple CPU high speed transmission area is less than the number of blocks used for the multiple CPU high-speed transmission dedicated instructions or the setting values of SD796 to SD799.

Magnitude relation between SD setting value and the number of empty blocks	Magnitude relation between the number of blocks used for the instructions (N1) [1)] and the number of empty relation blocks (N2) [2)]	
	N1 <= N2	N1 > N2
Setting values from SD796–SD799 <= Number of empty blocks [2)]	Executed	Not executed (no processing)
Setting values from SD796–SD799 > Number of empty blocks [2)]	Completed abnormally	

[1] The number of blocks used for the multiple CPU high-speed transmission dedicated instruction.

[2] The number of empty blocks in the multiple CPU high-speed transmission area.

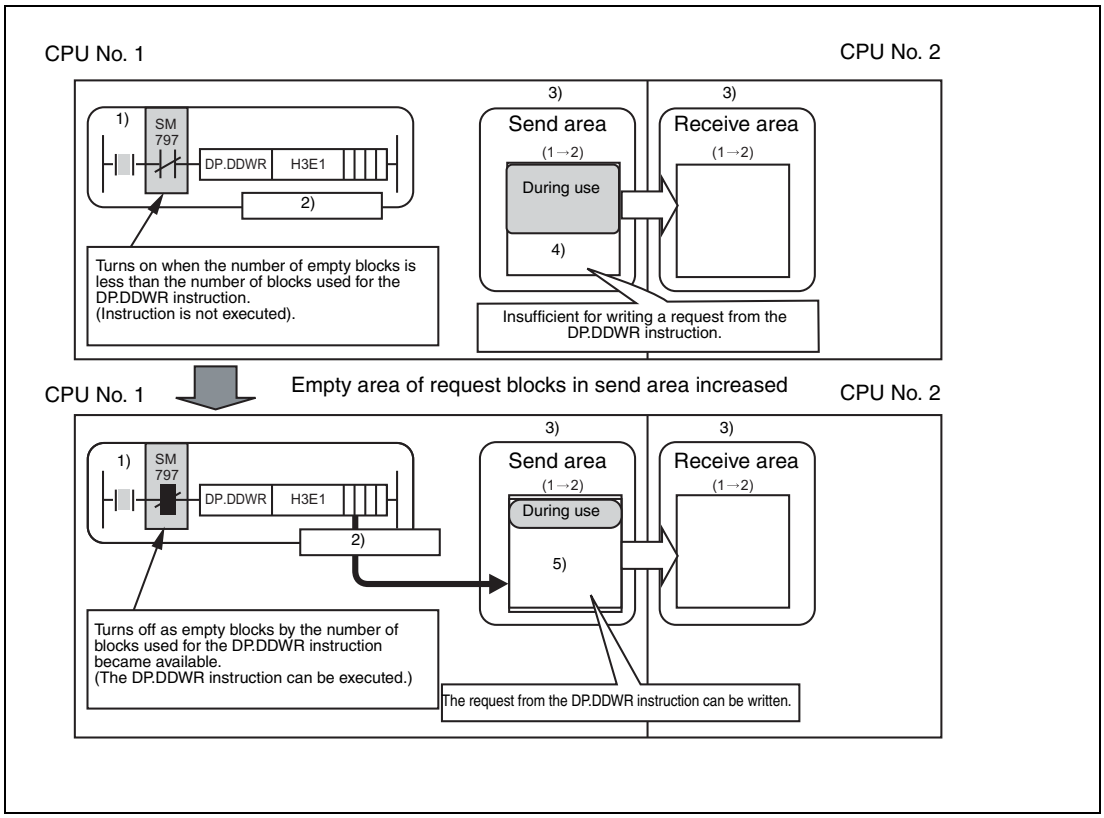
Interlock when using multiple CPU high-speed transmission dedicated instruction

Special relays SM796 to SM799 („No. of set blocks cannot be secured“) can be used as an interlock for the multiple CPU high-speed transmission dedicated instruction.

When executing the multiple CPU high-speed transmission dedicated instructions concurrently, use SM796 to SM799 as an interlock for the instructions.

NOTE *When using special relays SM796 to SM799, set the maximum number of blocks for the instruction used for each CPU to special registers SD796 to SD799. (For example, when the maximum number of blocks for the multiple CPU high-speed transmission dedicated instruction to be executed to CPU No.3 is 5, set 5 to SD798.)*

When the number of empty blocks in the multiple CPU high speed transmission area becomes equal to or less than the number of blocks set at SD796 to SD799, the corresponding special relay (SM796 to SM799) turns on.



[1] Execution command

[2] Number of request blocks: 4

[3] Multiple CPU high speed transmission area

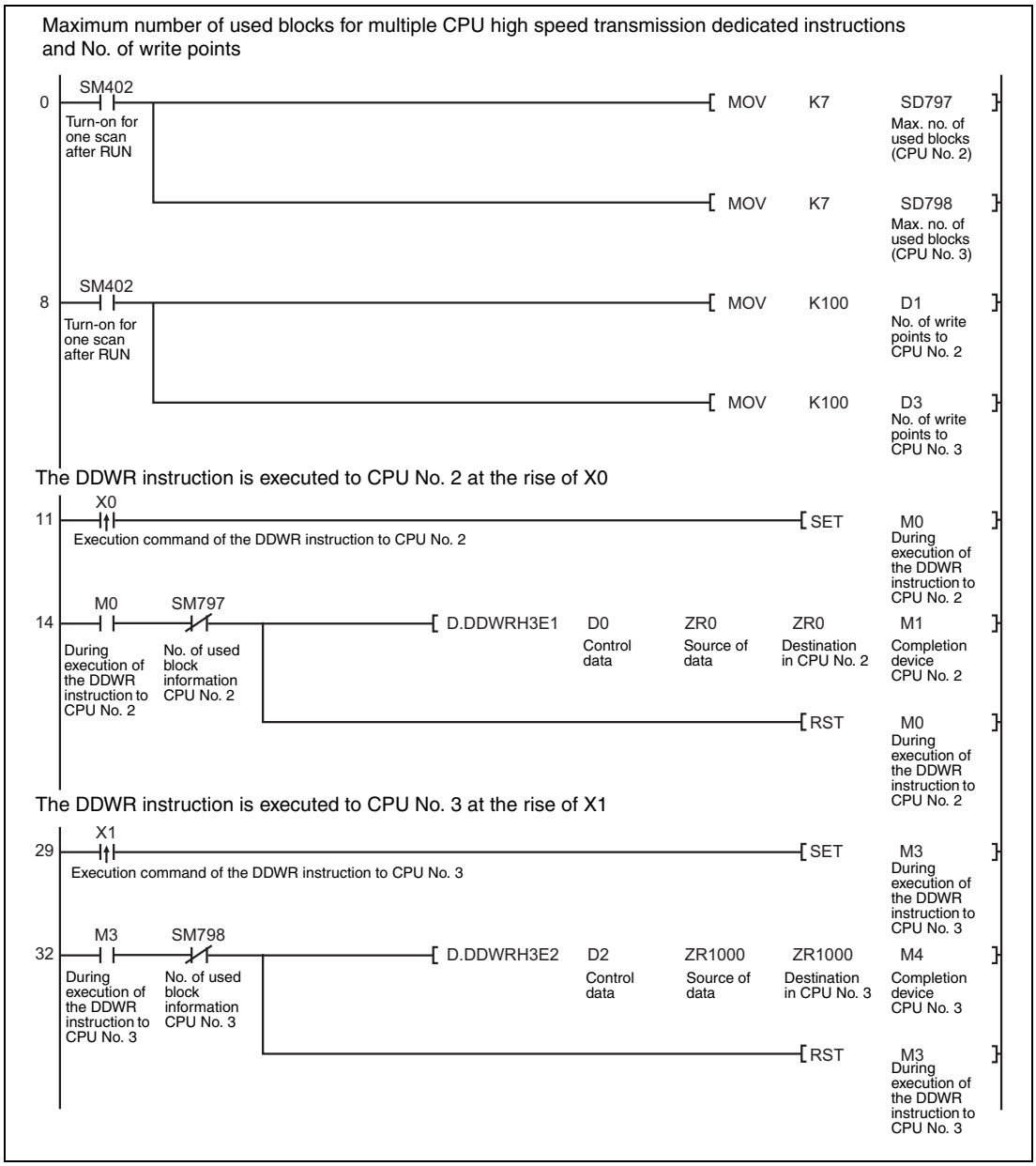
[4] Number of empty blocks: 2

[5] Number of empty blocks: 8

Program Example

Program example when SM796 to SM799 are used as an interlock

The following shows a program that executes the D.DDWR instruction to CPU No. 2 at the rise of X0, and executes the D.DDWR instruction to CPU No. 3 at the rise of X1.



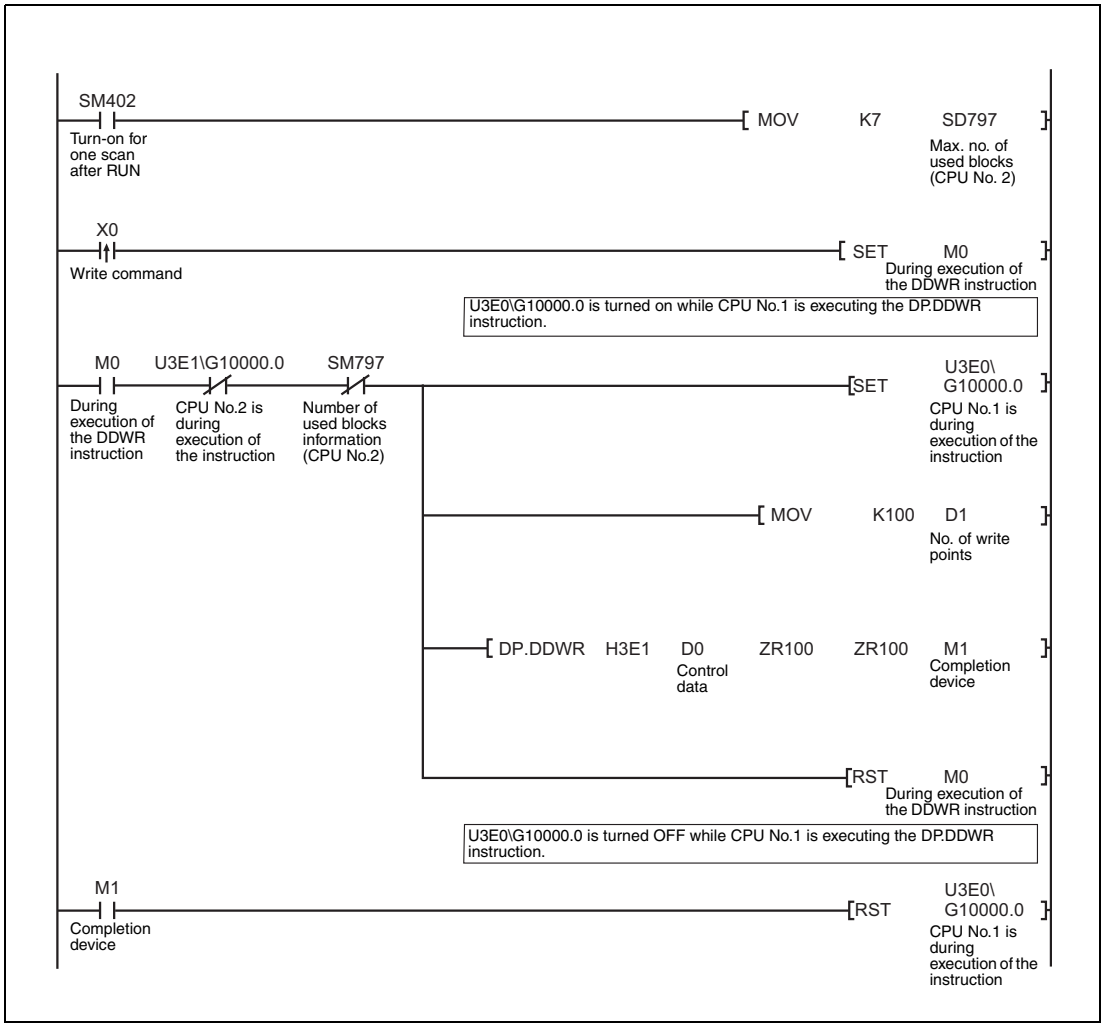
Program Example

Program example when the multiple CPU high-speed transmission dedicated instructions are executed to CPU modules by turns

When the multiple CPU high-speed transmission dedicated instructions are executed to Universal model QCPUs by turns, release an interlock to prevent the concurrent execution. Use the cyclic transmission area device (from U3E□\G10000) as an interlock.

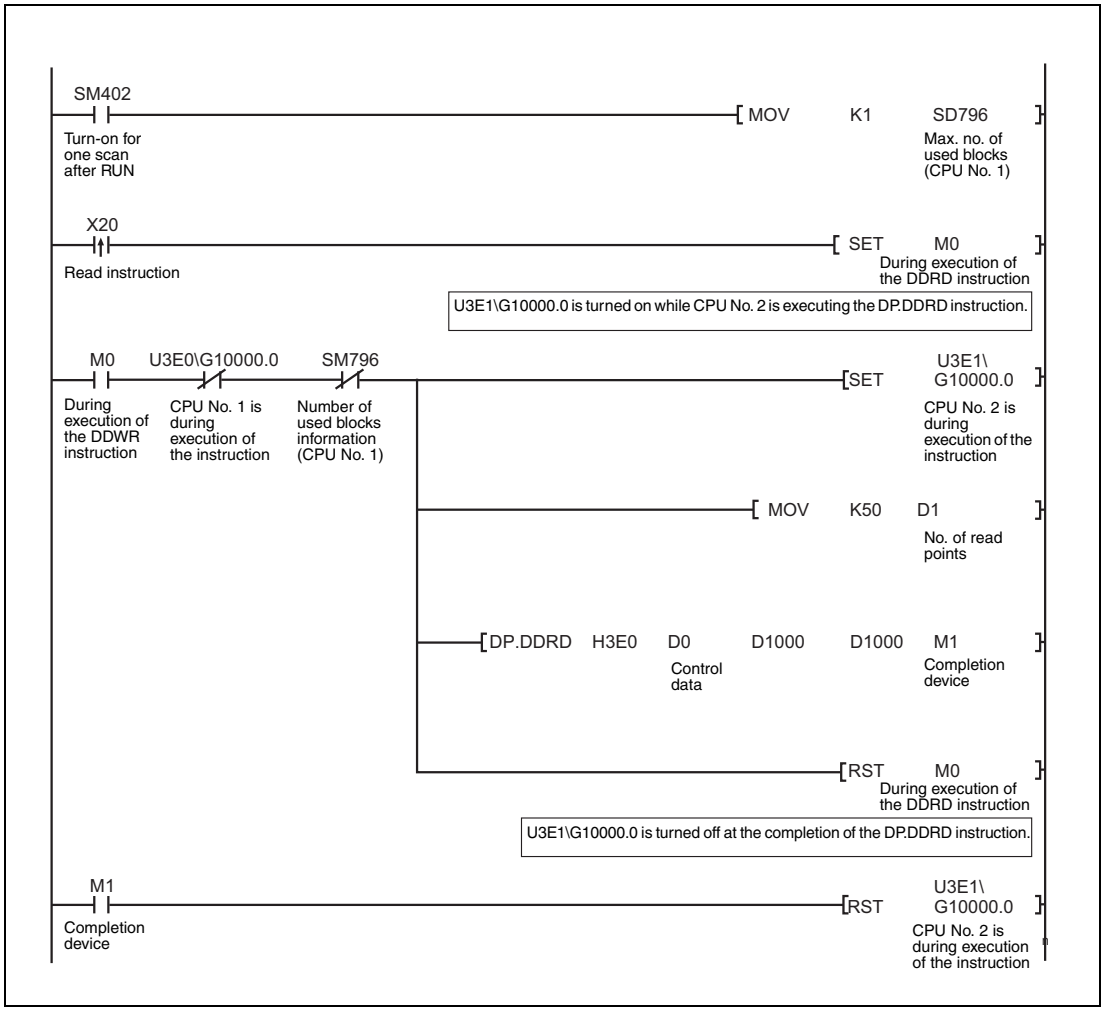
The following shows a program example when the multiple CPU high-speed transmission dedicated instructions are executed at CPU No.s 1 and 2 by turns.

Program example when the multiple CPU high-speed transmission dedicated instruction is executed at CPU No. 1:



Program Example

Program example when the multiple CPU high-speed transmission dedicated instruction is executed at CPU No. 2:



Program Example

Program example when data exceeding 100 words are written/read with the multiple CPU high-speed transmission dedicated instruction

The maximum number of write/read points that can be processed with the multiple CPU high-speed transmission dedicated instruction is 100 words. Data exceeding 100 words can be written/read by executing the multiple CPU high-speed transmission dedicated instruction at several times.

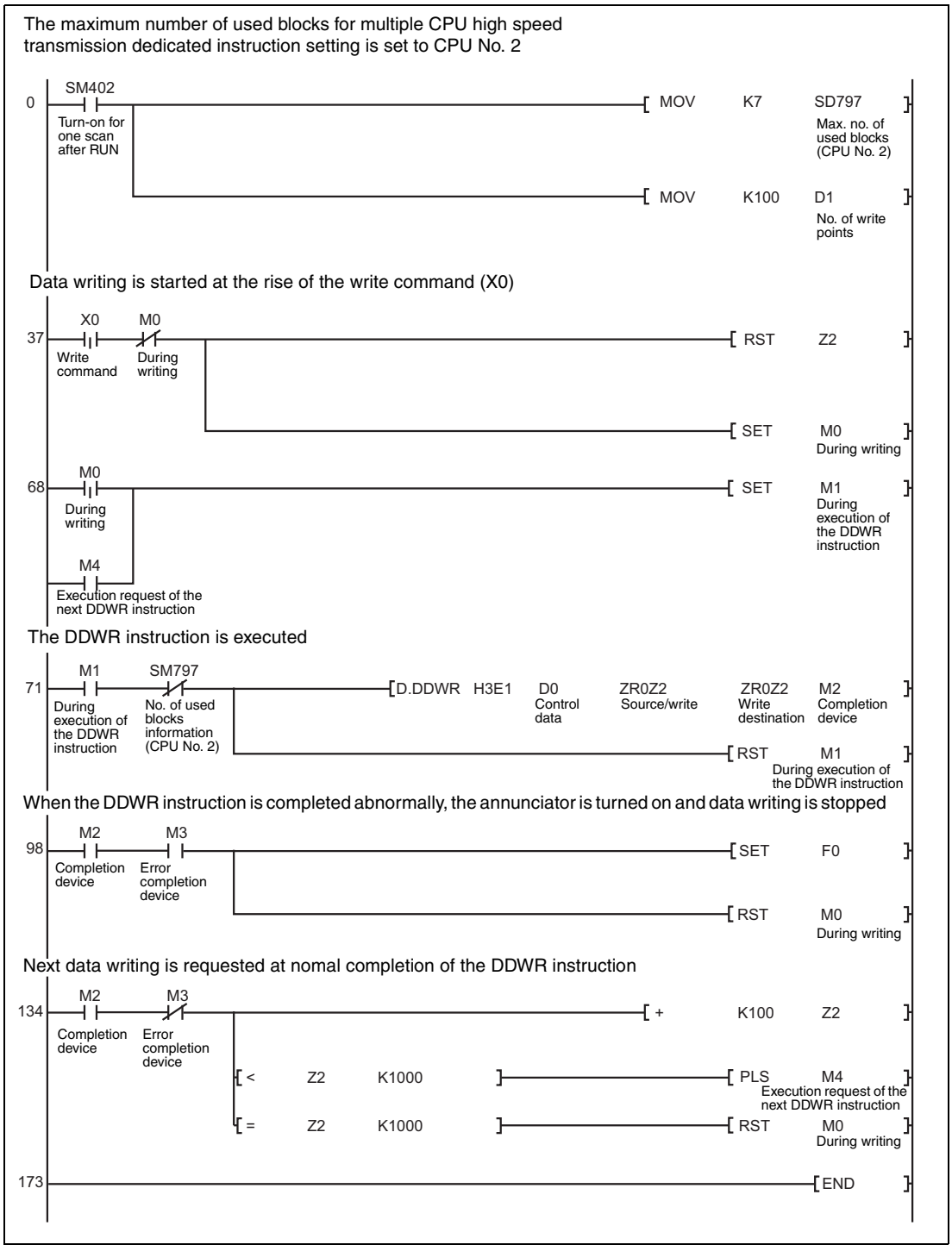
The following shows a program example using the D.DDWR/DP.DDWR instruction of the multiple CPU high-speed transmission dedicated instruction. The similar program can be used when using the D.DDRD/DP.DDRD instruction of the multiple CPU high-speed transmission dedicated instruction.

Program Example

Program example when one D.DDWR/DP.DDWR instruction is executed

The following shows a program example that writes ZR0 to ZR999 (1000 points) in CPU No. 1 to ZR0 to ZR999 in CPU No. 2 with the D.DDWR instruction.

In the following program example, the next D.DDWR instruction is executed after the completion device of the D.DDWR instruction (M2) turns on so that only one D.DDWR instruction may be executed.

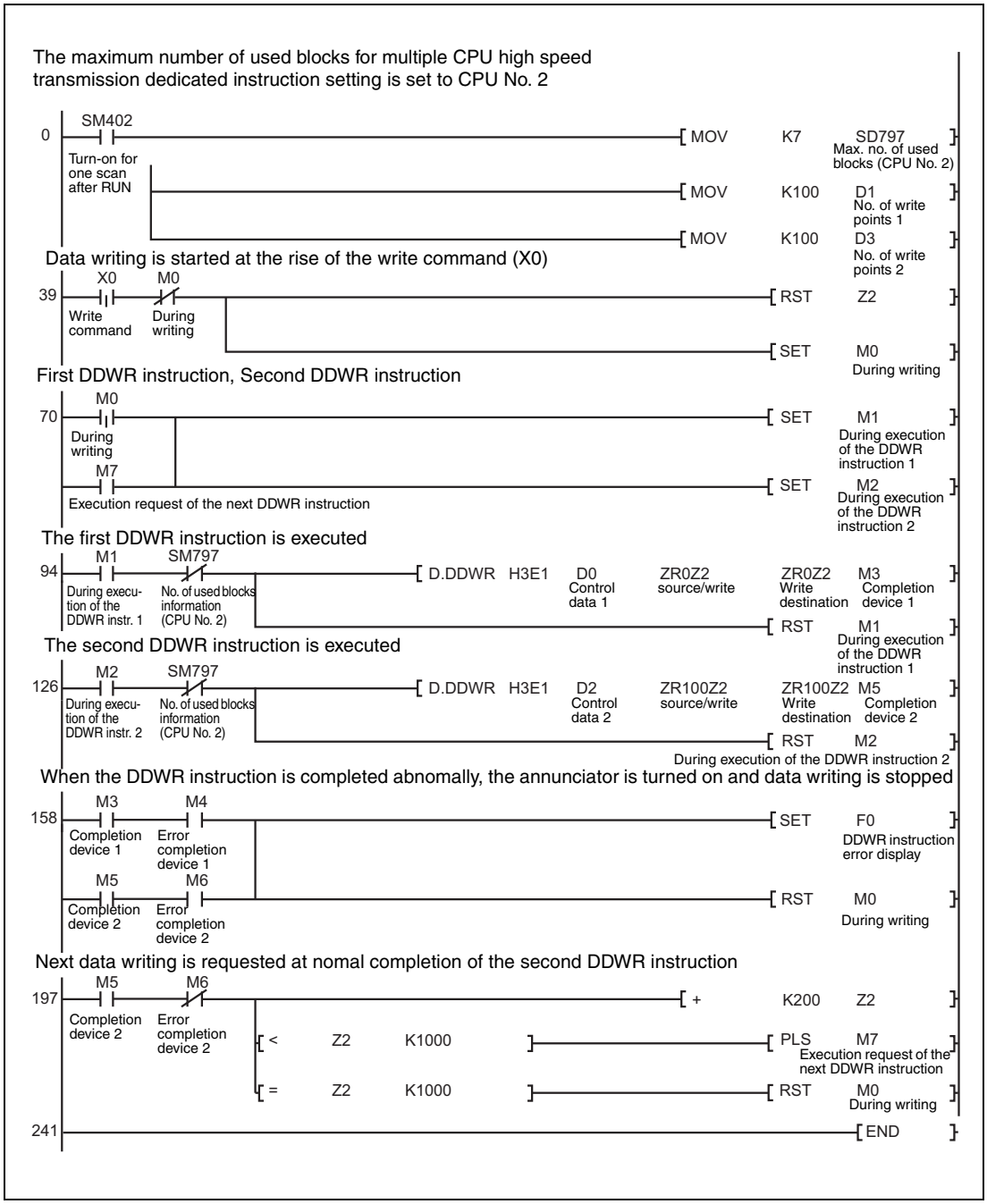


Program Example

Program example when the D.DDWR/DP.DDWR instructions are executed concurrently

The following shows a program example that writes ZR0 to ZR999 (1000 points) in CPU No. 1 to ZR0 to ZR999 in CPU No. 2 with the D.DDWR instruction. As shown on the program example, multiple CPU device write/read instructions can be executed concurrently.

When reading/writing devices with the multiple CPU high-speed transmission dedicated instructions concurrently, the more the total number of blocks in the multiple CPU high speed transmission area (send area), the more the time taken to complete reading/writing with the multiple CPU high-speed transmission dedicated instruction can be shortened.



10.2 Multiple CPU high-speed transmission instructions

10.2.1 D.DDWR, DP.DDWR

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	

[1] Q03UDCPU, Q04UDHCPU, Q06UDHCPU: first 5 digits of serial number is 10012 or higher QnUDE(H)CPU

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word [5)]		Bit	Word				
n [1)]	—	●	●	—	—	—	—	—	—
s1 [2)]	—	●[3)]	●[4)]	—	—	—	—	—	—
s2 [2)]	—	●	●	—	—	—	—	—	—
d1 [2)]	—	●	●	—	—	—	—	—	—
d2 [2)]	●[6)]	—	●[4)]	—	—	—	—	—	—

[1] Index modification cannot be made to setting data n.

[2] Index modification cannot be made to setting data from s1 to d2.

[3] Local devices cannot be used.

[4] File registers cannot be used per program.

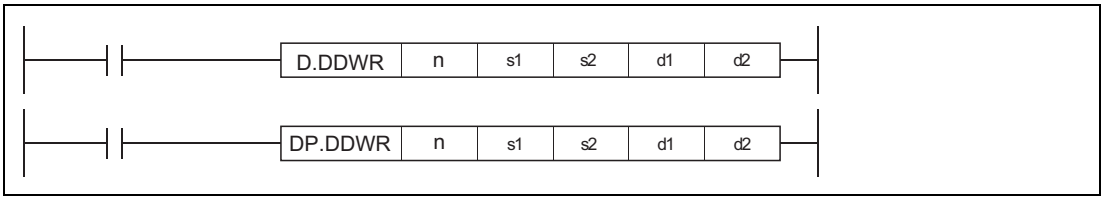
[5] FD @□ (indirect specification) cannot be used.

[6] FX and FY cannot be used.

GX IEC Developer

MELSEC Instruction	Ladder Diagram	IEC Instruction List

GX Works2



Variables

<table>
<tr><th>Set data</th><th colspan="4">Meaning</th><th>Setting Range</th><th>Set By</th><th>Data Type</th></tr>
<tr><td>n</td><td colspan="4">The result of dividing the start I/O number of another CPU by 16
CPU No. 1: 3E0H,
CPU No. 2: 3E1H,
CPU No. 3: 3E2H,
CPU No. 4: 3E3H</td><td>—</td><td>—</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="4">s1</td><td colspan="5">Start device of the host CPU that stores control data</td><td rowspan="4">Device name</td></tr>
<tr><td>Set data</td><td>Item</td><td>Meaning/Set Data</td><td>Setting Range</td><td>Set By</td></tr>
<tr><td>(s1)+0</td><td>Completion status</td><td>An execution result upon completion of the instruction is stored.
• 0000H:
No errors (normal completion)
• Other than 0000H:
Error code (error completion)</td><td>—</td><td>System</td></tr>
<tr><td>(s1)+1</td><td>Number of write points</td><td>Set the number of write points in units of words.</td><td>1–100</td><td>User</td></tr>
<tr><td>s2</td><td colspan="5">Start device of the host CPU that stores data to be written</td><td>Device name</td></tr>
<tr><td>d1</td><td colspan="5">Start device of another CPU that stores write data</td><td>Device [1)]
Character string [2,3)]</td></tr>
<tr><td>d2</td><td colspan="5">Completion device</td><td>Bit</td></tr>
</table>

[1] By specifying a file register (R, ZR), data can be written to devices in another CPU, outside the range of host CPU.

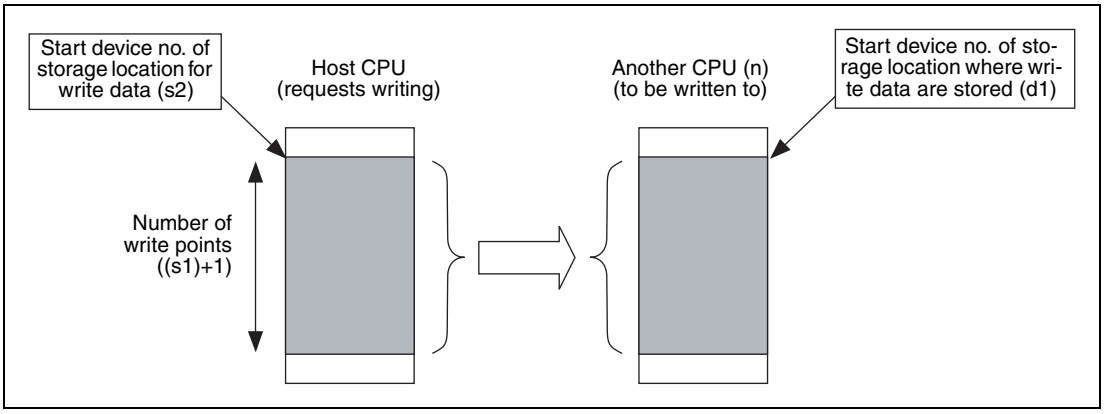
[2] By specifying the start device by " ", devices can be written to devices in another CPU, outside the range of host CPU.

[3] Indexed devices cannot be specified (e.g. D0Z0).

Functions **Writing devices to another CPU**

D.DDWR/DP.DDWR Write instruction

In a multiple CPU system, data stored in a device specified by host CPU (s2) or later is stored by the number of write points specified by ((s1)+1) into a device specified by another CPU (n) (d1) or later.



Whether to complete the D.DDWR/DP.DDWR instruction normally can be checked by the completion device ((d2)+0) and completion status display device ((d2)+1).

- Completion device ((d2)+0)
  Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing.
- Completion status display device ((d2)+1)
  This device turns on/off depending on the status upon completion of the instruction.

  Normal completion: Off

  Error completion: Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing
  (At error completion, an error code is stored at control data ((s1)+0): Completion status).

The number of blocks used for the instruction depends on the number of write points (refer to section 10.1). The following table shows the number of blocks used for the instruction:

Number of write points specified by the instruction	Number of blocks used by the D.DDWR/DP.DDWR instruction
1–4	1
5–20	2
21–36	3
37–52	4
53–68	5
69–84	6
85–100	7

The instruction will be completed abnormally when there are no empty blocks in the multiple CPU high speed transmission area.

Set the number of blocks used for the instruction at special registers (SD796 to SD799), and use the special relays (SM796 to SM799)as an interlock prevent error completion (refer to section 10.1).

NOTES

Digit specification of bit device is possible for n, s2, and d1. Note that when the digit specification of bit device is made to s2 or d1, the following conditions must be met.

- *Digits are specified by 16 bits (4 digits).*
- *The start bit device is multiples of 16 (10H).*

Execute this instruction after checking that the write target CPU is powered on. Not doing so may end up no processing.

If changing a range of the device specified at setting data between after execution of the instruction and turn-on of the completion device, data to be stored by system (completion status, completion device) cannot be stored normally.

SB, SW, SM, and SD include system information area. Take care not to destroy the system information when writing data to the devices above with the D.DDWR/DP.DDWR instruction of the multiple CPU high-speed transmission dedicated instruction.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- Specified another CPU is wrong or the multiple CPU high-speed transmission dedicated instruction cannot be used in the setting.
  (Error code 4350)
  - A reserved CPU has been specified.
  - Unmounted CPU has been specified.
  - The value in n (the start I/O number of the other CPU divided by 16) is out of the range of 3E0H to 3E3H.
  - The instruction was executed without setting "Use multiple CPU high speed transmission".
  - The instruction was executed with the Q02UCPU.
  - Host CPU has been specified.
  - The CPU where the instruction cannot be executed has been specified.
- The instruction cannot be executed with the CPU.
  (Error code 4351)

  The other CPU does not support this instruction.
- The number of devices is wrong. (Error code 4352)
- A device that cannot be used for the instruction has been specified. (Error code 4353)
- A device has been specified by the character string that cannot be used. (Error code 4354)
- The number of write points ((s1)+1)) is other than 1 to 100. (Error code 4355)

In any of the following cases, the instruction is completed abnormally, and an error code is stored into a device specified at completion status storage device ((s1)+0)).

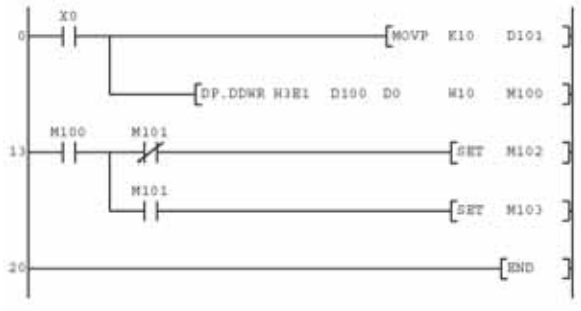
- The request of the instruction to the target CPU is more than the acceptable value (no empty blocks exist in the multiple CPU high speed transmission area).
  (Error code 0010H)
- A device for another CPU specified at s2 cannot be used at another CPU, or is out of device range.
  (Error code 1001H)
- The number of write points set with the D.DDWR/DP.DDWR instruction is 0.
  (Error code 1080H)
- The response of the instruction from another CPU cannot be returned (no empty blocks exist in the multiple CPU high speed transmission area). (Error code 1003H)

Program Example

DP.DDWR

This program stores data by 10 words starting from D0 in host CPU into W10 or later in CPU No. 2 when X0 turns on.

Ladder diagram



10.2.2 D.DDRD, DP.DDRD

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
				●[1)]	

[1] Q03UDCPU, Q04UDHCPU, Q06UDHCPU: first 5 digits of serial number is 10012 or higher QnUDE(H)CPU

Devices

	Usable Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H (16#)	Other
	Bit	Word [5)]		Bit	Word				
n [1)]	—	●	●	—	—	—	—	—	—
s1 [2)]	—	●[3)]	●[4)]	—	—	—	—	—	—
s2 [2)]	—	●	●	—	—	—	—	—	—
d1 [2)]	—	●	●	—	—	—	—	—	—
d2 [2)]	●[6)]	—	●[4)]	—	—	—	—	—	—

[1] Index modification cannot be made to setting data n.

[2] Index modification cannot be made to setting data from s1 to d2.

[3] Local devices cannot be used.

[4] File registers cannot be used per program.

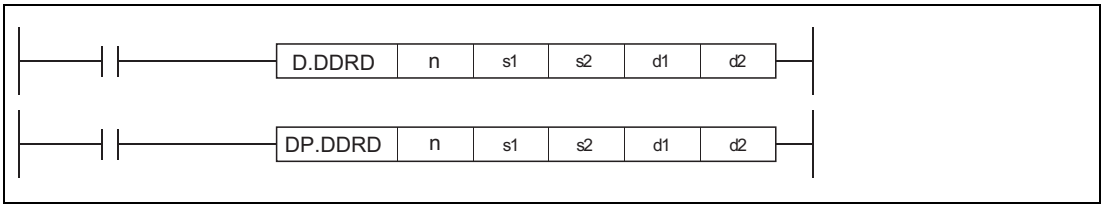
[5] FD @□ (indirect specification) cannot be used.

[6] FX and FY cannot be used.

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set data	Meaning			Setting Range	Set By	Data Type
n	The result of dividing the start I/O number of another CPU by 16 CPU No. 1: 3E0H, CPU No. 2: 3E1H, CPU No. 3: 3E2H, CPU No. 4: 3E3H			—	—	BIN 16-bit
s1	Start device of the host CPU that stores control data					Device name
	Set data	**Item**	**Meaning/Set Data**	**Setting Range**	**Set By**	
	(s1)+0	Completion status	An execution result upon completion of the instruction is stored. • 0000H: No errors (normal completion) • Other than 0000H: Error code (error completion)	—	System	
	(s1)+1	Number of read points	Set the number of read points in units of words.	1–100	User	
s2	Start device of the another CPU that stores data to be read					Device name
d1	Start device of host CPU that stores read data					Device [1)] Character string [2,3)]
d2	Completion device					Bit

[1] By specifying a file register (R, ZR), data can be read to devices in another CPU, outside the range of host CPU.

[2] By specifying the start device by " ", devices can be read to devices in another CPU, outside the range of host CPU.

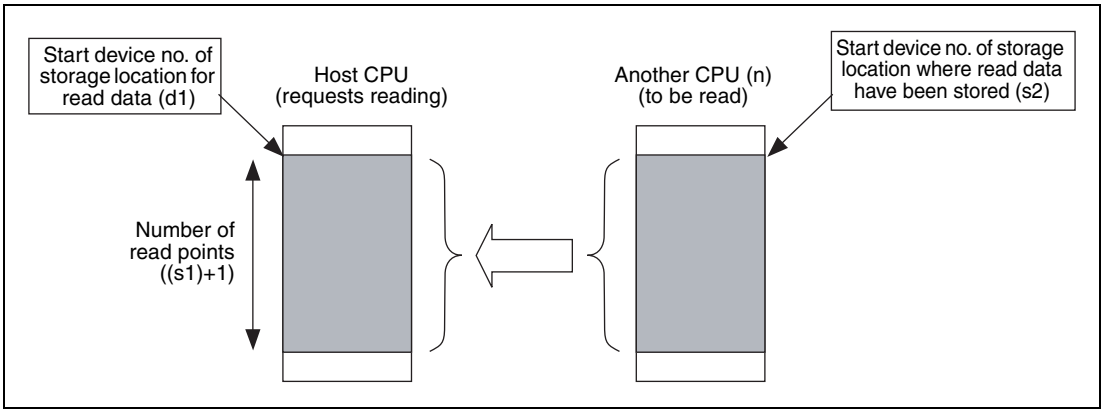
[3] Indexed devices cannot be specified (e.g. D0Z0).

Functions

Reading devices from another CPU

D.DDRD/DP.DDRDR Read instruction

In multiple CPU system, data stored in a device specified by another CPU (n) (s2) or later is stored by the number of read points specified by ((s1)+1) into a device specified by host CPU (d1) or later.



Whether to complete the D.DDRD/DP.DDRD instruction normally can be checked by the completion device ((d2)+0) and completion status display device ((d2)+1).

- END processing in scan data that CPU completed the instruction turns on the device ((d2)+0) and the next END processing turns off the device.
- This device ((d2)+1) turns on/off depending on the status upon completion of the instruction.

  Normal completion: Off

  Error completion: Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing. (At error completion, an error code is stored at control data ((s1)+0): Completion status).

The number of blocks used for the instruction is independent of the number of read points (refer to section 10.1). The following table shows the number of blocks used for the instruction:

Number of read points specified by the instruction	Number of blocks used by the D.DDRD/DP.DDRD instruction
1–100	1

The instruction will be completed abnormally when there are no empty blocks in the multiple CPU high speed transmission area.

Set the number of blocks used for the instruction at special registers (SD796 to SD799), and use the special relays (SM796 to SM799) as an interlock prevent error completion (refer to section 10.1).

NOTES

Digit specification of bit device is possible for n, s2, and d1. Note that when the digit specification of bit device is made to s2 or d1, the following conditions must be met.

- *Digits are specified by 16 bits (4 digits).*
- *The start bit device is multiples of 16 (10H).*

Execute this instruction after checking that the read target CPU is powered on. Not doing so may end up no processing.

If changing a range of the device specified at setting data between after execution of the instruction and turn-on of the completion device, data to be stored by system (completion status, completion device) cannot be stored normally.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- Specified another CPU is wrong or the multiple CPU high-speed transmission dedicated instruction cannot be used in the setting.
  (Error code 4350)
  - A reserved CPU has been specified.
  - Unmounted CPU has been specified.
  - The value in n (the start I/O number of the other CPU divided by 16) is out of the range of 3E0H to 3E3H.
  - The instruction was executed without setting "Use multiple CPU high speed transmission".
  - The instruction was executed with the Q02UCPU.
  - Host CPU has been specified.
  - A CPU where the instruction cannot be executed has been specified.
- The instruction cannot be executed with the CPU.
  (Error code 4351)

  The other CPU does not support this instruction.
- The number of devices is wrong. (Error code 4352)
- A device that cannot be used for the instruction has been specified. (Error code 4353)
- A device has been specified by the character string that cannot be used. (Error code 4354)
- The number of read points ((s1)+1)) is other than 1 to 100. (Error code 4355)

In any of the following cases, the instruction is completed abnormally, and an error code is stored into a device specified at completion status storage device ((s1)+0).

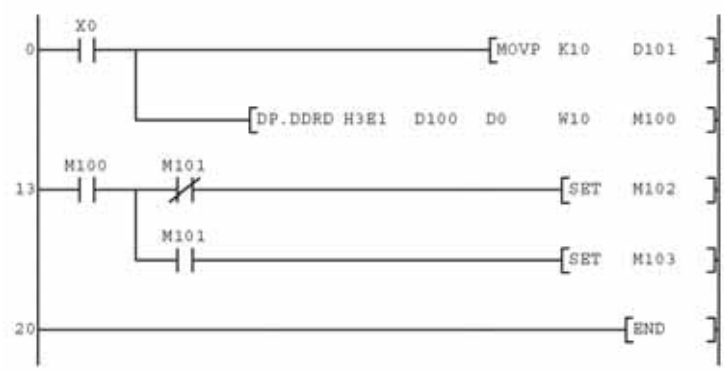
- The request of the instruction to the target CPU is more than the acceptable value (no empty blocks exist in the multiple CPU high speed transmission area).
  (Error code 0010H)
- A device for another CPU specified at s2 cannot be used at another CPU, or is out of device range.
  (Error code 1001H)
- The number of read points set with the D.DDRD/DP.DDRD instruction is 0.
  (Error code 1081H)
- The response of the instruction from another CPU cannot be returned (no empty blocks exist in the multiple CPU high speed transmission area).
  (Error code 1003H)

Program Example

DP.DDRD

This program stores data by 10 words starting from D0 in host CPU No. 2 into W10 or later in host CPU when X0 turns on.

Ladder diagram



MELSEC Instruction List

```
 0  LD       X0
 1  MOVP     K10    D101
 3  DP.DDRD  H3E1   D100   D0    W10   M100
13  LD       M100
14  MPS
15  ANI      M101
16  SET      M102
17  MPP
18  AND      M101
19  SET      M103
20  END
```

11 Instructions for MELSEC System Q

11.1 Instruction for a redundant system

Switches between the control system and standby system at the END processing of the scan executed with the SP.CONTSW instruction.

Function	MELSEC-Instruction in MELSEC-Editor	MELSEC-Instruction in IEC-Editor
System switching	SP.CONTSW	CONTSW_SP_M

NOTE *For more information of a redundant system refer to the User‘s manuals of the redundant CPU modules Q12PRHCPU and Q25PRHCPU.*

11.1.1 SP.CONTSW

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
			●		

Devices

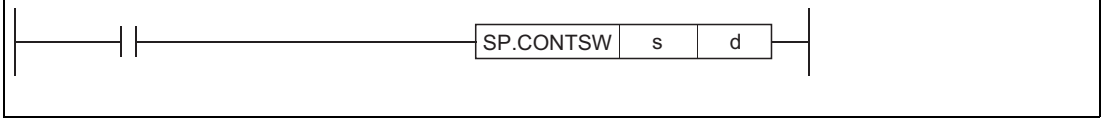
	Devices								
	Internal Devices (System, User)		File Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constant K, H	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	●	—
d	●	●[1)]	●[1)]	—	—	—	—	—	—

[1] The bit specification for the word device is available.

GX IEC Developer

MELSEC Instruction List	Ladder Diagram	IEC Instruction List

GX Works2



Variables

Set Data	Meaning	Data Type
s	Value other than 0 and used to identify the processing that issued the system switching request	BIN 16-bit
d	Error completion device number	Bit

Functions **System switching instruction**

SP.CONTSW **System switching instruction**

Switches between the control system and standby system at the END processing of the scan executed with the SP.CONTSW instruction.

When using the SP.CONTSW instruction for system switching, the "manual switching enable flag (SM1592)" must have been turned ON (enabled) in advance.

s is provided to identify the processing block of the program where system switching occurred when multiple SP.CONTSW instructions are used. At s, specify a value within the ranges -32768 to -1 and 1 to 32767 (1H to FFFFH).

The s value specified for the SP.CONTSW instruction is stored into the "system switching instruction argument (SD6)" of the error common information when the system switching is normally completed. The s value specified for the SP.CONTSW instruction can be confirmed in the error common information of the PLC diagnostics dialog box on the programming tool. When multiple SP.CONTSW instructions are executed during the same scan, the argument of the SP.CONTSW instruction executed first is stored into the system switching instruction argument (SD6).

When system switching is normally completed, the value specified for the SP.CONTSW instruction is stored into the "system switching instruction argument (SD1602)" of the new control system CPU module. The new control system CPU module means the CPU module that was switched from the standby system to the control system by the SP.CONTSW instruction. By reading the SD1602 value from the new control system CPU module, which the SP.CONTSW instruction was used for system switching can be confirmed.

The error completion device is turned ON by the control system CPU module when system switching by the SP.CONTSW instruction was unsuccessful.

- When OPERATION ERROR is detected due to any of the following reasons at the execution of the SP.CONTSW instruction, the error completion device is turned ON during the instruction execution.
  - 0 is specified at s of the executed SP.CONTSW instruction.
  - The "manual switching enable flag (SM1592)" is OFF.
  - The SP.CONTSW instruction was executed by the standby system in the separate mode.
  - The SP.CONTSW instruction was executed in the debug mode.
- If systems could not be switched due to any of the reasons given in the following table, the error completion device turns ON when system switching is executed in the END processing. When the error completion device was turned ON due to unsuccessful system switching, 16 is stored into the "Reason(s) for system switching (SD1588)" and the reason No. of the following table is stored into the "Reason(s) for system switching failure (SD1589)".

Reason No.	Reasons for System Switching Failure
0	Normally completed
1	Tracking cable is disconnected or faulty.
2	Hardware fault, power-off, reset or watchdog timer error occurred in the standby system.
3	Watchdog timer error occurred in the control system.
4	Preparations being made for tracking transfer.
5	Communication time-out.
6	Stop error occurred in the standby system. (Excluding watchdog timer error)
7	Operating status different between the control system and standby system.
8	Memory copy being executed from the control system to the standby system.
9	Write during RUN being executed.
10	Network fault detected by the standby system.

Use a user program or a programming tool to turn OFF the error completion bit that has turned ON.

If normal system switching is performed by the execution of the SP.CONTSW instruction with the error completion device ON, the error completion device of the new standby system CPU module is also turned OFF. When system switching is performed due to a factor other than the SP.CONTSW instruction, however, the error completion device is not turned OFF.

Operation Errors

In the following cases an operation error occurs, the error flag (SM0) is turned ON and an error code is stored into SD0:

- The value specified at s is 0 at execution of the SP.CONTSW instruction.
  (Error code: 4100)
- The manual switching enable flag (SM1592) is OFF (disable) at execution of the SP.CONTSW instruction.
  (Error code: 4120)
- The SP.CONTSW instruction was executed by the standby system CPU module in the separate mode.
  (Error code: 4121)
- The SP.CONTSW instruction was executed in the debug mode.
  (Error code: 4121)

If system switching was unsuccessful, the error flag (SM0) is turned ON and the error code 6220 is stored into SD0.

- The tracking cable is disconnected or faulty.
- Hardware fault, power-off, reset or watchdog timer error occurred in the standby system.
- Watchdog timer error occurred in the control system.
- Preparations are being made for tracking transfer.
- Communication time-out occurred.
- Stop error, excluding watchdog timer error, occurred in the standby system.
- The operating status differs between the control system and standby system.
- Memory copy is being executed from the control system to the standby system.
- Write during RUN is being executed.
- Network fault was detected by the standby system.

Program Example

SP.CONTSW

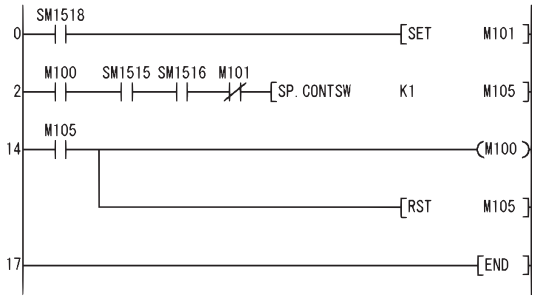
The following program executes system switching on the leading edge of the system switching command (M100).

If the system switching command (M100) remains ON, the SP.CONTSW instruction is also executed by the new control system CPU module after system switching. Therefore, M101 is added to the execution conditions as a consecutive switching prevention flag.

MELSEC Instruction List

IEC Instruction List

Ladder Diagram



12 Instructions for Special Function Modules

Instructions	Function
Instructions for serial communication modules	Reading of received data in an interrupt program; Reading, registration or deletion of user frames; Transmission of data using user frames
Instructions for PROFIBUS/DP interface modules	Reading or writing of data from and to the buffer memory of a PROFIBUS/DP interface module
Instructions for ETHERNET interface modules	Writing and reading of data to and from fixed buffer; Opening and closing of connections, Clearing of error codes; Re-initialization of the ETHERNET interface module
Instructions for MELSECNET/H	Setting of stations for duplex network
Instructions for CC-Link	Parameter setting, Setting of automatic refresh parameters Reading of data from the buffer memory of an station connected to CC-Link or from the PLC CPU of this station; Writing of data to the buffer memory of an station connected to CC-Link or to the PLC CPU of this station; Reading and writing from and to the automatic updated buffer memory

12.1 Instructions for serial communication modules

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Reading of received data from a QJ71C24 in an interrupt program	Z.BUFRCVS	BUFRCVS_M
Reading of user registered frames	G.GETE	GETE_M
	GP.GETE	GETEP_M
Registration or deletion of user registered frames	G.PUTE	PUTE_M
	GP.PUTE	PUTEP_M
Transmission of user frames	G.PRR	PRR_M
	GP.PRR	PRRP_M

12.1.1 BUFRCVS

CPU

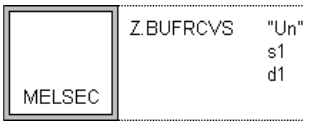
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

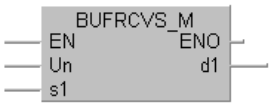
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
d1	●	●	●	—	—	—	—	—	—

GX IEC Developer

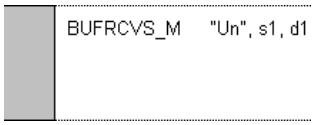
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

Set Data	Meaning	Range	Contents is stored by	Data Type
„Un“	Head I/O address of the serial communication module (The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)	0 to FEH	User	BIN 16-bit
s1	Reception channel number 1: Channel 1 (CH1) 2: Channel 2 (CH2)	1 or 2		
d1	Head number of the devices that stores received data			Address

Set Data	Meaning	Description	Range	Contents is stored by
(d1)+0	Data length	Length of the received data The unit (bytes or words) is set in the parameters.	—	System
(d1)+1 to (d1)+n	Received data	In this area the data read from the receive area of the buffer memory is stored sequencely in ascending order.		

Functions

Reading of received data from the QJ71C24

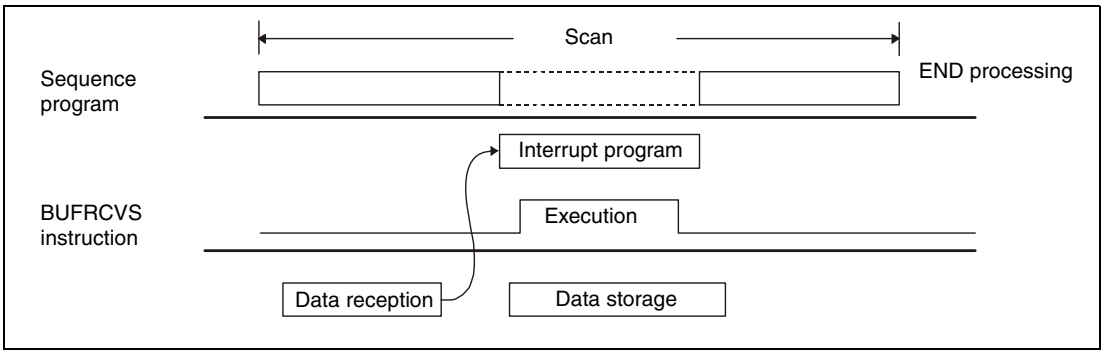
BUFRCVS Data read

The BUFRCVS instruction reads data sent from an external device to the communication module QJ71C24 from the buffer memory of the QJ71C24 and stores the data in the CPU module.

The BUFRCVS instruction can identify the address of the reception area in the buffer memory and read relative receive data to the area designated with d1.

When the data transfer is completed, the reception data read request (X3/XA) or the reception abnormal detection signal (X4/XB) is turned off automatically. It is not necessary to turn on the reception data completion signal (Y1/Y8) when received data is read by the BUFRCVS instruction.

The BUFRCVS instruction is used by an interrupt program and its processing is completed in one scan. The following figure shows the timing when the BUFRCVS instruction is being executed:



NOTES

When received data is read with a BUFRCVS instruction in an interrupt program, the data of the same interface can not be read again in the main program. Thus the BUFRCVS instruction cannot used together with the following instructions:

- the INPUT instruction

- the BIDIN instruction

- the FROM instruction in combination with input/output signals of the communication module

The BUFRCVS and the CSET instruction cannot be executed at the same time.

The area specified with d1 in the PLC CPU must be large enough to store all data sent from the external device. If this area is to small, the data that can not be stored, is lost.

Operation Errors

When the BUFRCVS instruction is completed abnormally, the error flag SM0 is set, and an error code is stored in SD0. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFFH or less, please refer to chapter 13 of this manual for error diagnostics.
- When the error code is 7000H or higher, please refer to the user's manual of the serial communication module QJ71C24.

If an error occurs during data reception (indicated by the input signals X4 and XB), the error code is written to the buffer memory addresses 258H and 268H of the communication module and can be used for diagnostics.

Program Example

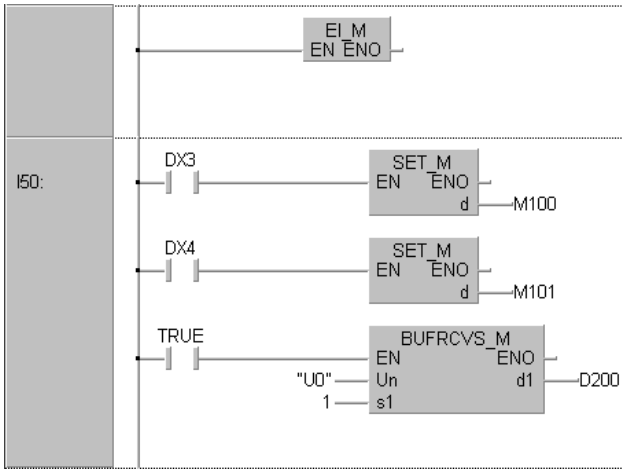
BUFRCVS

The following program reads the data received via channel 1 of a QJ71C24 with the head address X/Y0 and stores the data from D200 onward. Only channel 1 issues an interrupt. When data is received, the interrupt program 50 (I50) is processed. The internal relays M100 and M101 are used as interface with the main program. If data was received correctly, M100 is set. When an error occurs during reception of the data, M101 is set. Both relays are reset in the main program.

MELSEC Instruction List

```
        EI
        FEND
MELSEC
I50     LD      DX3
        SET     M100
MELSEC  LD      DX4
        SET     M101
        LD      M400
        Z.BUFRCVS "U0"
                  K1
                  D200
        IRET
```

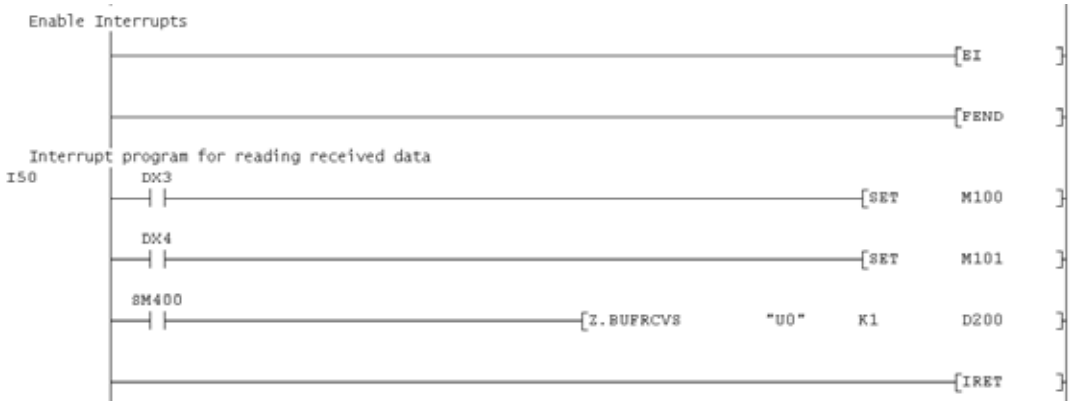
Ladder Diagram (GX IEC Developer)



IEC Instruction List

```
        LD      TRUE
        EI_M
I50:    LD      DX3
        SET_M   M100
        LD      DX4
        SET_M   M101
        LD      TRUE
        BUFRCVS_M "U0",  1,  D200
```

Ladder Diagram (GX Works2)



12.1.2 GETE, GETEP

CPU

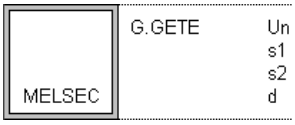
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

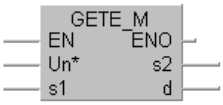
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
d	●	●	●	—	—	—	—	—	—

GX IEC Developer

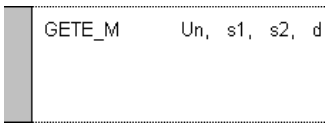
MELSEC Instruction List



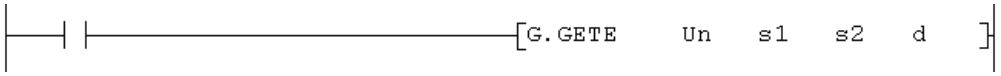
Ladder Diagram



IEC Instruction List



GX Works2



Variables

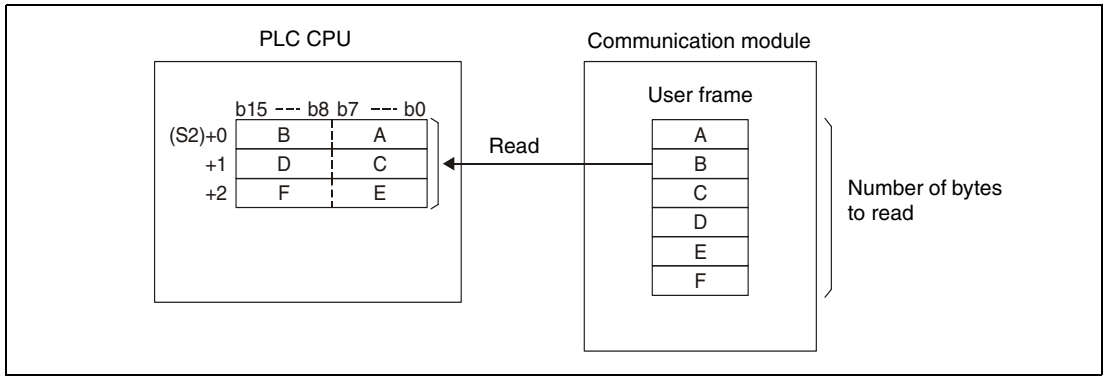
<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>Un</td><td colspan="4">Head I/O address of the serial communication module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)</td><td>0 to FE_H</td><td>User</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="6">s1</td><td colspan="4">Head number of the devices that store control data</td><td></td><td></td><td rowspan="6">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s1)+0</td><td>Dummy</td><td>Used by the system</td><td>0</td><td>—</td></tr>
<tr><td>(s1)+1</td><td>Read result</td><td>Indicates whether an error has occured during execution of the instruction:
0000_H: No error
Any value other than 0000_H: An error has occured and the stored value is an error code.</td><td></td><td>System</td></tr>
<tr><td>(s1)+2</td><td>Frame number</td><td>Number of the user frame</td><td>1000 to 1199</td><td rowspan="2">User</td></tr>
<tr><td>(s1)+3</td><td>Number of bytes to read</td><td>Max. number of bytes of the user frame that can be stored in the area specified by s2</td><td>1 to 80</td></tr>
<tr><td></td><td>Number of read bytes</td><td>Number of bytes of the user frame that has been read</td><td>1 to 80</td><td>System</td></tr>
<tr><td>s2</td><td colspan="4">Head number of the devices that store the read data</td><td>User
System</td><td>Address</td></tr>
<tr><td rowspan="4">d</td><td colspan="4">Bit device which is set for one scan after completion of the GETE instruction. (d)+1 indicates an abnormal completion of the instruction.</td><td></td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d)+0</td><td>Instruction completed</td><td>Indicates the completion of the GETE instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d)+1</td><td>Instruction completed with error</td><td>Indicates the abnormal completion of the GETE instruction.
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions

Reading of user registered frames

GETE Data read

The GETE instruction reads data from a user frame in a serial communication module and stores the data in the PLC CPU. The head address of the communication module is specified with Un.

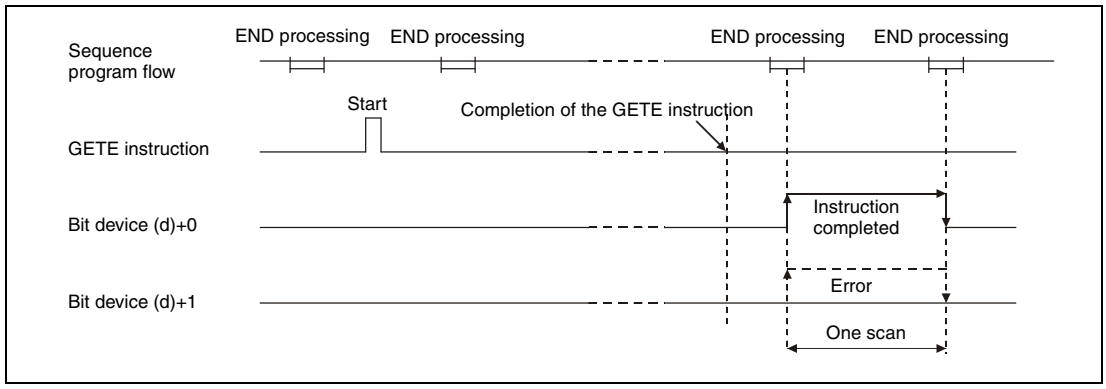


During GETE instruction execution, another GETE or PUTE instruction cannot be executed. If an attempt is made to execute a GETE or PUTE instruction during execution of a GETE instruction, the system waits until the execution of the instruction already being processed is completed.

Whether the execution of the GETE instruction has been finished can be checked with the devices (d)+0 and (d)+1:

- The bit device (d)+0 turns ON at the END processing of the scan in which the GETE instruction has been completed and turns OFF at the next END processing.
- The bit device (d)+1 indicates an error during execution of the GETE instruction. When the instruction is completed normal, this device stays OFF. When an error occurs during execution of the GETE instruction, (d)+1 turns ON at the END processing of the scan in which the GETE instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the GETE instruction is being executed:



Operation Errors

When an error occurs during execution of the GETE instruction, the bit device (d)+1 is set and an error code is written to (s1)+1. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFFH or less, refer to chapter 13 of this manual for error diagnostics.
- When the error code is 7000H or higher, you will find more information in the user's manual of the serial communication module.

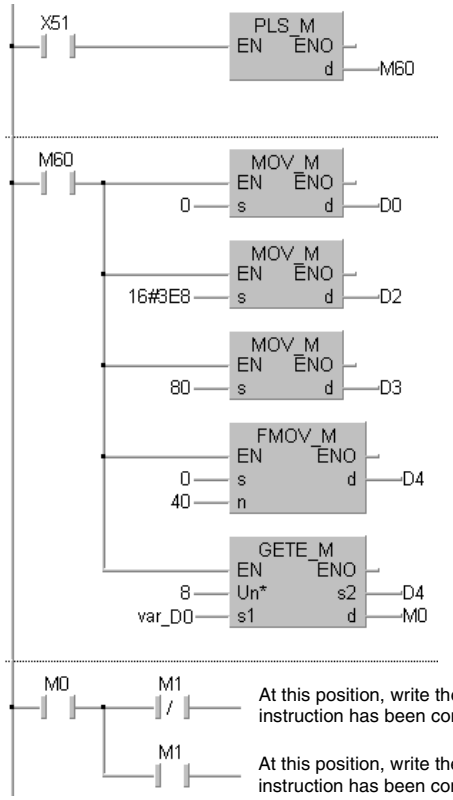
Program Example

GETE

The following program reads data of the user frame with the number 3E8H from a QJ71C24 and stores the data in the QCPU from data register D4 onward. The communication module occupies the input/output signals from X/Y80 to X/Y9F.

- IEC editors

Ladder Diagram (GX IEC Developer)



IEC Instruction List

```
LD          X51
PLS_M       M60
```

```
LD          M60
MOV_M       0,          D0
MOV_M       16#3E8,     D2
MOV_M       80,         D3
FMOV_M      0,          40,         D4
GETE_M      8,          var_D0,     D4,     M0
```

The devices and instructions used are explained in the above ladder diagram.

```
LD          M0
ANDN        M1
```

An instruction at his position will be executed when the GETE instruction has been completed normally.

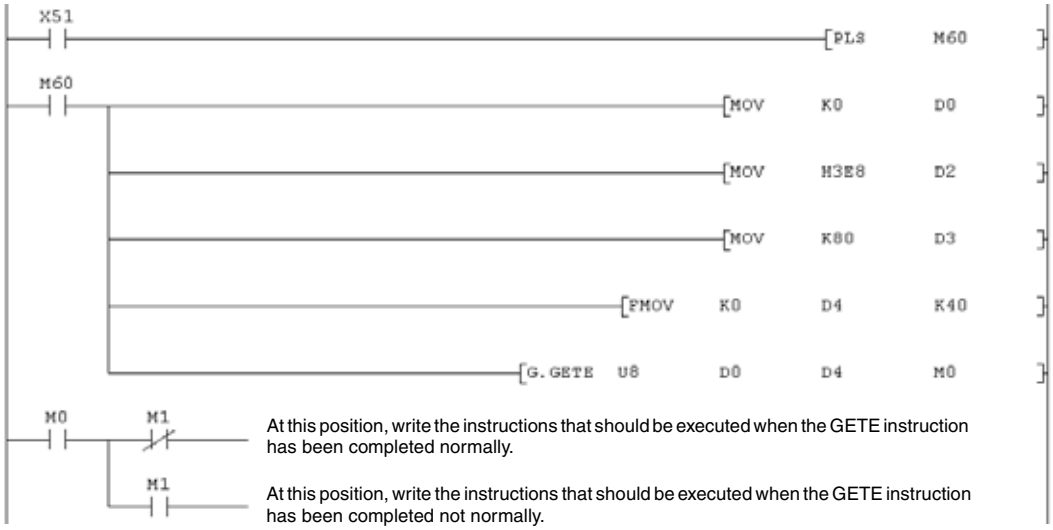
```
LD          M0
AND         M1
```

An instruction at this position will be executed when the GETE instruction has been completed with an error.

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



MELSEC Instruction List

MELSEC

```
LD      X51
PLS     M60
```

MELSEC

```
LD      M60
MOV     K0    D0
MOV     H3E8  D2
MOV     K80   D3
FMOV    K0    D4  K40
G.GETE  U8    D0  D4   M0
```

MELSEC

```
LD      M0
MPS
ANI     M1
```
An instruction at this position will be executed when the GETE instruction has been completed normally
```
MPP
AND     M1
```
An instruction at this position will be executed when the GETE instruction has been completed with an error.

12.1.3 PUTE, PUTEP

CPU

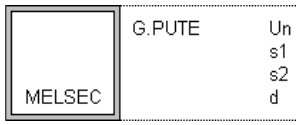
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

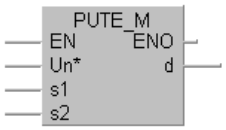
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
d	●	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List



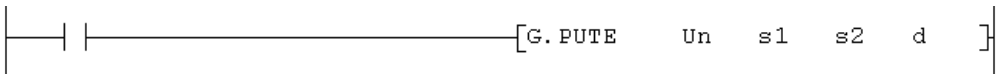
Ladder Diagram



IEC Instruction List

```
PUTE_M    Un, s1, s2, d
```

GX Works2



Variables

Set Data	Meaning	Range	Contents is stored by	Data Type
Un	Head I/O address of the serial communication module (The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)	0 to FEH	User	BIN 16-bit

Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="6">s1</td><td colspan="5">Head number of the devices that store control data</td><td rowspan="6">BIN 16-bit</td></tr>
<tr><td></td><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td></td><td>(s1)+0</td><td>Selection: Register or delete user frame</td><td>Designate whether to register or to delete the user frame specified by (s1)+2:
• 1: Register
• 3: Delete</td><td>1 or 3</td><td>User</td></tr>
<tr><td></td><td>(s1)+1</td><td>Register/delete result</td><td>Indicates whether an error has occured during execution of the instruction:
0000H: No error
Any value other than 0000H: An error has occured and the stored value is an error code.</td><td>—</td><td>System</td></tr>
<tr><td></td><td>(s1)+2</td><td>Frame No.</td><td>Number of the user frame to register or to delete</td><td>1000 to 1199</td><td rowspan="2">User</td></tr>
<tr><td></td><td>(s1)+3</td><td>Number of bytes to register</td><td>Number of bytes of the user frame to be registered. Please set also a value between 1 and 80 as dummy when deleting a user frame [(s1)+0 = 3].</td><td>1 to 80</td></tr>
<tr><td>s2</td><td colspan="4">Head number of the devices that store the data to be registered.</td><td>User</td><td>Address</td></tr>
<tr><td rowspan="4">d</td><td colspan="5">Bit device which is set for one scan after completion of the PUTE instruction. (d)+1 indicates an abnormal completion of the instruction.</td><td rowspan="4">Bit</td></tr>
<tr><td></td><td>Operand</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td></td><td>(d)+0</td><td>Instruction completed</td><td>Indicates the completion of the PUTE instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td></td><td>(d)+1</td><td>Instruction completed with error</td><td>Indicates the abnormal completion of the PUTE instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions **Registration or deletion of user frames**

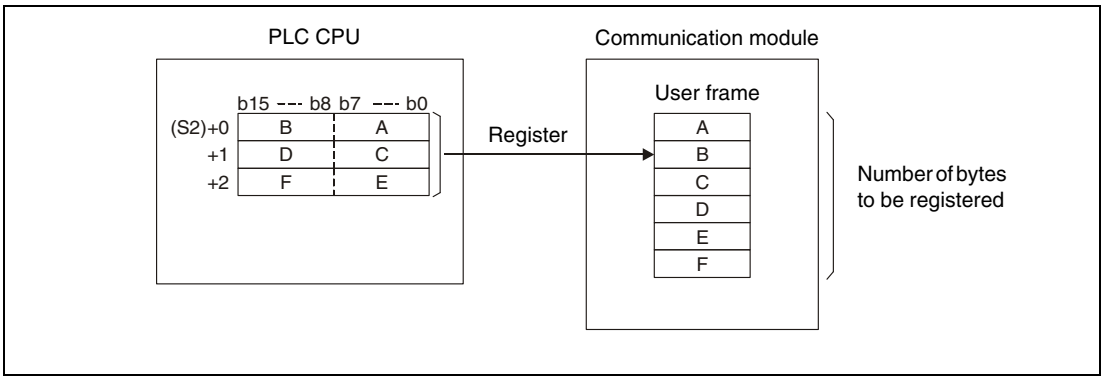
PUTE **Register or delete user frames**

The PUTE instruction is used to register or delete user frames in a serial communication module. The head address of the serial communication module is specified with Un.

Registering a user frame

When registering a user frame, write „1“ to the device designated with (s1)+0. Data from the devices starting with the device designated by s2 will be registered in accordance with the control data.

Since each device can store two bytes of data, the number of necessary devices equals half the number of data bytes.
If for instance six bytes are to be registered in a user frame, two additional devices must be reserved after s2:



Deletion of a user frame

To delete the user frame, whose number is written in (s1)+2, write „3“ to the device designated with (s1)+0.

Although the number of bytes [(s1)+3] and the area specified with s2 are not used during deletion, these settings are required for the PUTE instruction format. Write any value between 1 and 80 to the device designated by (s1)+3 and choose a dummy for s2.

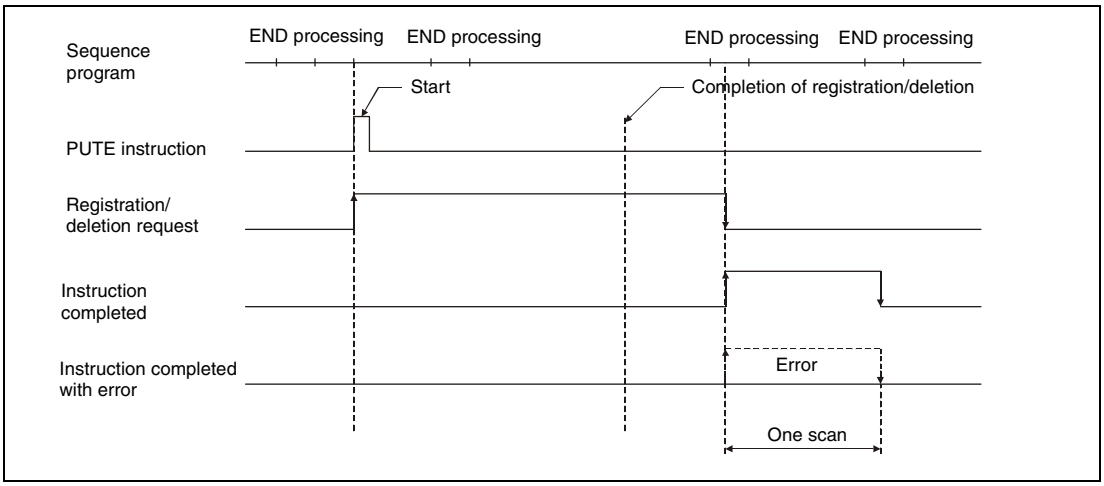
Operation conditions

During execution of a PUTE instruction, it is not possible to execute another PUTE or GETE instruction. If an attempt is made to execute one of these instructions when a PUTE instruction is already being executed, the system waits until the execution of the instruction already being processed is completed.

Whether the execution of the PUTE instruction has been finished or not can be checked with the devices (d)+0 and (d)+1:

- The bit device (d)+0 turns ON with the END processing of the scan in which the PUTE instruction has been completed and turns OFF at the next END processing.
- The bit device (d)+1 indicates an error during execution of the PUTE instruction. When the instruction is completed normal, this device stays OFF. When an error occurs during execution of the PUTE instruction, (d)+1 turns ON at the END processing of the scan in which the PUTE instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing for the PUTE instruction:



Operation Error

When an error occurs during execution of the PUTE instruction, the bit device (d)+1 is set and an error code is written to (s1)+1. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFFH or less, please refer to chapter 13 of this manual for error diagnostics.
- When the error code is 7000H or higher, please refer to the user's manual of the serial communication module.

Program Example

PUTE

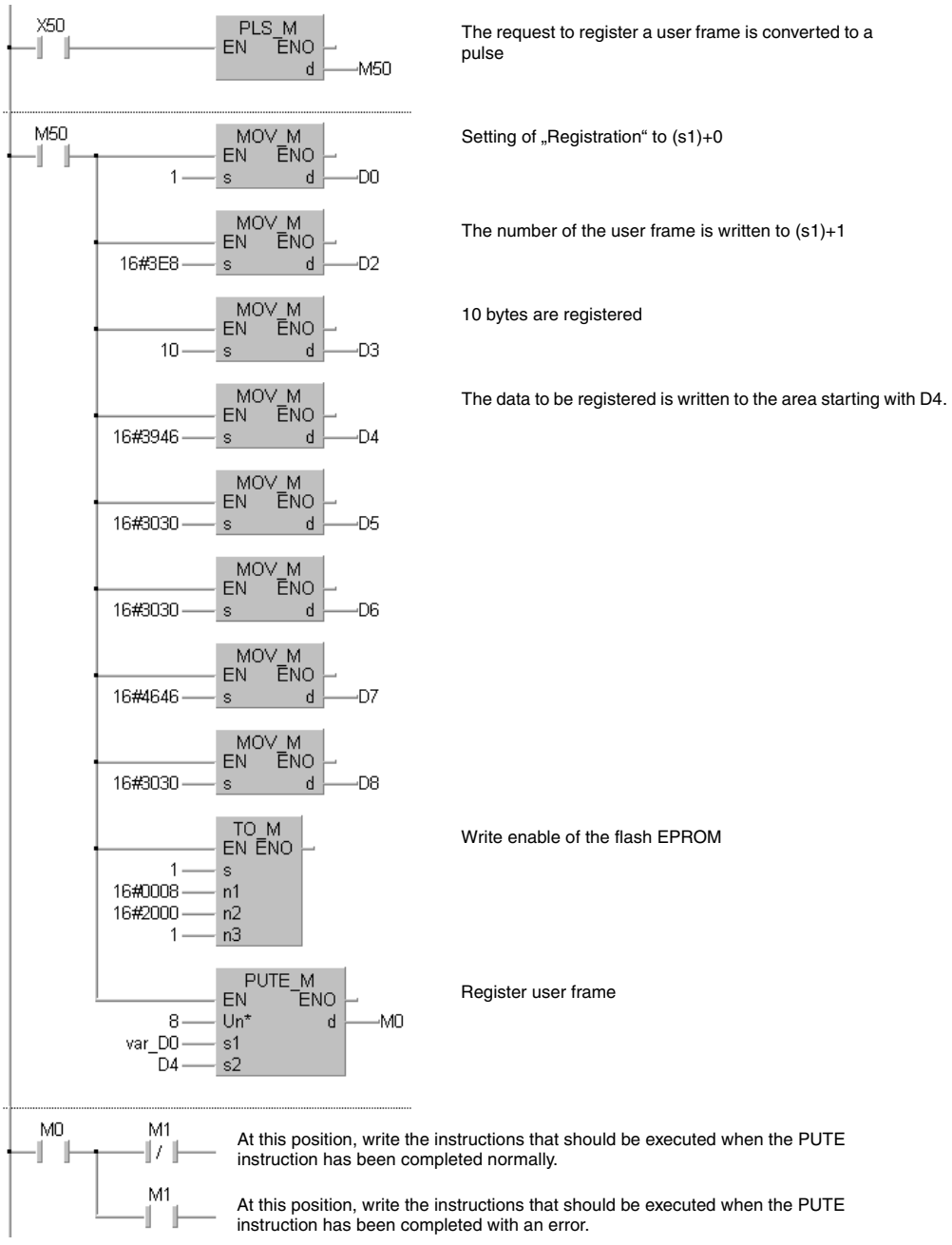
The following program registers data to the user frame with the number 3E8H. A QJ71C24 is used as communication module. It occupies the input/output signals from X/Y80 to X/Y9F.

NOTE

When using the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

- IEC editors

Ladder Diagram (GX IEC Developer)



IEC Instruction List

```
LD          X50
PLS_M       M50
```

```
LD          M50
MOV_M       1,          D0
MOV_M       16#3E8,     D2
MOV_M       10,         D3
MOV_M       16#3946,    D4
MOV_M       16#3030,    D5
MOV_M       16#3030,    D6
MOV_M       16#4646,    D7
MOV_M       16#3030,    D8
TO_M        1,          16#0008,    16#2000,    1
PUTE_M      8,          var_D0,     D4,         M0
```

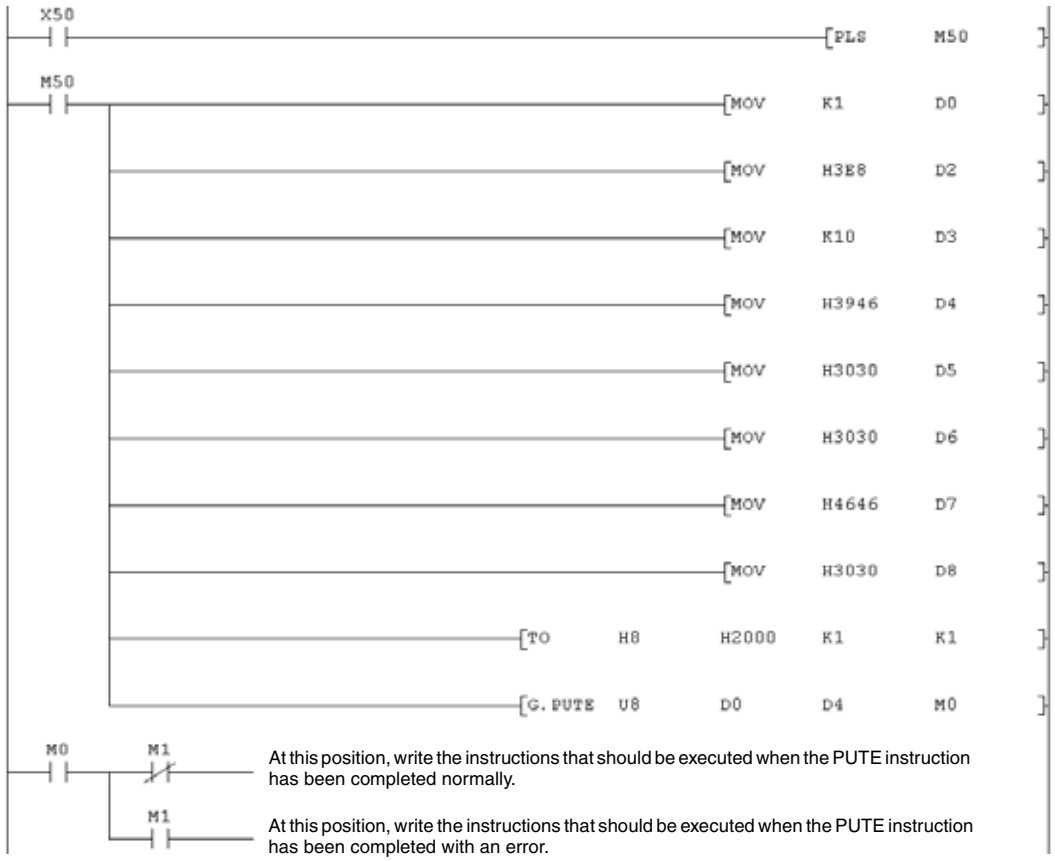
For an explanation of the devices and instructions used please see the ladder diagram on the previous page.

```
LD          M0
ANDN        M1
```
An instruction at this position will be executed when the PUTE instruction has been completed normally.

```
LD          M0
AND         M1
```
An instruction at this position will be executed when the PUTE instruction has been completed with an error.

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous pages.

Ladder Diagram (GX Works2)



At this position, write the instructions that should be executed when the PUTE instruction has been completed normally.

At this position, write the instructions that should be executed when the PUTE instruction has been completed with an error.

MELSEC Instruction List

MELSEC

```
LD      X50
PLS     M50
```

MELSEC

```
LD      M50
MOV     K1      D0
MOV     H3E8    D2
MOV     K10     D3
MOV     H3946   D4
MOV     H3030   D5
MOV     H3030   D6
MOV     H4646   D7
MOV     H3030   D8
TO      H8      H2000   K1      K1
G.PUTE  U8      D0      D4      M0
```

MELSEC

```
LD      M0
MPS
ANI     M1
```

An instruction at this position will be executed when the PUTE instruction has been completed normally.

```
MPP
AND     M1
```

At this position, write the instructions that should be executed when the PUTE instruction has been completed with an error.

12.1.4 PRR, PRRP

CPU

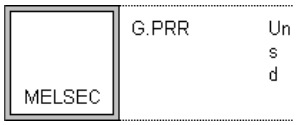
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

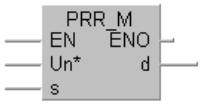
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
d1	●	●	●	—	—	—	—	—	—

GX IEC Developer

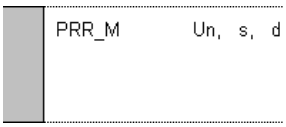
MELSEC Instruction List



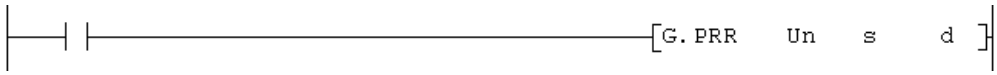
Ladder Diagram



IEC Instruction List



GX Works2



Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>Un</td><td colspan="4">Head I/O address of the serial communication module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)</td><td>0 to FEH</td><td>User</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="7">s</td><td colspan="5">Head number of the devices that store control data.</td><td rowspan="7">Address</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s)+0</td><td>Transmission channel</td><td>Designation of the channel used to transmit data
1: Channel 1 (CH1)
2: Channel 2 (CH2)</td><td>1 or 2</td><td>User</td></tr>
<tr><td>(s)+1</td><td>Transmission result</td><td>Indicates whether an error has occured during execution of the instruction:
0000H: No error
Any value other than 0000H: An error has occured and the stored value is an error code.</td><td>—</td><td>System</td></tr>
<tr><td>(s)+2</td><td>Addition of CR/LF</td><td>Designate whether or not to add CR/LF to the transmission data
0: Do not add CR/LF
1: Add CR/LF</td><td>0 or 1</td><td rowspan="3">User</td></tr>
<tr><td>(s)+3</td><td>Transmission pointer</td><td>Pointer to the first address of the device area which stores the data to be transmitted.</td><td>1 to 100</td></tr>
<tr><td>(s)+4</td><td>Number of user frames</td><td>Designation of the number of user frames to be transmitted.</td><td>1 to 100</td></tr>
<tr><td rowspan="4">d</td><td colspan="5">Bit device which is set for one scan after completion of the PRR instruction. (d)+1 indicates an abnormal completion of the instruction.</td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d)+0</td><td>Instruction completed</td><td>Indicates the completion of the PRR instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d)+1</td><td>Instruction completed with error</td><td>Indicates the abnormal completion of the PRR instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions

Transmission of user frames

PRR Transmit user frames

The PRR instruction transmits data using user frames to the communication module designated by Un. Information about the processing of the instruction are stored from the device designated by s. The contents of the user frames has to be set in the communication module before the PRR instruction is executed.

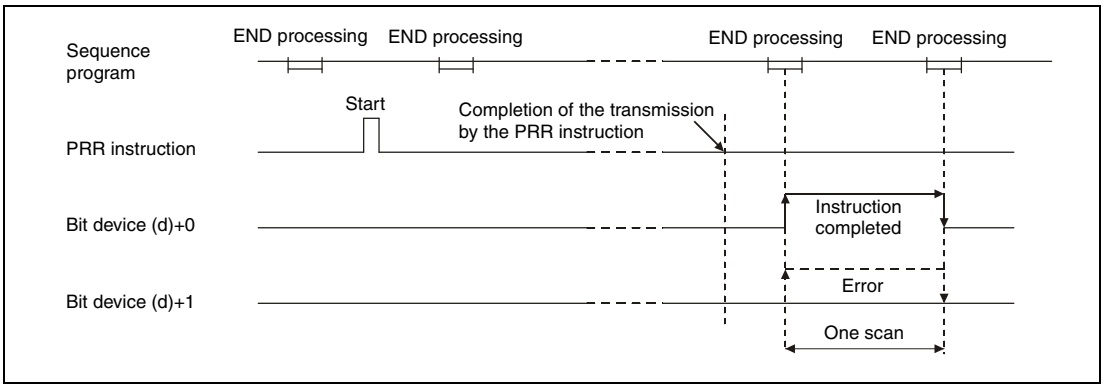
While a PRR instruction is being executed the following instructions cannot be executed for the same channel of the commnication module:
OUTPUT instruction, ONDEMAND instruction, BIDOUT instruction and other PRR instructions.

If an attempt is made to execute any of the above instructions while an PRR instruction is being executed, the system waits until the PRR instruction already being executed is completed.

Whether the execution of the PRR instruction has been finished can be checked with the devices (d)+0 and (d)+1:

- The bit device (d)+0 turns ON with the END processing of the scan in which the PRR instruction has been completed and turns OFF at the next END processing.
- The bit device (d)+1 indicates an error during execution of the PRR instruction. When the instruction is completed normal, this device stays OFF. When an error occurs during the execution of the PRR instruction, (d)+1 turns ON at the END processing of the scan in which the PRR instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing for the PRR instruction:



Operation Error

When an error occurs during execution of the PUTE instruction, the bit device (d)+1 is set and an error code is written to (s1)+1. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFFH or less, please refer to chapter 13 of this manual for error diagnostics.
- When the error code is 7000H or higher, please refer to the user's manual of the serial communication module.

Program Example

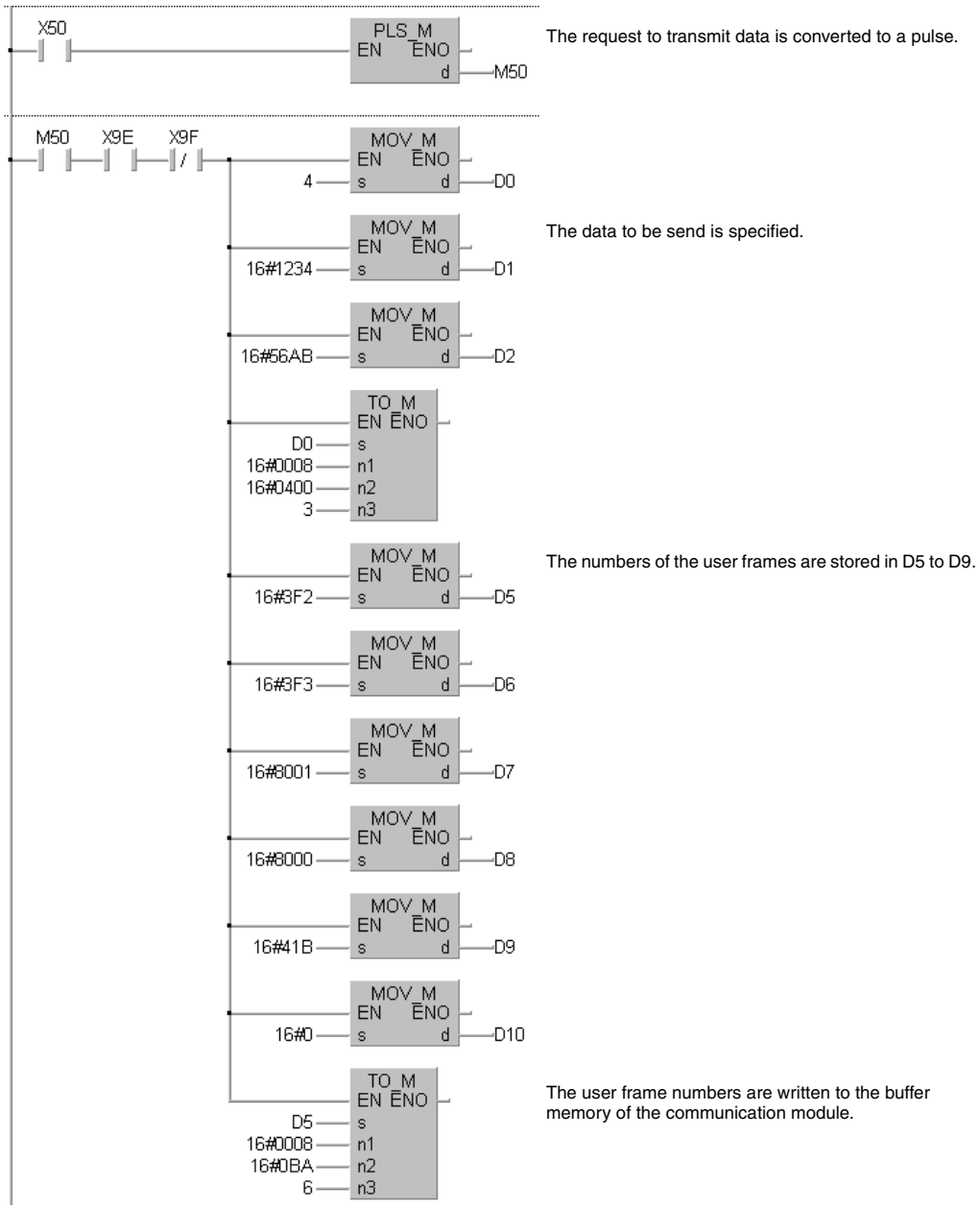
PRR

The program for this example transmits data and the first five user frames. The communication module QJ71C24 is used. It occupies the input/output signals from X/Y80 to X/Y9F. The following data registers are used in the program:

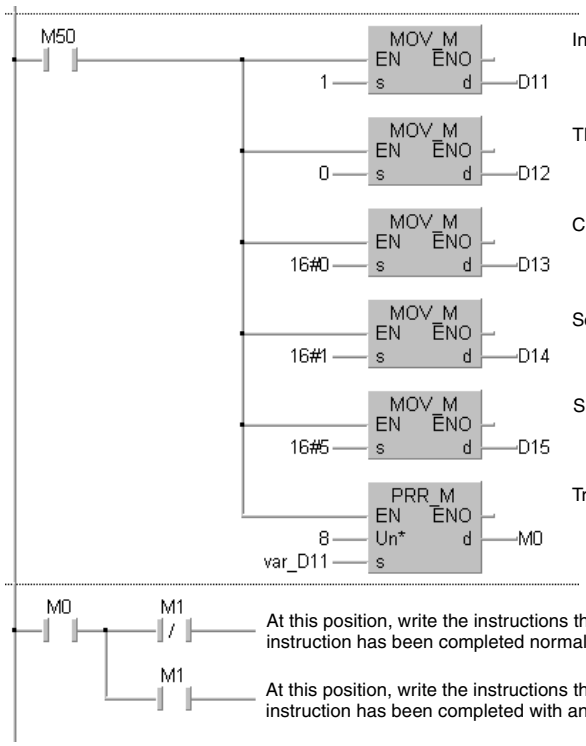
Data register	Contents	Meaning	
D0	0004H	Number of bytes to send	
D1	3412H	Data to be send	
D2	AB56H		
D5	03F2H	Numbers of the user frames	
D6	03F3H		
D7	8001H		
D8	8000H		
D9	041BH		
D10	0000H		
D11	0001H	(s)+0	Interface: CH1
D12	0000H or error code	(s)+1	Transmission result
D13	0000H	(s)+2	CR/LF is not added
D14	0001H	(s)+3	Transmission pointer
D15	0005H	(s)+4	Number of data frames to transmit

NOTE *When using the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- IEC editors
  Ladder Diagram of the GX IEC Developer (part 1)



Ladder Diagram of the GX IEC Developer (continued)



IEC Instruction List

```
LD      X50
PLS_M   M50
```

```
LD      M50
AND     X9E
ANDN    X9F
MOV_M   4,      D0
MOV_M   16#1234, D1
MOV_M   16#56AB, D2
TO_M    D0,     16#0008, 16#0400,3
MOV_M   16#3F2,  D5
MOV_M   16#3F3,  D6
MOV_M   16#8001, D7
MOV_M   16#8000, D8
MOV_M   16#41B,  D9
MOV_M   16#0,    D10
TO_M    D5,     16#0008, 16#0BA,  6
```

For an explanation of the devices and instructions used please see the above ladder diagram.

```
LD      M50
MOV_M   1,              D11
MOV_M   0,              D12
MOV_M   16#0,           D13
MOV_M   16#1,           D14
MOV_M   16#5,           D15
PRR_M   8,              var_D11,      M0
```

```
LD      M0
ANDN    M1
```

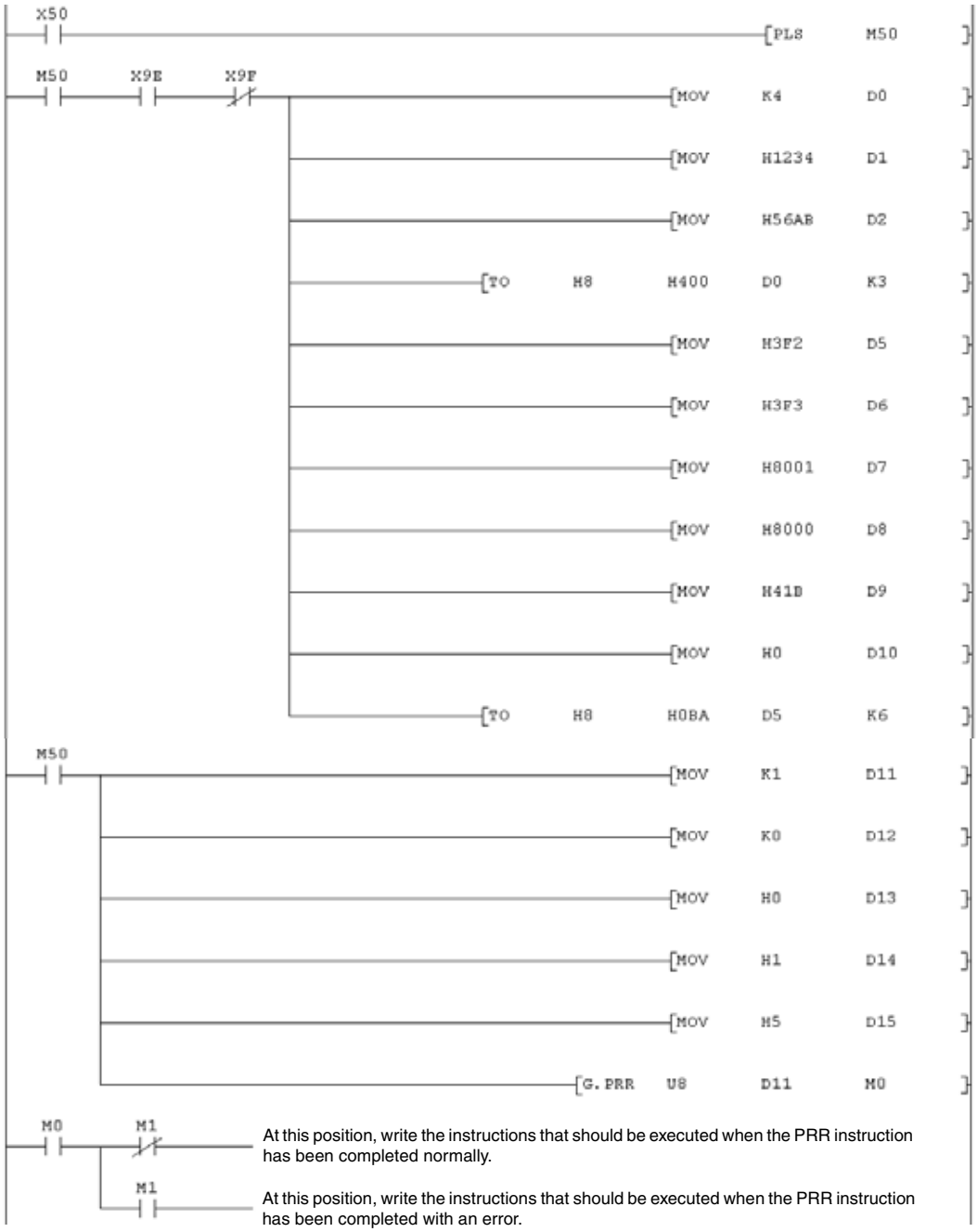
An instruction at this position will be executed when the PRR instruction has been completed normally.

```
LD      M0
AND     M1
```

An instruction at this position will be executed when the processing of the PRR instruction has been completed with an error.

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer.

Ladder Diagram (GX Works2)



MELSEC Instruction List

```
MELSEC
        LD      X50
        PLS     M50

MELSEC
        LD      M50
        AND     X9E
        ANI     X9F
        MOV     K4      D0
        MOV     H1234   D1
        MOV     H56AB   D2
        TO      H8      H400    D0      K3
        MOV     H3F2    D5
        MOV     H3F3    D6
        MOV     H8001   D7
        MOV     H8000   D8
        MOV     H41B    D9
        MOV     H0      D10
        TO      H8      H0BA    D5      K6

MELSEC
        LD      M50
        MOV     K1      D11
        MOV     K0      D12
        MOV     H0      D13
        MOV     H1      D14
        MOV     H5      D15
        G.PRR   U8      D11     M0

MELSEC
        LD      M0
        MPS
        ANI     M1
An instruction at this position will be executed when the PRR instruction has been completed normally.
        MPP
        AND     M1
At this position, write the instructions that should be executed when the PRR instruction has been completed with an error.
```

12.2 Instructions for PROFIBUS/DP interface modules

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Reading of data from the buffer memory of a PROFIBUS/DP interface module	G.BBLKRD	BBLKRD_M
	GP.BBLKRD	BBLKRDP_M
Writing of data to the buffer memory of a PROFIBUS/DP interface module	G.BBLKWR	BBLKWR_M
	GP.BBLKWR	BBLKWRP_M

12.2.1 BBLKRD, BBLKRDP

CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

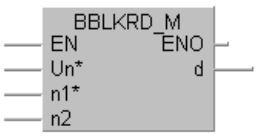
Devices MELSEC Q

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
n1	—	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n2	—	●	●	—	—	—	—	●	—

GX IEC Developer

MELSEC Instruction List

Ladder Diagram



IEC Instruction List

GX Works2

Variables

Set Data	Meaning	Data Type
Un	Head I/O number of the PROFIBUS interface module on the base unit	BIN 16-bit
n1	Head address of the buffer memory of the PROFIBUS interface module from where the reading of the data is started.	
d	Head address of the device area in the PLC CPU where the read data is stored	Device name
n2	Number of data to read	BIN 16-bit

Functions

Reading of data from the buffer memory of a PROFIBUS interface module

BBLKRD / BBLKRDP Reading of data

The BBLKRD instruction is used to read data from the buffer memory of the PROFIBUS interface modules QJ71PB92D and QJ71PB93D. While reading, data separation is prevented.

The QJ71PB93 must be prepared for the BBLKRD instruction by setting of the output signal Y0A. When the PROFIBUS module in turn sets the input signal X0A, the BBLKRD instruction can be executed. The output signal Y0A must be reset when the reading of the buffer memory is completed.

Allowable ranges and designation of the devices:

- Un (Head I/O address of the PROFIBUS interface module): 0 to FFH
  (Only the upper two digits of the 3-digit-address are used. E. g. the head address X/Y100 is set as 10H.)
- n1 (Head address in the buffer memory): The specified address must be exist.
- d (Head address of the target area): The designated device must be exist.
- n2 (Number of data to read)
  For a QJ71PB92D: 1 to 960 words (1 to 3C0H)
  For a QJ71PB93D: 1 to 122 words (1 to 7AH)

NOTES

Only a single BBLKRD instruction can be executed in one scan.

The BBLKRD and the BBLKWR instruction (section 12.2.2) are working independently.

The transmision delay time increases when the BBLKRD instruction is used.

The BBLKRD instruction is not executed when the output module has not been set in the data module setting in the master station parameter.

Operation Error

In the following cases an operation error occurs, the error flag SM0 is set, and an error code is stored in SD0:

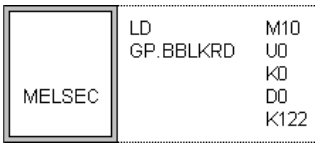
- When a value that exceeds the specificable range is set for the set data. (Error code: 4101)
- By the addition of the head address of the buffer memory designated by n1 and the number of data to be read designated by n2 the size of the buffer memory is exceeded. (Error code: 4101)
- The number of data to be read (designated by n2) is larger than the available device area starting with the head address designated by d. (Error code: 4101)

Program Example

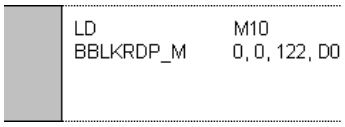
BBLKRDP

When the relay M10 is set, 122 words of data are read from the buffer memory of the PROFIBUS interface module with the head I/O address X/Y0. The reading is started at the buffer memory address 0 while the storage of the data is started from register D0 onward.

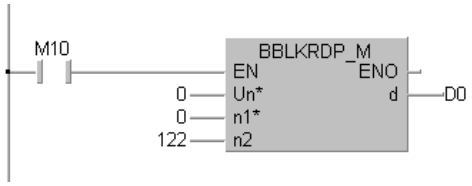
MELSEC Instruction List



IEC Instruction List



Ladder Diagram (GX IEC Developer)



Ladder Diagram (GX Works2)

12.2.2 BBLKWR, BBLKWRP

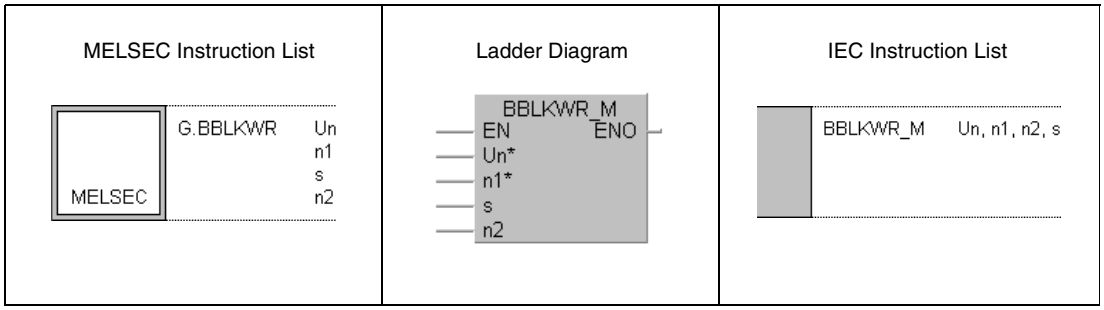
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

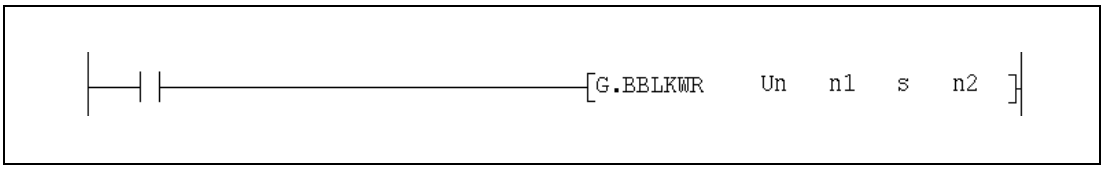
Devices MELSEC Q

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
n1	—	●	●	—	—	—	—	●	—
s	—	●	●	—	—	—	—	—	—
n2	—	●	●	—	—	—	—	●	—

GX IEC Developer



GX Works2



Variables

Set Data	Meaning	Data Type
Un	Head I/O number of the PROFIBUS interface module on the base unit	BIN 16-bit
n1	Head address of the buffer memory of the PROFIBUS interface module from where the writing of the data is started.	
s	Head address of the device area in the PLC CPU where the data is stored that is to be send to the PROFIBUS interface module.	Device name
n2	Number of data to be send to the PROFIBUS interface module	BIN 16-bit

Functions

Writing of data to the buffer memory of a PROFIBUS interface module

BBLKWR / BBLKWRP Writing of data

The BBLKWR instruction writes data to the buffer memory of the PROFIBUS interface modules QJ71PB92D and QJ71PB93D. Data separation is prevented during the write operation.

The QJ71PB93 must be prepared for the BBLKWR instruction by setting of the output signal Y0B. When the PROFIBUS module in turn sets the input signal X0B, the BBLKWR instruction can be executed. After completion of the writing to the buffer memory the output signal Y0B must be reset.

Allowable ranges and designation of the devices:

- Un (Head I/O address of the PROFIBUS interface module): 0 to FFH
  (Only the upper two digits of the 3-digit-address are used. E. g. the head address X/Y100 is set as 10H.)
- n1 (Head address in the buffer memory): The specified address must be exist.

  The head address for the QJ71PB93 has an offset of 100H. Thus, 100H must be subtracted from the desired head address when designating n1. For example the head address 100H is specified as „0H“ and the head address 120H is specified as „20H“.
- d (Head address of the source area): The designated device must be exist.
- n2 (Number of data to write)
  For a QJ71PB92D: 1 to 960 words (1 to 3C0H)
  For a QJ71PB93D: 1 to 122 words (1 to 7AH)

NOTES

Only a single BBLKWR instruction can be executed in one scan.

The BBLKRD and the BBLKWR instruction (section 12.2.1) are working independently.

The transmision delay time increases when the BBLKWR instruction is used.

The BBLKRD instruction is not executed when the input module has not been set in the data module setting in the master station parameter.

Operation Error

In the following cases an operation error occurs, the error flag SM0 is set, and an error code is stored in SD0:

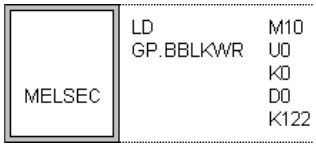
- When a value that exceeds the specificable range is set for the set data. (Error code: 4101)
- By the addition of the head address of the buffer memory designated by n1 and the number of data to write (designated by n2) the size of the buffer memory is exceeded.
  (Error code: 4101)
- The number of data to be write (designated by n2) is larger than the available device area starting with the head address designated by d. (Error code: 4101)

Program Example

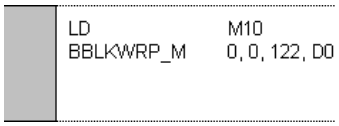
BBLKWRP

After the relay M10 is set, the contents of the data registers D0 to D121 (122 words) is written to the input area of the PROFIBUS/DP slave module QJ71PB93D. The input area starts at the buffer memory address 100H. Please note that the head address designated by n1 is specified with „0H“ in this case. The head I/O number of the PROFIBUS/DP slave module is X/Y0.

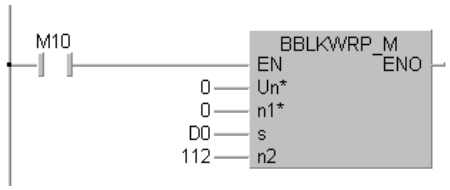
MELSEC Instruction List



IEC Instruction List



Ladder Diagram (GX IEC Developer)



Ladder Diagram (GX Works2)

12.3 Instructions for ETHERNET interface modules

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Reading of received data from fixed buffers	ZP.BUFRCV	BUFRCV_M
	Z.BUFRCVS	BUFRCVS_M
Sending of data to fixed buffers	ZP.BUFSND	BUFSND_M
Opening of a connection	ZP.OPEN	OPEN_M
Closing of a connection	ZP.CLOSE	CLOSE_M
Clearing of error information	ZP.ERRCLR	ERRCLR_M
Reading of error information	ZP.ERRRD	ERRRD_M
Reinitialization of a ETHERNET interface module	ZP.UINI	UINI_M

12.3.1 BUFRCV

CPU

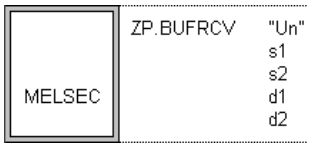
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

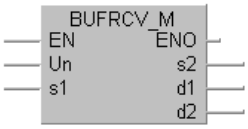
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	—	—
d1	—	●	●	—	—	—	—	—	—
d2	●	●	●	—	—	—	—	—	—

GX IEC Developer

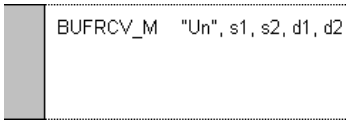
MELSEC Instruction List



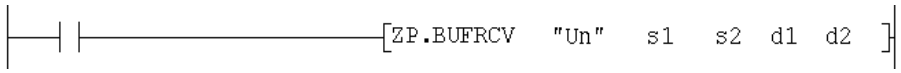
Ladder Diagram



IEC Instruction List



GX Works2



Variables

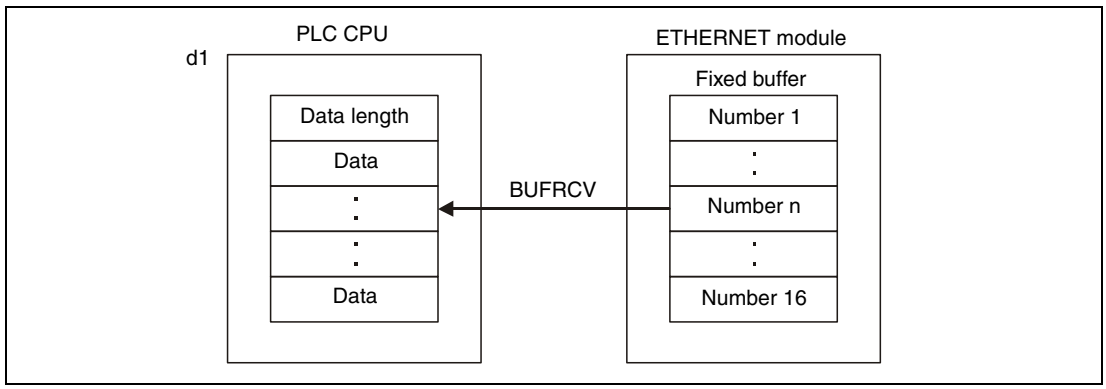
<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>„Un“</td><td colspan="4">Head I/O address of the ETHERNET interface module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)</td><td>0 to FE_H</td><td rowspan="2">User</td><td rowspan="2">BIN 16-bit</td></tr>
<tr><td>s1</td><td colspan="4">Connection number</td><td>1 to 16</td></tr>
<tr><td rowspan="4">s2</td><td colspan="6">Head number of the devices where control data for execution of this instruction is stored.</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td colspan="2">Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s2)+0</td><td>System area</td><td colspan="2">Used by the system</td><td rowspan="2">—</td><td rowspan="2">System</td></tr>
<tr><td>(s2)+1</td><td>Execution result of the instruction</td><td colspan="2">Indicates whether an error has occured during execution of the instruction:
0000_H: No error
Any value other than 0000_H: An error has occured. The stored value is an error code which is explained either in the user’s manual of the ETHERNET interface module or in chapter 13 of this manual.</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="5">d1</td><td colspan="6">Head number of the device area where the received data is stored.</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td colspan="2">Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td rowspan="3">(d1)+0</td><td rowspan="3">Length of the received data</td><td colspan="2">With procedure (binary data):
Number of words read from the fixed buffer</td><td>1 to 1017 words</td><td rowspan="4">System</td><td rowspan="4">BIN 16-bit</td></tr>
<tr><td colspan="2">With procedure (ASCII data):
Number of words read from the fixed buffer</td><td>1 to 508 words</td></tr>
<tr><td colspan="2">Without procedure (binary data):
Number of bytes read from the fixed buffer</td><td>1 to 2016 bytes</td></tr>
<tr><td>(d1)+1 to (d1)+n</td><td>Received data</td><td colspan="2">In this area the data read from the fixed buffer is stored sequentially in ascending order.</td><td>—</td></tr>
<tr><td rowspan="4">d2</td><td colspan="6">Bit device which is set for one scan after completion of the BUFRCV instruction. (d)+1 indicates an abnormal completion of the instruction.</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td colspan="2">Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d2)+0</td><td>Instruction completed</td><td colspan="2">Indicates the completion of the BUFRCV instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td><td rowspan="2">Bit</td></tr>
<tr><td>(d2)+1</td><td>Instruction completed with error</td><td colspan="2">Indicates the abnormal completion of the BUFRCV instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions

Reading of received data from fixed buffer (Execution of the instruction in the main program)

BUFRCV Data read

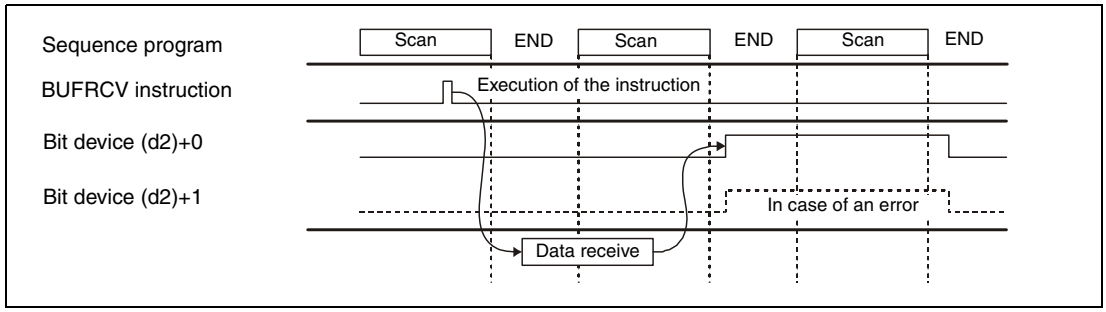
With the BUFRCV instruction, Data sent by an external Station to an ETHERNET interface module via fixed buffer communication can be read from the ETHERNET module and stored in the PLC CPU. The BUFRCV instruction is executed in the main program, whereas the BUFRCVS instruction is used in an interrupt program. Where the data should be stored is specified with d1:



Whether the execution of the BUFRCV instruction has been finished can be checked with the devices (d2)+0 and (d2)+1:

- The bit device (d2)+0 turns ON with the END processing of the scan in which the BUFRCV instruction has been completed and turns OFF at the next END processing.
- The bit device (d2)+1 indicates an error during execution of the BUFRCV instruction. When the instruction is completed normal, this device stays OFF. When an error occurs during the execution of the BUFRCV instruction, (d2)+1 turns ON at the END processing of the scan in which the BUFRCV instruction has been completed and turns OFF at the next END processing.

The timing for the PRR instruction is shown in the following figure:



The BUFRCV instruction can be executed when the ETHERNET interface module indicates that data has been received. One bit is reserved in the buffer memory address 5005H for each of the 16 possible connections and is set when data has been received.

NOTE

It is not possible to read received data of the same connection with the BUFRCV instruction in the main programm and the BUFRCVS instruction in an interrupt program.

Operation Error

When the BUFRCV instruction is completed abnormally, the bit device (d2)+1 is set, and an error code is stored in (s2)+1. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFFH or less, you will find more information in chapter 13 of this manual.
- When the error code is C001H or higher, please refer to the user's manual of the ETHERNET interface module.

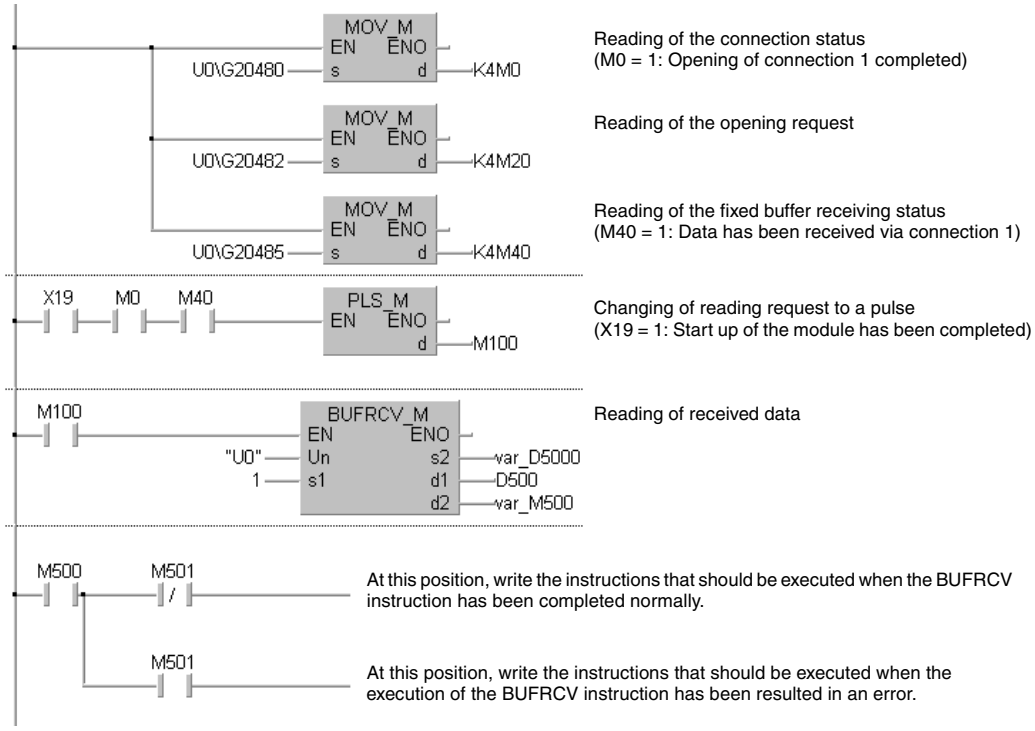
Program Example

BUFRCV

The following program reads received data from the fixed buffer for connection number 1. The input/output points X/Y0 to X/Y1F are occupied by the ETHERNET module.

- IEC editors (This program example is shown on the next page for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



IEC Instruction List

```
LD TRUE
MOV_M       U0\G20480, K4M0
MOV_M       U0\G20482, K4M20
MOV_M       U0\G20485, K4M40
LD          X19
AND         M0
AND         M40
PLS_M       M100
LD          M100
BUFRCV_M    "U0", K1, var_D5000, D500, var_M500
```

For an explanation of the devices and instructions used please see the above ladder diagram

```
LD          M500
ANDN        M501
```

At this position, write the instructions that should be executed when the BUFRCV instruction has been completed normally.

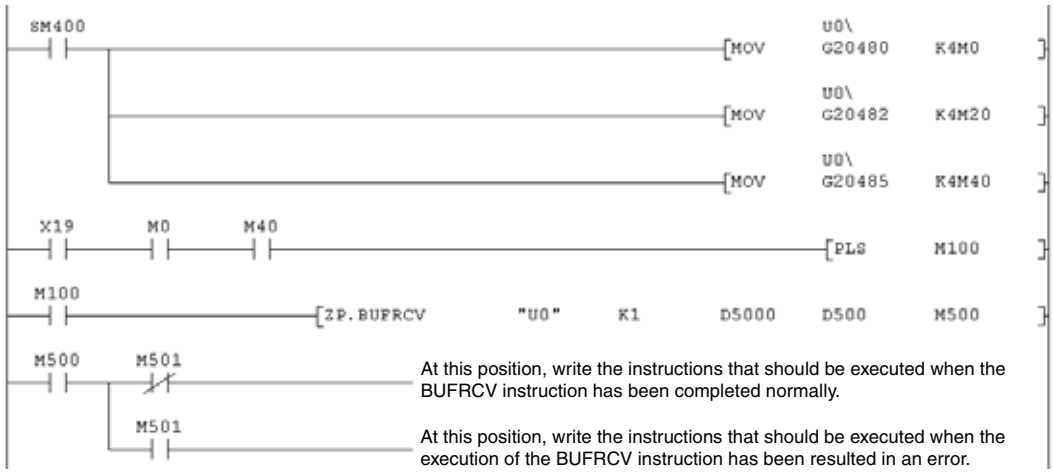
```
LD          M500
AND         M501
```

At this position, write the instructions that should be executed when the execution of the BUFRCV instruction has been resulted in an error.

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



MELSEC Instruction List

MELSEC	LD	SM400				
	MOV	U0\G20480	K4M0			
	MOV	U0\G20482	K4M20			
	MOV	U0\G20485	K4M40			
	LD	X19				
	AND	M0				
	AND	M40				
	PLS	M100				
	LD	M100				
	ZP.BUFRCV	"U0"	K1	D5000	D500	M500
MELSEC	LD	M500				
	ANI	M501				
	At this position, write the instructions that should be executed when the BUFRCV instruction has been completed normally.					
MELSEC	LD	M500				
	AND	M501				
	At this position, write the instructions that should be executed when the execution of the BUFRCV instruction has been resulted in an error.					

12.3.2 BUFRCVS

CPU

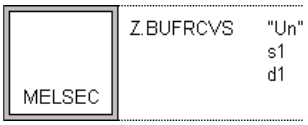
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

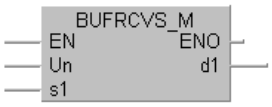
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
d1	—	●	●	—	—	—	—	—	—

GX IEC Developer

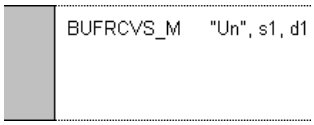
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

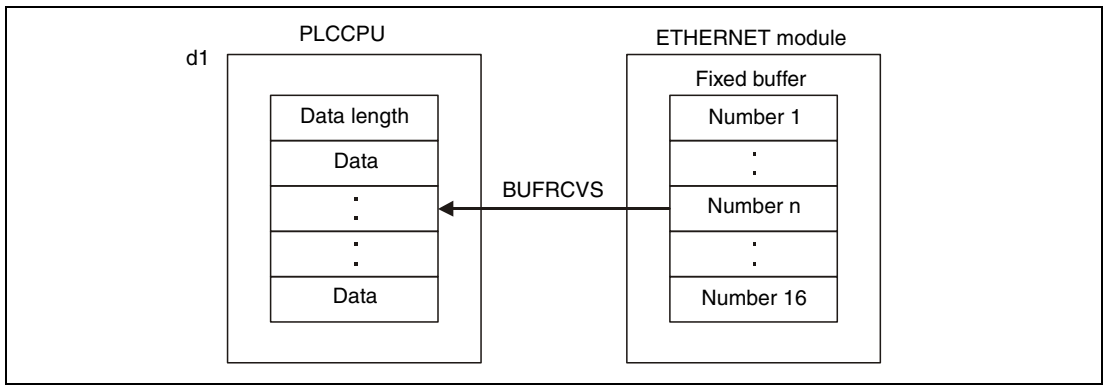
Set Data	Meaning			Range	Contents is stored by	Data Type
„Un“	Head I/O address of the ETHERNET interface module (The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)			0 to FEH	User	BIN 16-bit
s1	Connection number			1 to 16		
d1	Head number of the device area where the received data is stored.					BIN 16-bit
	Set Data	**Meaning**	**Description**	**Range**	**Contents is stored by**	
	(d1)+0	Length of the received data (Number of word or bytes read from the fixed buffer)	With procedure (binary data):)	1 to 1017 words	System	
			With procedure (ASCII data):	1 to 508 words		
			Without procedure (binary data):	1 to 2016 bytes		
	(d1)+1 to (d1)+n	Received data	In this area the data read from the fixed buffer is stored sequentially in ascending order.	—		

Functions

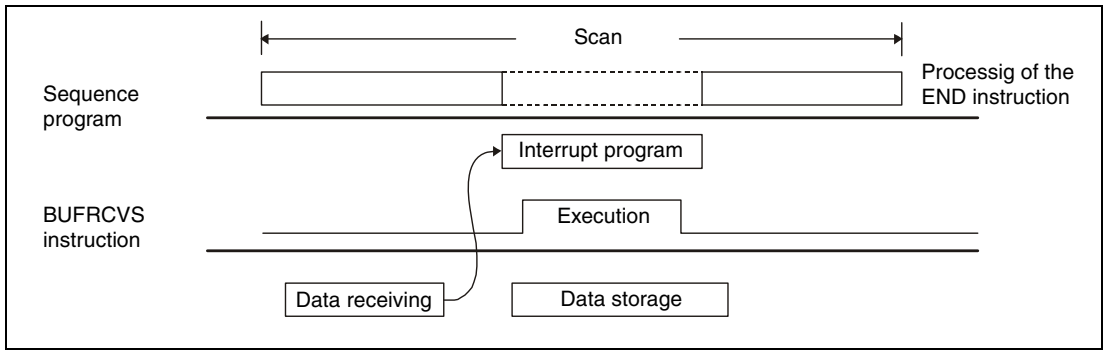
Reading of received data from fixed buffer (Execution of the instruction in an interrupt program)

BUFRCVS Data read

With the BUFRCVS instruction, Data sent by an external Station to an ETHERNET interface module via fixed buffer communication can be read from the ETHERNET module and stored in the PLC CPU. The BUFRCVS instruction is executed in an interrupt program, whereas the BUFRCV instruction is used in the main program. Where the data should be stored is specified with d1:



The processing of the BUFRCVS instruction is completed within one scan. The following figure shows the timing of the BUFRCVS instruction:



In order to read receive data with an interrupt program, it is necessary to perform both the interrupt settings and interrupt pointer settings with parameter settings of GX (IEC) Developer.

NOTES

It is not possible to read received data of the same connection with the BUFRCV instruction in the main programm and the BUFRCVS instruction in an interrupt program.

The BUFRCVS instruction can also used for an serial communication module QJ71C24 (see chapter 11.1.1).

Operation Error

When the BUFRCV instruction is completed abnormally, the error flag SM0 is set, and an error code is stored in SD0. For more information about the error codes please refer to the following manuals:

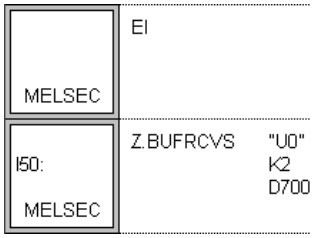
- When the error code is 4FFFH or less, you will find more information in chapter 13 of this manual.
- When the error code is C001H or higher, please refer to the user's manual of the ETHERNET interface module.

Program Example

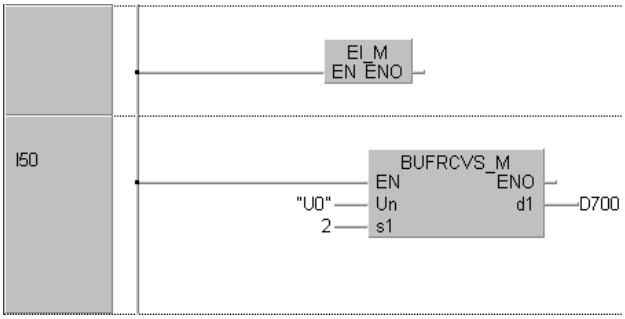
BUFRCVS

The following program reads received data from the fixed buffer for connection number 2. The head I/O number of the ETHERNET module is X/Y0.

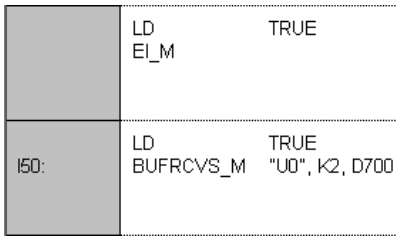
MELSEC Instruction List



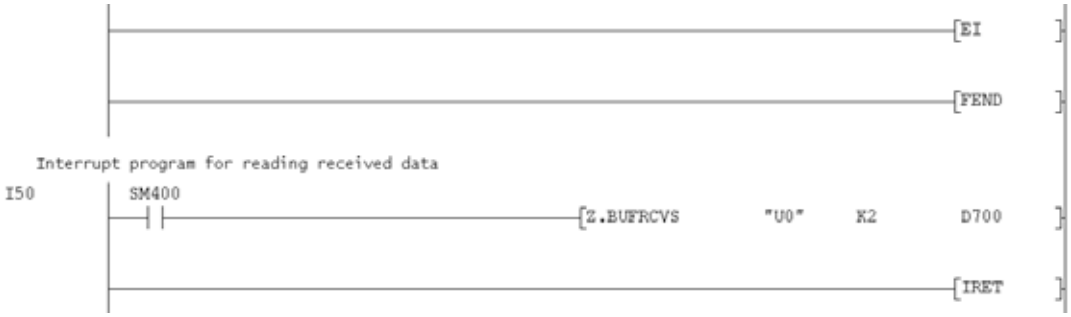
Ladder Diagram (GX IEC Developer)



IEC Instruction List



Ladder Diagram (GX Works2)



12.3.3 BUFSND

CPU

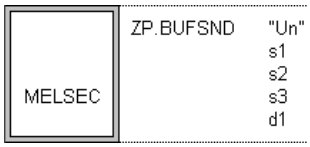
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

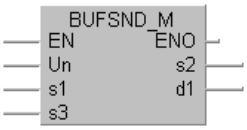
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	—	—
s3	—	●	●	—	—	—	—	—	—
d1	●	●	●	—	—	—	—	—	—

GX IEC Developer

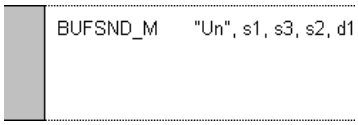
MELSEC Instruction List



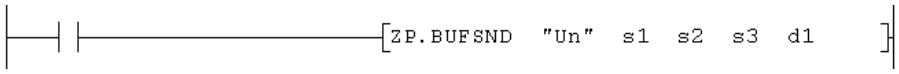
Ladder Diagram



IEC Instruction List



GX Works2



Variables

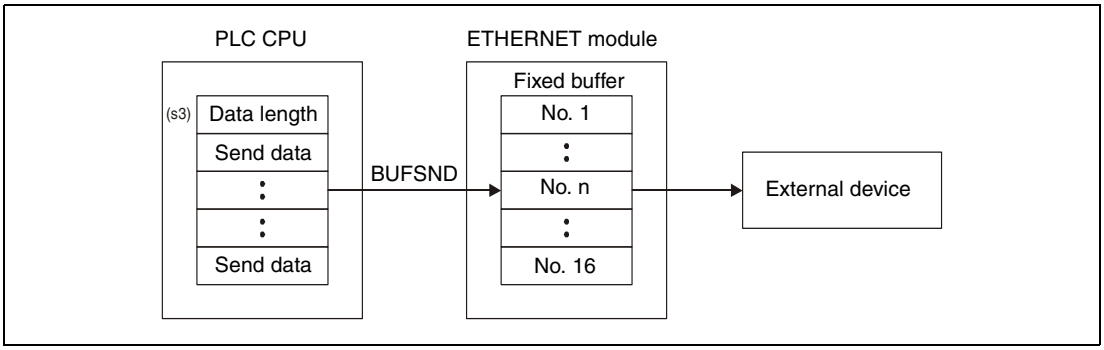
<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>„Un“</td><td colspan="4">Head I/O address of the ETHERNET interface module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)</td><td>0 to FEH</td><td rowspan="2">User</td><td rowspan="2">BIN 16-bit</td></tr>
<tr><td>s1</td><td colspan="4">Connection number</td><td>1 to 16</td></tr>
<tr><td rowspan="4">s2</td><td colspan="4">Head number of the devices where control data for execution of this instruction is stored.</td><td></td><td rowspan="4">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s2)+0</td><td>System area</td><td>Used by the system</td><td rowspan="2">—</td><td rowspan="2">System</td></tr>
<tr><td>(s2)+1</td><td>Execution result of the instruction</td><td>Indicates whether an error has occured during execution of the instruction:
0000H: No error
Any value other than 0000H: An error has occured. The stored value is an error code which is explained either in the user’s manual of the ETHERNET interface module or in chapter 13 of this manual.</td></tr>
<tr><td rowspan="5">s3</td><td colspan="4">Head number of the devices where the send data is stored.</td><td></td><td rowspan="5">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td rowspan="3">(s3)+0</td><td rowspan="3">Length of the data to be send</td><td>Designation of the amount of data that is to be transferred to the fixed buffer when a procedure (binary data) is used for communication.</td><td>1 to 1017 words</td><td rowspan="4">User</td></tr>
<tr><td>Designation of the amount of data that is to be transferred to the fixed buffer when a procedure (ASCII data) is used for communication.</td><td>1 to 508 words</td></tr>
<tr><td>Designation of the amount of data that is to be transferred to the fixed buffer when a non procedure protokoll (binary data) is used for communication.</td><td>1 to 2046 bytes</td></tr>
<tr><td>(s3)+1 to (s3)+n</td><td>Data to be send</td><td>The data stored in this are is send to the ETHERNET module.</td><td>—</td></tr>
<tr><td rowspan="4">d1</td><td colspan="4">Bit device which is set for one scan after completion of the BUFSND instruction. (d)+1 indicates an abnormal completion of the instruction.</td><td></td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d1)+0</td><td>Instruction completed</td><td>Indicates the completion of the BUFSND instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d1)+1</td><td>Instruction completed with error</td><td>Indicates the abnormal completion of the BUFSND instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions

Sending of data to fixed buffer

BUFSND Data send

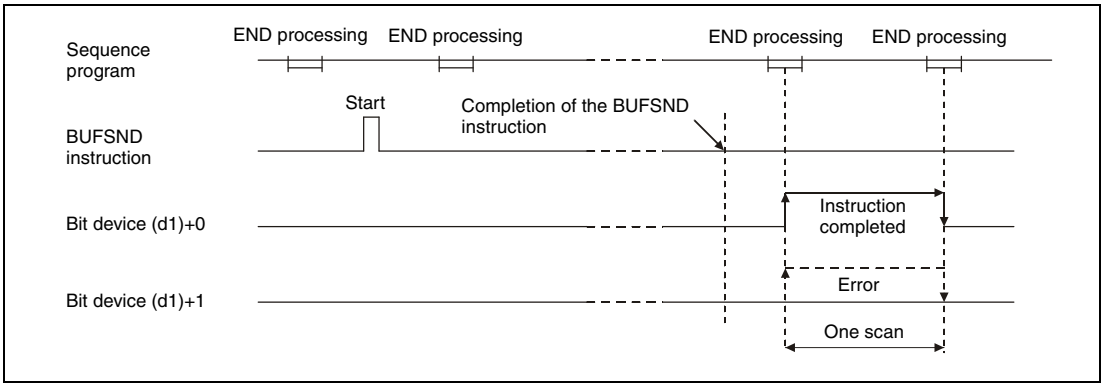
Data which is to be send through fixed buffer communication to an external device connected to an ETHERNET interface module is send to this module by the BUFSND instruction in advance. The data is stored in the PLC CPU from the device designated by (s3)+1 onward:



Whether the execution of the BUFSND instruction has been finished can be checked with the devices (d)+0 and (d)+1:

- The bit device (d)+0 turns ON at the END processing of the scan in which the BUFSND instruction has been completed and turns OFF at the next END processing.
- The bit device (d)+1 indicates an error during execution of the BUFSND instruction. When the instruction is completed normal, this device stays OFF. When an error occurs during execution of the BUFSND instruction, (d)+1 turns ON at the END processing of the scan in which the BUFSND instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the BUFSND instruction is being executed:



The BUFSND instruction is executed when the command for this instruction switches from off to on.

Operation Error

When the BUFRCV instruction is completed abnormally, the bit device (d1)+1 is set, and an error code is stored in (s2)+1. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFFH or less, you will find more information in chapter 13 of this manual.
- When the error code is C001H or higher, please refer to the user's manual of the ETHERNET interface module.

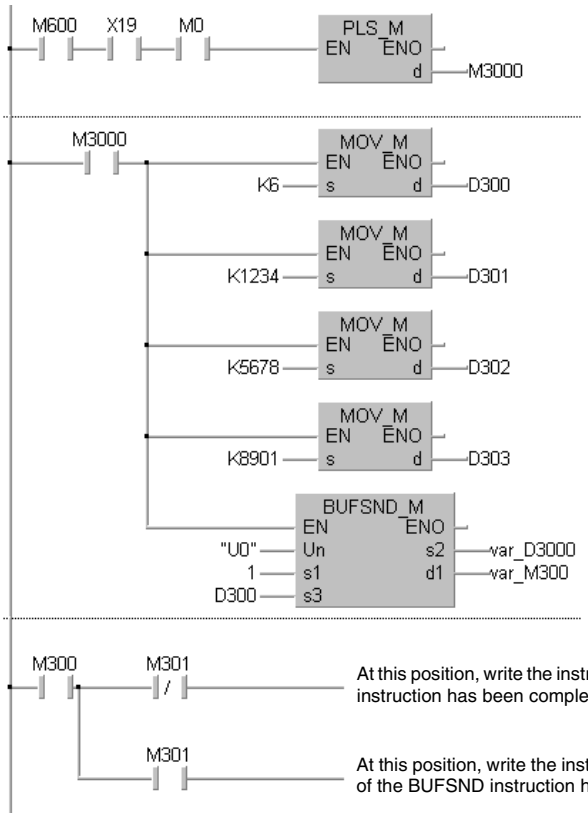
Program Example

BUFSND

The following program writes data to the fixed buffer for connection 1. The head I/O number of the ETHERNET module is X/Y0.

- IEC editors (On the next page the same program example is shown for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



IEC Instruction List

```
LD          M600
AND         X19
AND         M0
PLS_M       M3000
LD          M3000
MOV_M       K6, D300
MOV_M       K1234, D301
MOV_M       K5678, D302
MOV_M       K8901, D303
BUFSND_M    "U0", K1, D300, var_D3000, var_M300
```

TFor an explanation of the devices and instructions used please see the above ladder diagram.

```
LD          M300
ANDN        M301
```

At this position, write the instructions that should be executed when the BUFSND instruction has been completed normally.

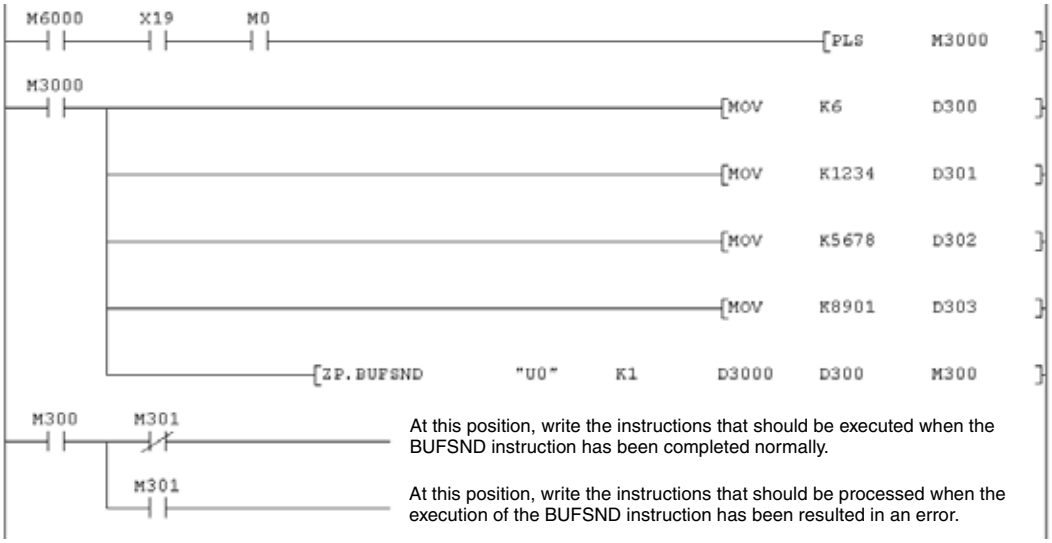
```
LD          M300
AND         M301
```

At this position, write the instructions that should be processed when the execution of the BUFSND instruction has been resulted in an error.

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



MELSEC Instruction List

```
MELSEC
LD      M6000
AND     X19
AND     M0
PLS     M3000
LD      M3000
MOV     K6      D300
MOV     K1234   D301
MOV     K5678   D302
MOV     K8901   D303
ZP.BUFSND "U0"  K1      D3000   D300    M300

MELSEC
LD      M300
ANI     M301
At this position, write the instructions that should be executed when the BUFSND instruction has been completed normally.

MELSEC
LD      M300
AND     M301
At this position, write the instructions that should be processed when the execution of the BUFSND instruction has been resulted in an error.
```

12.3.4 OPEN

CPU

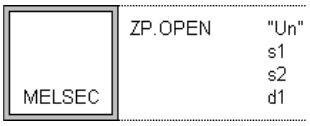
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

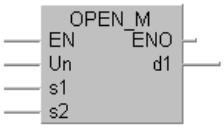
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	—	—
d1	●	●	●	—	—	—	—	—	—

GX IEC Developer

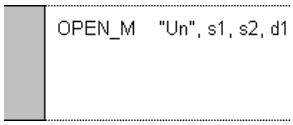
MELSEC Instruction List



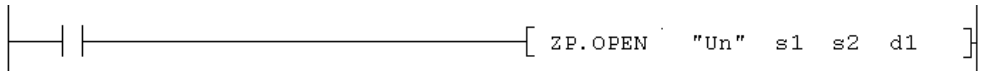
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Range	Contents is stored by	Data Type
„Un“	Head I/O address of the ETHERNET interface module (The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)	0 to FEH	User	BIN 16-bit
s1	Connection number	1 to 16		

Variables

<table>
<tr><th>Set Data</th><th colspan="3">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="5">s2</td><td colspan="5">Head number of the devices where control data for the execution of this instruction is stored.</td><td rowspan="5">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s2)+0</td><td>Source of the parameter settings</td><td>Designate which settings are used to open the connection:
• 0000H: The connection will be opened with the settings made in GX (IEC) Developer.
• 8000H: The connection will be opened with the settings stored in the devices (s2)+2 to (s2)+9.</td><td>0000H
or
8000H</td><td>User</td></tr>
<tr><td>(s2)+1</td><td>Execution result of the instruction</td><td>Indicates whether an error has occured during execution of the instruction:
0000H: No error
Any value other than 0000H: An error has occured. The stored value is an error code which is explained either in the user's manual of the ETHERNET interface module or in chapter 13 of this manual.</td><td>—</td><td>System</td></tr>
<tr><td>(s2)+2</td><td>Application setting area</td><td>The bits of this device are used to make settings for the connection:
• Bit 0: Usage of fixed buffers
0: The buffer is used for transmission, fixed buffer communication is not executed
1: The buffer is used for receiving
• Bit 1: Destination existence confirmation
0: No confirm
1: Confirm
• Bit 7: Pairing open setting
0: No pairs
1: Pairs
• Bit 8: Communication method (protocol)
0: TCP/IP
1: UDP/IP
• Bit 9: Fixed buffer communication
0: With procedure
1: Without procedure
• Bit 13: Issue an interrupt when receiving fixed buffer
0: No interrupt
1: An Interrupt is issued
• Bits 14 and 15: Active or passive opening
Bit 15/14 = 00: Active open or UDP/IP
Bit 15/14 = 10: Unpassive open
Bit 15/14 = 11: Full passive open</td><td>Please see the description on the left.</td><td>User</td></tr>
</table>

Variables

<table>
<tr><th>Set Data</th><th colspan="3">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="5">s2</td><td>Set Data</td><td>Meaning</td><td>Description</td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td></tr>
<tr><td>(s2)+3</td><td>Port No. of the ETHERNET module</td><td>Designate the port No. of the ETHERNET interface module.</td><td>408H to 1388H
138BH to FFFEH</td><td rowspan="4">User</td><td rowspan="4">BIN 16-bit</td></tr>
<tr><td>(s2)+4
(s2)+5</td><td>Destination IP address</td><td>IP address of the external device to communicate with.
When the IP address FFFFFFFFH is set, data is exchanged with simultaneous broadcast.</td><td>1H to FFFFFFFFH</td></tr>
<tr><td>(s2)+6</td><td>Destination Port No.</td><td>Port No. of the external device to communicate with.
(FFFFH = Simultaneous broadcast)</td><td>401H to FFFFH</td></tr>
<tr><td>(s2)+7 to (s2)+9</td><td>Destination ETHERNET address</td><td>When the external device supports the ARP function set either 000000000000H or FFFFFFFFFFFFH.When the external device support does not support the ARP function set the destination ETHERNET address.</td><td>Please see the description on the left.</td></tr>
<tr><td rowspan="4">d1</td><td colspan="6">Bit device which is set for one scan after completion of the OPEN instruction. (d)+1 indicates an abnormal completion of the instruction.</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d1)+0</td><td>Instruction completed</td><td>Indicates the completion of the OPEN instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td><td rowspan="2">Bit</td></tr>
<tr><td>(d1)+1</td><td>Instruction completed with error</td><td>Indicates the abnormal completion of the OPEN instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions

Opening of a connection

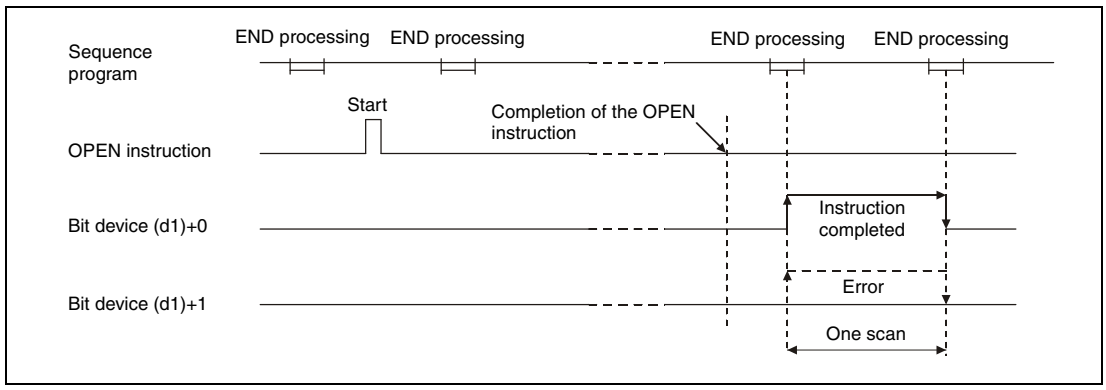
OPEN Open connection

This instruction performs the open processing for a connection specified by s1 for the module designated by Un.

Whether the execution of the OPEN instruction has been finished can be checked with the devices (d1)+0 and (d1)+1:

- The bit device (d1)+0 turns ON at the END processing of the scan in which the OPEN instruction has been completed and turns OFF at the next END processing.
- The bit device (d1)+1 indicates an error during execution of the OPEN instruction. When the instruction has been completed normal, this device stays OFF. When an error occurs during execution of the OPEN instruction, (d1)+1 turns ON at the END processing of the scan in which the OPEN instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the OPEN instruction is being executed:



The OPEN instruction is executed when the command for this instruction switches from off to on.

NOTE

Never execute the open/close processing using input/output signals and the OPEN or CLOSE dedicated instructions simultaneously for the same connection. It will result in malfunctions.

Operation Error

When an error occurs during the processing of the OPEN instruction, the bit device (d1)+1 is set, and an error code is stored in (s2)+1. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFFH or less, you will find more information in chapter 13 of this manual.
- When the error code is C001H or higher, please refer to the user's manual of the ETHERNET interface module.

Program Example

OPEN

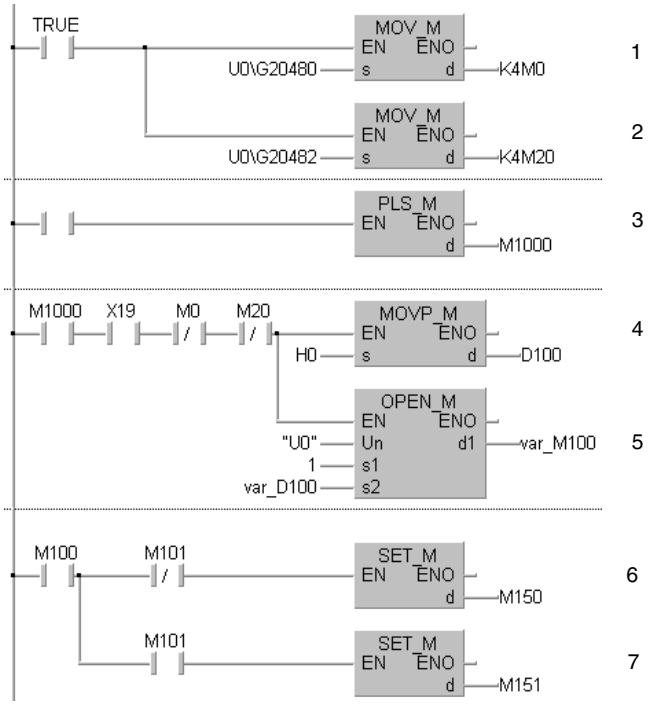
The following program active opens the connection number 1 for TCP/IP communication. The head I/O address of the is X/Y0.

NOTE

For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

- Ladder Diagram (GX IEC Developer)
  For the following example it is neccesary to set the parameters with the GX (IEC) Developer in advance. Another example where the settings are made with the OPEN instruction is shown on the next page.

Setting of the parameters is done using the GX Works2 or the GX IEC Developer



[1] Reading of the connection status (M0 = 1: Opening of connection 1 has been completed)

[2] Reading of the open request (M20 = 1: Opening of connection 1 is requested)

[3] The signal to open the connection is converted to a pulse.

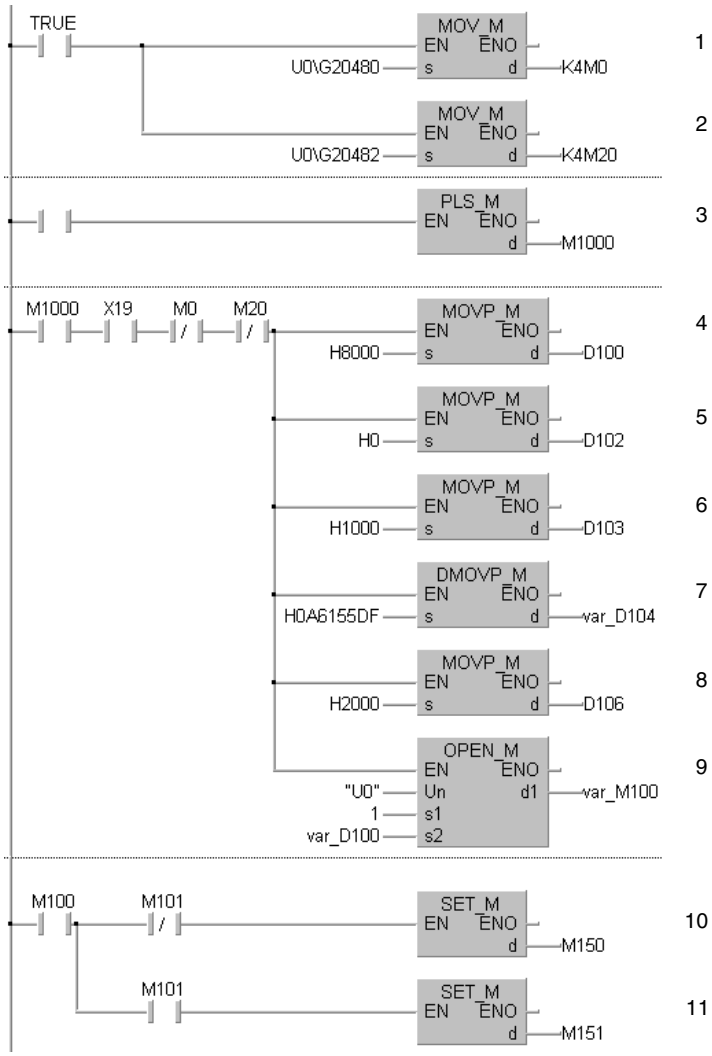
[4] The source for the parameters is set (0000H = External setting).

[5] Opening of connection 1

[6] M150 is set when the opening of the connection has been completed without an error.

[7] M151 is set when an error has occured during the opening of the connection.

The settings for the connection are stored in the devices designated by (s2)+2 to (s2)+9



[1] Reading of the connection status (M0 = 1: Opening of connection 1 has been completed)

[2] Reading of the open request (M20 = 1: Opening of connection 1 is requested)

[3] The signal to open the connection is converted to a pulse.

[4] The source for the parameters is set (8000H = Parameters are stored in (s2)+2 to (s2)+9))

[5] The application setting is stored in (s2)+2.

[6] The port No. of the ETHERNET module is written to (s2)+3.

[7] The IP address (10.97.85.223) of the external device is stored in (s2)+4 and (s2)+5.

[8] In (s2)+6 the port No. of the external device is stored.

[9] Opening of connection 1

[10] M150 is set when the opening of the connection has been completed without an error.

[11] M151 is set when an error has occured during the opening of the connection.

- IEC Instruction List

Setting of the parameters is done using the GX Works2 or the GX IEC Developer

```
LD      TRUE
MOV_M   U0\G20480, K4M0                                  1
MOV_M   U0\G20482, K4M20                                 2
LD      M123
PLS_M   M1000                                            3
LD      M1000
AND     X19
ANDN    M0
ANDN    M20
MOVP_M  H0,      D100                                    4
OPEN_M  "U0", K1, var_D100,      var_M100                9

LD      M100
ANDN    M101
SET_M   M150                                             10
LD      M100
AND     M101
SET_M   M151                                             11
```

The settings for the connection are stored in the devices designated by (s2)+2 to (s2)+9

```
LD      TRUE
MOV_M   U0\G20480, K4M0                                  1
MOV_M   U0\G20482, K4M20                                 2
LD      M123
PLS_M   M1000                                            3
LD      M1000
AND     X19
ANDN    M0
ANDN    M20
MOVP_M  H8000,   D100                                    4
MOVP_M  H0,      D102                                    5
MOVP_M  H1000,   D103                                    6
DMOVP_M H0A8155DF,       var_D104                        7
MOVP_M  H2000,   D106                                    8
OPEN_M  "U0", K1, var_D100,      var_M100                9

LD      M100
ANDN    M101
SET_M   M150                                             10
LD      M100
AND     M101
SET_M   M151                                             11
```

[1] Reading of the connection status (M0 = 1: Opening of connection 1 has been completed)

[2] Reading of the open request (M20 = 1: Opening of connection 1 is requested)

[3] The signal to open the connection is converted to a pulse.

[4] The source for the parameters is set (0000H = External, 8000H = Devices (s2)+2 to(s2)+9))

[5] The application setting is stored in (s2)+2.

[6] The port No. of the ETHERNET module is written to (s2)+3

[7] The IP address (10.97.85.223) of the external device is stored in (s2)+4 and (s2)+5.

[8] In (s2)+6 the port No. of the external device is stored.

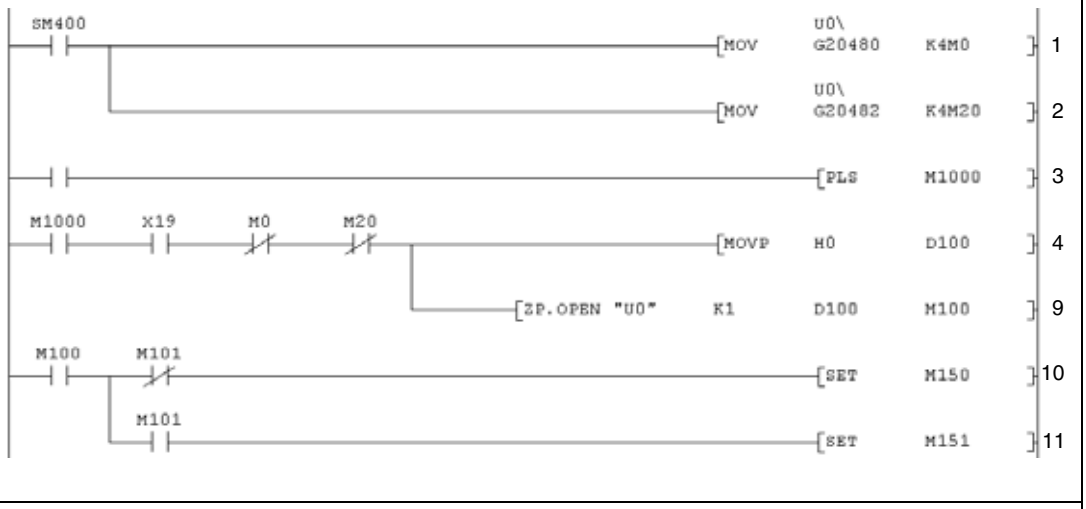
[9] Opening of connection 1

[10] M150 is set when the opening of the connection has been completed without an error.

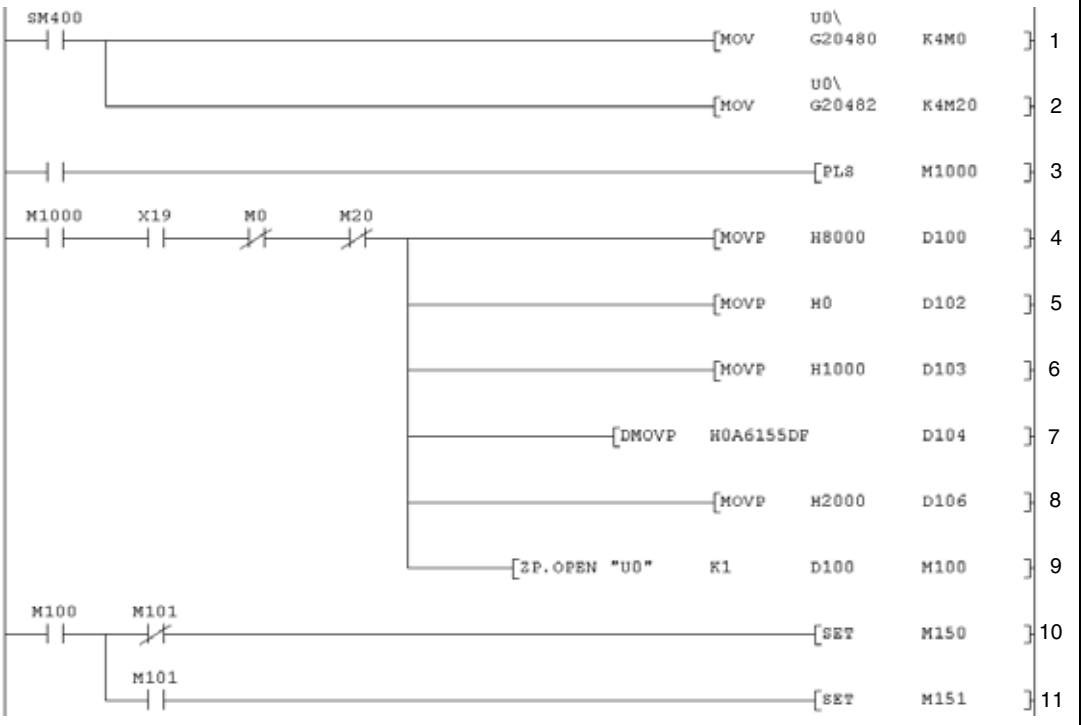
[11] M151 is set when an error has occured during the opening of the connection.

● Ladder Diagram (GX Works2)

Setting of the parameters is done using the GX Works2 or the GX IEC Developer



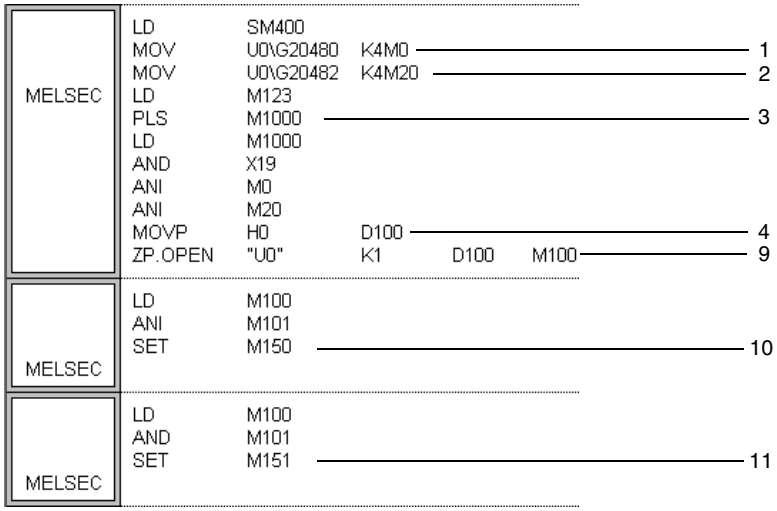
The settings for the connection are stored in the devices designated by (s2)+2 to (s2)+9



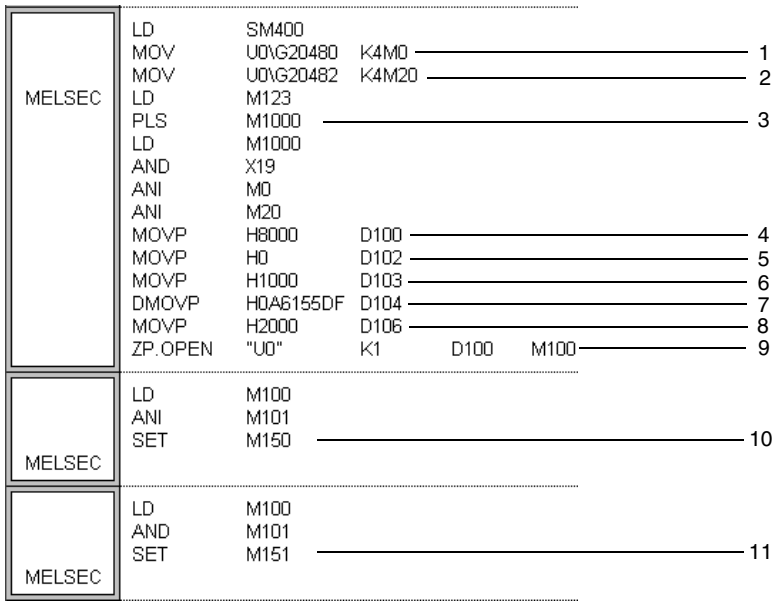
[1] Reading of the connection status (M0 = 1: Opening of connection 1 has been completed)
[2] Reading of the open request (M20 = 1: Opening of connection 1 is requested)
[3] The signal to open the connection is converted to a pulse.
[4] The source for the parameters is set (0000H = External, 8000H = Devices (s2)+2 to(s2)+9))
[5] The application setting is stored in (s2)+2.
[6] The port No. of the ETHERNET module is written to (s2)+3
[7] The IP address (10.97.85.223) of the external device is stored in (s2)+4 and (s2)+5.
[8] In (s2)+6 the port No. of the external device is stored.
[9] Opening of connection 1
[10] M150 is set when the opening of the connection has been completed without an error.
[11] M151 is set when an error has occured during the opening of the connection.

- MELSEC Instruction List

Setting of the parameters is done using the GX Works2 or the GX IEC Developer



The settings for the connection are stored in the devices designated by (s2)+2 to (s2)+9



[1] Reading of the connection status (M0 = 1: Opening of connection 1 has been completed)

[2] Reading of the open request (M20 = 1: Opening of connection 1 is requested)

[3] The signal to open the connection is converted to a pulse.

[4] The source for the parameters is set (0000H = External, 8000H = Devices (s2)+2 to(s2)+9))

[5] The application setting is stored in (s2)+2.

[6] The port No. of the ETHERNET module is written to (s2)+3

[7] The IP address (10.97.85.223) of the external device is stored in (s2)+4 and (s2)+5.

[8] In (s2)+6 the port No. of the external device is stored.

[9] Opening of connection 1

[10] M150 is set when the opening of the connection has been completed without an error.

[11] M151 is set when an error has occured during the opening of the connection.

12.3.5 CLOSE

CPU

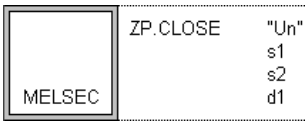
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

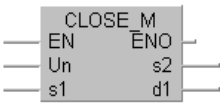
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	●	—
s2	—	●	●	—	—	—	—	—	—
d1	●	●	●	—	—	—	—	—	—

GX IEC Developer

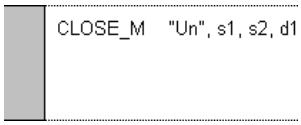
MELSEC Instruction List



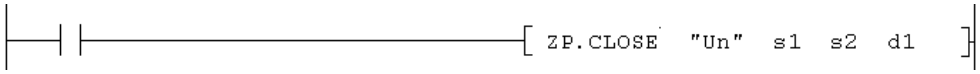
Ladder Diagram



IEC Instruction List



GX Works2



Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>„Un“</td><td colspan="4">Head I/O address of the ETHERNET interface module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)</td><td>0 to FE_H</td><td rowspan="2">User</td><td rowspan="2">BIN 16-bit</td></tr>
<tr><td>s1</td><td colspan="4">Number of the connection</td><td>1 to 16</td></tr>
<tr><td rowspan="4">s2</td><td colspan="6">Head number of the devices where control data for the execution of this instruction is stored</td><td rowspan="4">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td colspan="2">Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s2)+0</td><td>System area</td><td colspan="2">Used by the system</td><td rowspan="2">—</td><td rowspan="2">System</td></tr>
<tr><td>(s2)+1</td><td>Execution result of the instruction</td><td colspan="2">Indicates whether an error has occured during execution of the instruction:
0000_H: No error
Any value other than 0000_H: An error has occured. The stored value is an error code which is explained either in the user’s manual of the ETHERNET interface module or in chapter 13 of this manual.</td></tr>
<tr><td rowspan="4">d1</td><td colspan="6">Bit device which is set for one scan after completion of the CLOSE instruction. (d)+1 indicates an abnormal completion of the instruction</td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td colspan="2">Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d1)+0</td><td>Instruction completed</td><td colspan="2">Indicates the completion of the CLOSE instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d1)+1</td><td>Instruction completed with error</td><td colspan="2">Indicates that an error has occured during the processing of the CLOSE instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions

Closing of a connection

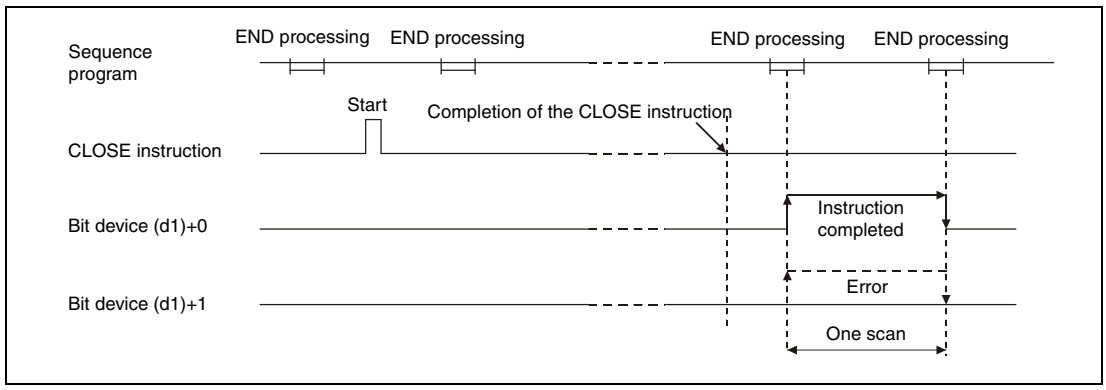
CLOSE Close connection

This instruction closes the connection specified by s1 for the module designated by Un (disconnecting connections).

Whether the execution of the CLOSE instruction has been finished can be checked with the devices (d1)+0 and (d1)+1:

- The bit device (d1)+0 turns ON at the END processing of the scan in which the CLOSE instruction has been completed and turns OFF at the next END processing.
- The bit device (d1)+1 indicates an error during execution of the CLOSE instruction. When the instruction has been completed normal, this device stays OFF. When an error occurs during execution of the CLOSE instruction, (d1)+1 turns ON at the END processing of the scan in which the CLOSE instruction has been completed and turns OFF at the next END processing.

The timing for the CLOSE instruction is shown in the following figure:



The CLOSE instruction is executed when the command for this instruction switches from off to on.

NOTE

Never execute the open/close processing using input/output signals and the OPEN or CLOSE dedicated instructions simultaneously for the same connection. It will result in malfunctions.

Operation Error

When an error occurs during the processing of the CLOSE instruction, the bit device (d1)+1 is set, and an error code is stored in (s2)+1. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFFH or less, you will find more information in chapter 13 of this manual.
- When the error code is C001H or higher, please refer to the user's manual of the ETHERNET interface module.

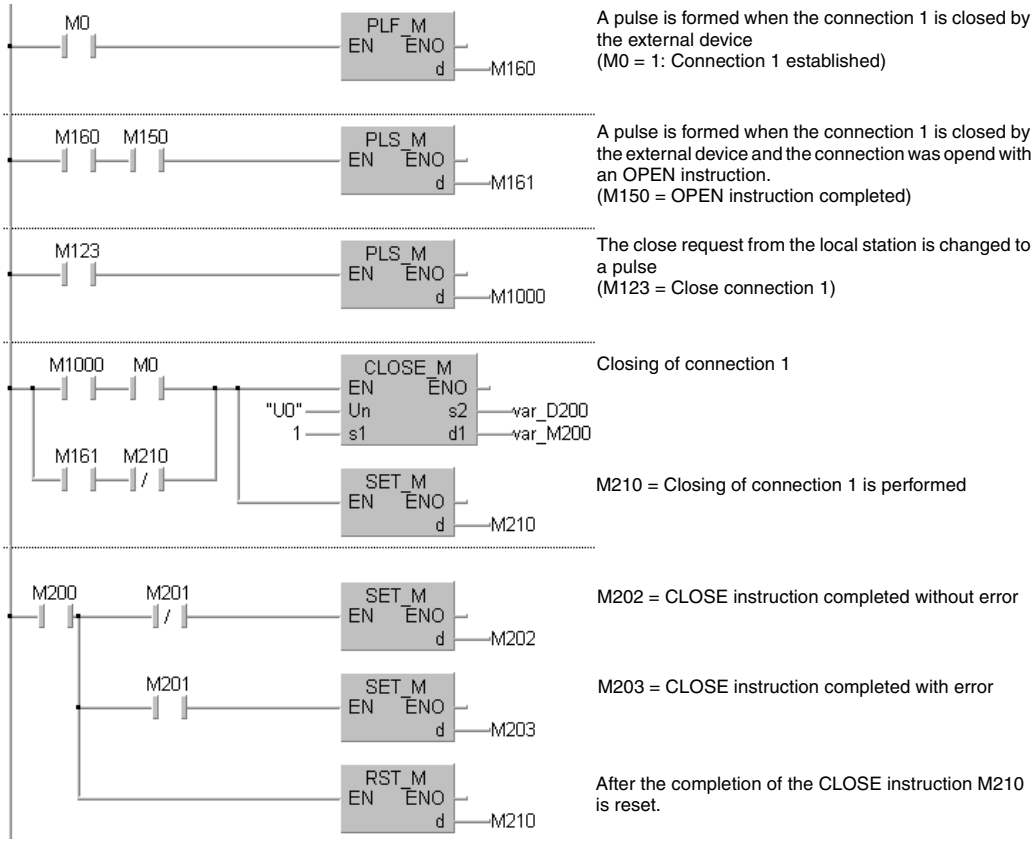
Program Example

CLOSE

The following program closes the connection number 1 of the ETHERNET module with the head I/O address X/Y0.

- IEC editors (On the next page the same program example is shown for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



IEC Instruction List

```
LD      M0
PLF_M   M160
LD      M160
AND     M150
PLS_M   M161
LD      M123
PLS_M   M1000
LD      M1000
AND     M0
OR(
        M161
ANDN    M210
)
CLOSE_M "U0", K1, var_D200, var_M200
SET_M   M210
```

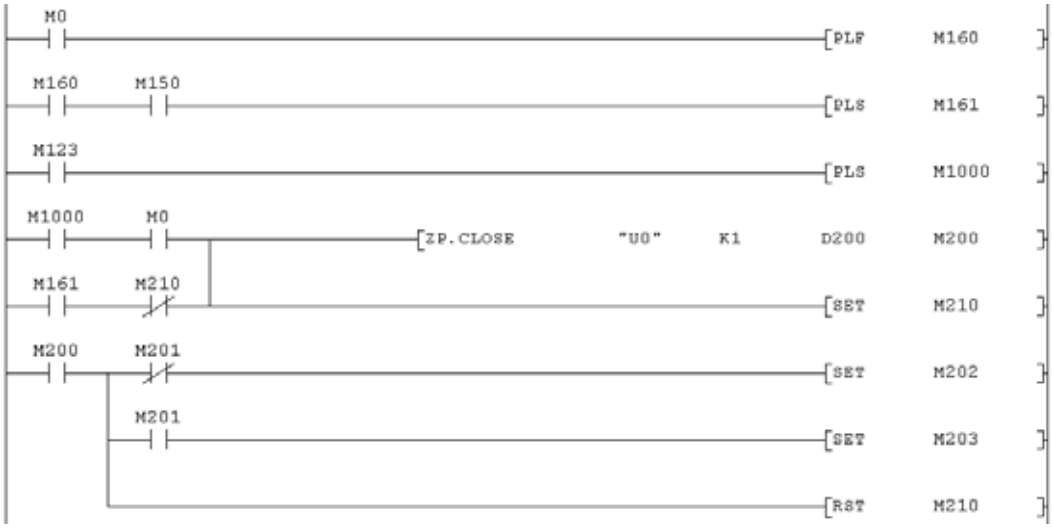
```
LD      M201
ANDN    M201
SET_M   M202
LD      M201
AND     M202
SET_M   M203
LD      M200
RST_M   M210
```

For an explanation of the devices and instructions used please see the above ladder diagram.

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



MELSEC Instruction List

MELSEC	LD	M0			
	PLF	M160			
	LD	M160			
	AND	M150			
	PLS	M161			
	LD	M123			
	PLS	M1000			
	LD	M1000			
	AND	M0			
	LD	M161			
	ANI	M210			
	ORB				
	ZP.CLOSE	"U0"	K1	D200	M200
	SET	M210			
MELSEC	LD	M200			
	MPS				
	ANI	M201			
	SET	M202			
	MRD				
	AND	M201			
	SET	M203			
	MPP				
	RST	M210			

12.3.6 ERRCLR

CPU

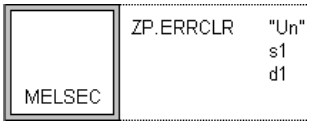
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

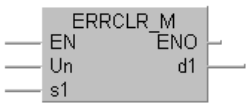
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
d1	●	●	●	—	—	—	—	—	—

GX IEC Developer

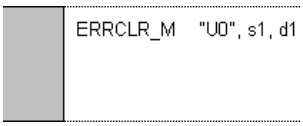
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>„Un“</td><td colspan="4">Head I/O address of the ETHERNET interface module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)</td><td>0 to FE_H</td><td>User</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="8">s1</td><td colspan="4">Head number of the devices where control data for the execution of this instruction is stored</td><td></td><td></td><td rowspan="8">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s1)+0</td><td>System area</td><td>Used by the system</td><td rowspan="2">—</td><td rowspan="2">System</td></tr>
<tr><td>(s1)+1</td><td>Execution result of the instruction</td><td>Indicates whether an error has occured during execution of the instruction:
0000_H: No error
Any value other than 0000_H: An error has occured. The stored value is an error code which is explained either in the user’s manual of the ETHERNET interface module or in chapter 13 of this manual.</td></tr>
<tr><td>(s1)+2</td><td>Error information to be cleared</td><td>Depending on the entered value an error code stored in the buffer memory is cleared and the „ERR.“ LED of the ETHERNET module is switched off.
Please refer to the description on the next page.</td><td>0000_H
0001_H
to
0016_H
0100_H
0101_H
0102_H
0103_H
FFFF_H</td><td rowspan="2">User</td></tr>
<tr><td>(s1)+3</td><td>Function</td><td>Choose between clearing of an error code and switching off of the „ERR.“ LED.
Please refer to the description on the next page.</td><td>0000_H
or
FFFF_H</td></tr>
<tr><td>(s1)+4
to
(s1)+7</td><td>System area</td><td>Used by the system</td><td>—</td><td>System</td></tr>
<tr><td rowspan="4">d1</td><td colspan="4">Bit device which is set for one scan after completion of the ERRCLR instruction. (d)+1 indicates an abnormal completion of the instruction.</td><td></td><td></td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d1)+0</td><td>Instruction completed</td><td>Indicates the completion of the ERRCLR instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d1)+1</td><td>Instruction completed with error</td><td>Indicates that an error has occured during the processing of the ERRCLR instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions **Clearing of errorcode and turning off the „ERR.“ LED**

ERRCLR Clearing operation

The ERRCLR instruction clears an error code stored in the buffer memory of the ETHERNET interface module. When the „ERR.“ LED at the front side of the module is lit, this indicator is turned off after processing of the ERRCLR instruction as well. This instruction also clears the areas in the buffer memory where the communication status is stored.

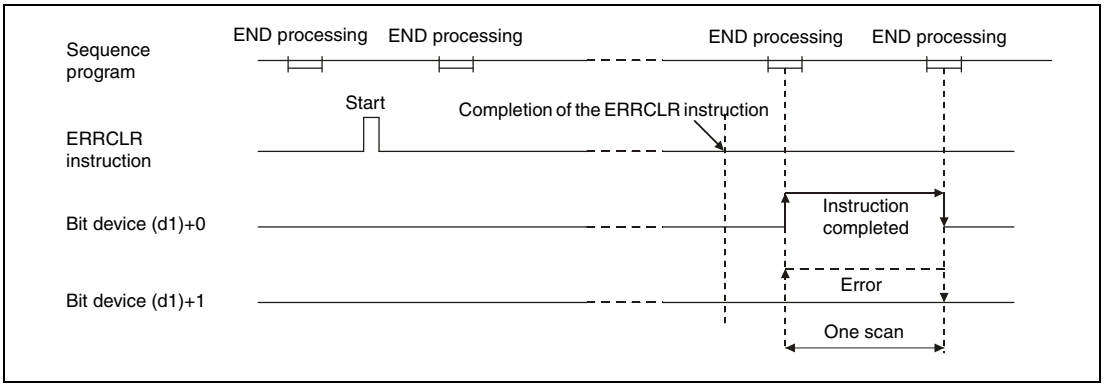
Which area of the buffer memory is cleared depends on the contents of the devices designated by (s1)+2 and (s1)+3:

Error or communication status area		Contents of		Action that will be performed
		(s1)+2	(s2)+3	
Initial error		0000H	0000H	• The buffer memory address 69H is cleared. • The „ERR.“ LED is switched off.
Error when opening an connection		0001H to 0016H (Number of the connection)		• The buffer memory address where an errorcode for the faulty connection is stored is cleared (7CH, 86H...). • The „ERR.“ LED is switched off.
Error log		0100H	FFFFH	• The error log (addresses E3H to 174H) is cleared.
Communication status	Status of the protocols	0101H		• The buffer memory addresses from 178H to 1FFH are cleared.
	E-mail receive status	0102H		• The buffer memory addresses from 5871H to 5B38H are cleared.
	E-mail send status	0103H		• The buffer memory addresses from 5B39H to 5CA0H are cleared.
All stored error codes or communication status areas		FFFFH		• All above mentioned buffer memory areas are cleared. • The „ERR.“ LED is switched off.

Whether the execution of the ERRCLR instruction has been finished can be checked with the devices (d1)+0 and (d1)+1:

- The bit device (d1)+0 turns ON at the END processing of the scan in which the ERRCLR instruction has been completed and turns OFF at the next END processing.
- The bit device (d1)+1 indicates an error during execution of the ERRCLR instruction. When the instruction has been completed normal, this device stays OFF. When an error occurs during execution of the ERRCLR instruction, (d1)+1 turns ON at the END processing of the scan in which the ERRCLR instruction has been completed and turns OFF at the next END processing.

The timing when executing the ERRCLR instruction is shown below:



Operation Error When an error occurs during the processing of the ERRCLR instruction, the bit device (d1)+1 is set, and an error code is stored in (s1)+1. For more information about the error codes please

refer to the following manuals:

- When the error code is 4FFFH or less, you will find more information in chapter 13 of this manual.
- When the error code is C001H or higher, please refer to the user's manual of the ETHERNET interface module.

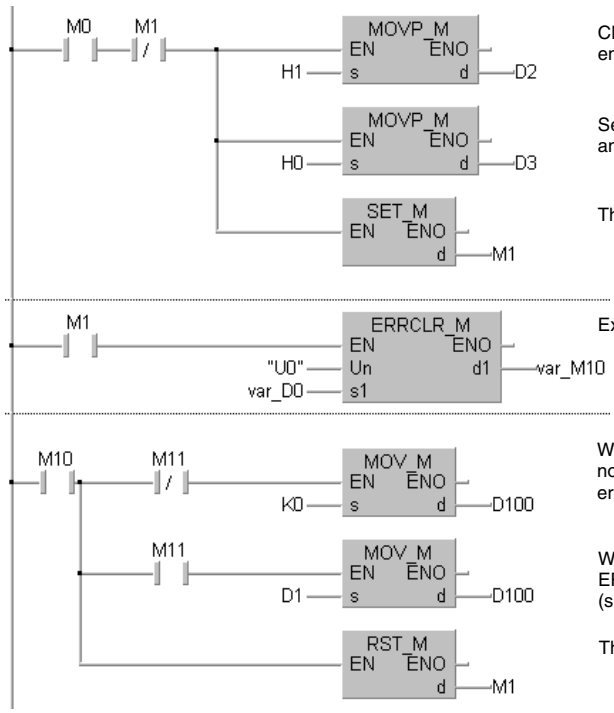
Program Example

ERRCLR

The following program is used to clear the error code issued for connection 1. The ETHERNET module occupies the inputs and outputs from X/Y0.

- IEC editors (On the next page this program example is shown for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



Clearing of the error code for connection 1 (1H is entered in (s1)+2)

Selection of the function: Clearing of the error code and turning off the „ERR." LED (Moving of 0H to (s1)+3)

The signal to start the ERRCLR instruction is set.

Execution of the ERRCLR instruction

When the ERRCLR instruction has been completed normally the data register D100 is cleared. In D100 the error code of the ERRCLR instruction is stored.

When an error has occured during execution of the ERRCLR instruction the error code ist moved from (s1)+1 (register D1) to D100.

The start signal for the ERRCLR instruction is reset.

IEC Instruction List

```
LD          M0
ANDN        M1
MOVP_M      H1,     D2
MOVP_M      H0,     D3
SET_M       M1
LD          M1
ERRCLR_M    "U0",   var_D0,   var_M10
```

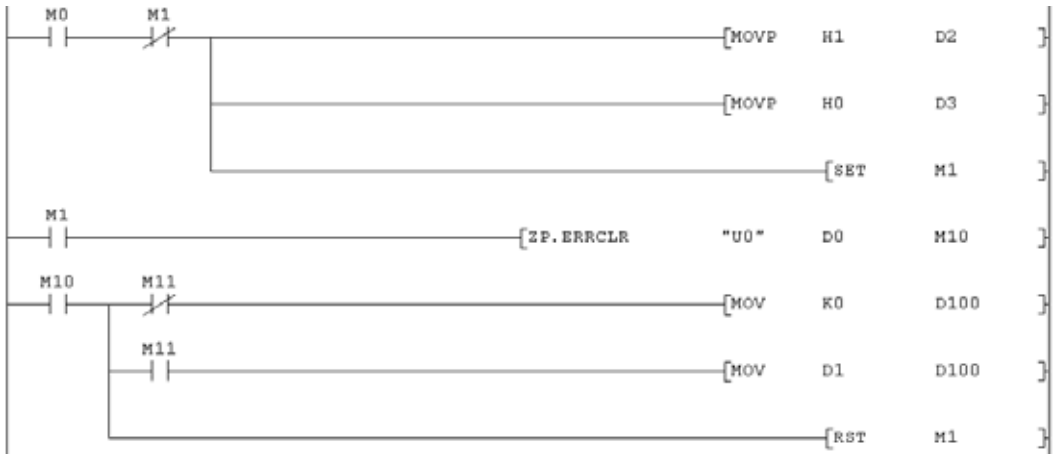
```
LD          M10
ANDN        M11
MOV_M       K0,     D100
LD          M10
AND         M11
MOV_M       D1,     D100
LD          M10
RST_M       M1
```

For an explanation of the devices and instructions used please see the above ladder diagram.

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



MELSEC Instruction List

```
MELSEC   LD       M0
         ANI      M1
         MOVP     H1      D2
         MOVP     H0      D3
         SET      M1
         LD       M1
         ZP.ERRCLR "U0"   D0      M10

MELSEC   LD       M10
         MPS
         ANI      M11
         MOV      K0              D100
         MRD
         AND      M11
         MOV      D1              D100
         MPP
         RST      M1
```

12.3.7 ERRRD

CPU

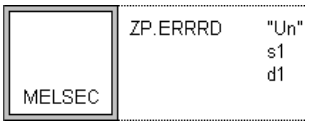
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

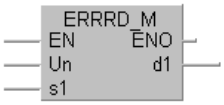
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
d1	●	●	●	—	—	—	—	—	—

GX IEC Developer

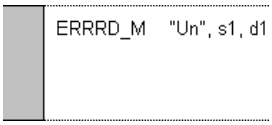
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>„Un“</td><td colspan="4">Head I/O address of the ETHERNET interface module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)</td><td>0 to FE_H</td><td>User</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="8">s1</td><td colspan="5">Head number of the devices where control data for the execution of this instruction is stored</td><td rowspan="8">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s1)+0</td><td>System are</td><td>Used by the system</td><td rowspan="2">—</td><td rowspan="2">System</td></tr>
<tr><td>(s1)+1</td><td>Execution result of the instruction</td><td>Indicates whether an error has occured during execution of the instruction:
0000_H: No error
Any value other than 0000_H: An error has occured. The stored value is an error code which is explained either in the user’s manual of the ETHERNET interface module or in chapter 13 of this manual.</td></tr>
<tr><td>(s1)+2</td><td>Error code to be read</td><td>Depending on the entered value an error code stored in the buffer memory is read:
• 0000_H: Initial error code which is entered in the buffer memory address 69_H.
• 0001_H to 0016_H: Error code for the connection with this number (Buffer memory address 7C_H, 86_H ...)</td><td>0000_H
0001_H
to
0016_H</td><td rowspan="2">User</td></tr>
<tr><td>(s1)+3</td><td>Function</td><td>Reading of the last issued error code</td><td>0000_H</td></tr>
<tr><td>(s1)+4</td><td>Read error code</td><td>Stores the error code read from the ETHERNET module
0000_H = No error
Other than 0000_H: error code</td><td>—</td><td>System</td></tr>
<tr><td>(s1)+5
to
(s1)+7</td><td>System area</td><td>Used by the system</td><td>—</td><td>System</td></tr>
<tr><td rowspan="4">d1</td><td colspan="5">Bit device which is set for one scan after completion of the ERRRD instruction.
(d)+1 indicates an abnormal completion of the instruction.</td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d1)+0</td><td>Instruction completed</td><td>Indicates the completion of the ERRRD instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d1)+1</td><td>Instruction completed with error</td><td>Indicates that an error has occured during the processing of the ERRRD instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions

Reading of an error code from an ETHERNET module

ERRRD Read error code

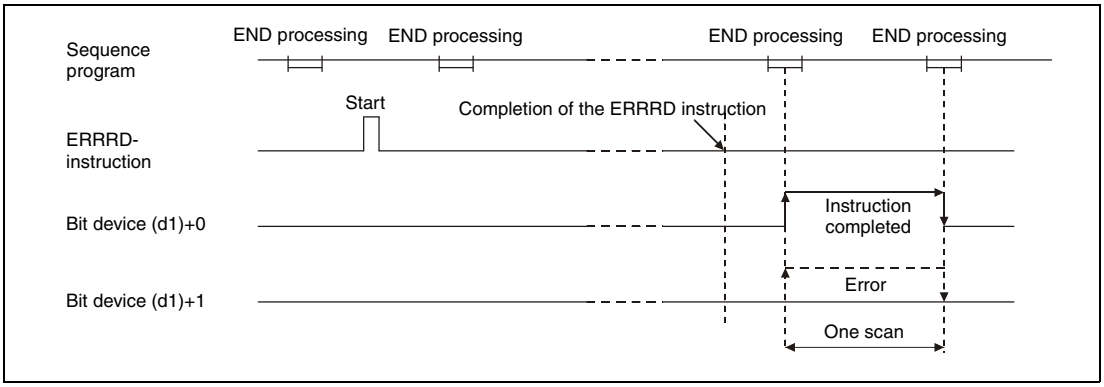
This instruction reads error code which is stored in the buffer memory of the ETHERNET interface module with the head I/O number designated by „Un“.

The device designated by (s1)+2 stores information about the buffer memory address to read from.

Whether the execution of the ERRRD instruction has been finished can be checked with the devices (d1)+0 and (d1)+1:

- The bit device (d1)+0 turns ON at the END processing of the scan in which the ERRRD instruction has been completed and turns OFF at the next END processing.
- The bit device (d1)+1 indicates an error during execution of the ERRRD instruction. When the instruction has been completed normal, this device stays OFF. When an error occurs during execution of the ERRRD instruction, (d1)+1 turns ON at the END processing of the scan in which the ERRRD instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the ERRRD instruction is being executed:



Operation Error

When an error occurs during the processing of the ERRRD instruction, the bit device (d1)+1 is set, and an error code is stored in (s1)+1. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFF$_H$ or less, you will find more information in chapter 13 of this manual.
- When the error code is C001$_H$ or higher, please refer to the user's manual of the ETHERNET interface module.

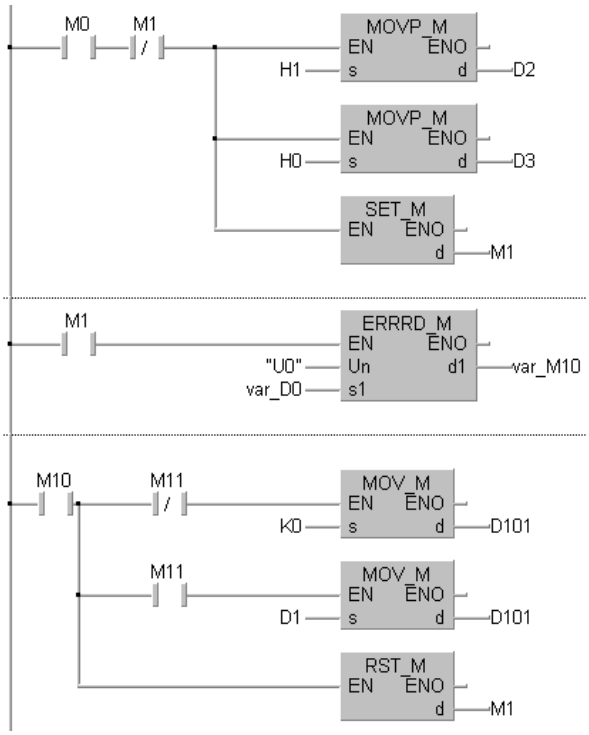
Program Example

ERRRD

The following program reads the error code which is issued if the opening of connection 1 has failed. The ETHERNET module has the head I/O address X/Y0.

- IEC editors (On the next page this program example is shown for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



Reading of the error code for connection 1 (1H is moved to (s1)+2)

Setting of the function: Reading of the last error code (0H is stored in (s1)+3)

The start signal for the ERRRD instruction is set.

The ERRRD instruction is executed when M1 changes from off to on.

When the ERRRD instruction has been completed normally the data register D101 is cleared. In D101 the error code of the ERRCLR instruction is stored.

When an error has occured during execution of the ERRCLR instruction the error code ist moved from (s1)+1 (register D1) to D101.

The start signal for the ERRRD instruction is reset.

IEC Instruction List

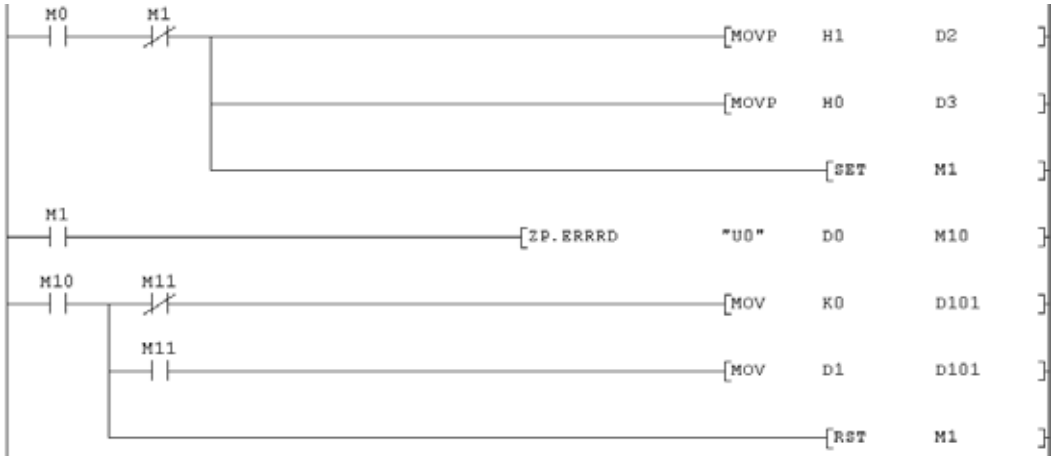
```
LD          M0
ANDN        M1
MOVP_M      H1,     D2
MOVP_M      H0,     D3
SET_M       M1
LD          M1
ERRRD_M     "U0",   var_D0,   var_M10

LD          M10
ANDN        M11
MOV_M       K0,     D101
LD          M10
AND         M11
MOV_M       D1,     D101
LD          M10
RST_M       M1
```

For an explanation of the devices and instructions used please see the above ladder diagram.

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



MELSEC Instruction List

MELSEC	LD	M0		
	ANI	M1		
	MOVP	H1	D2	
	MOVP	H0	D3	
	SET	M1		
	LD	M1		
	ZP.ERRRD	"U0"	D0	M10
MELSEC	LD	M10		
	MPS			
	ANI	M11		
	MOV	K0	D101	
	MRD			
	AND	M11		
	MOV	D1	D101	
	MPP			
	RST	M1		

12.3.8 UINI

CPU

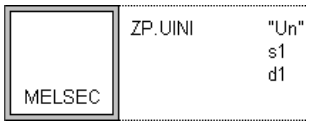
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

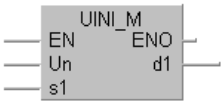
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
d1	●	●	●	—	—	—	—	—	—

GX IEC Developer

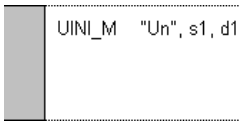
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

<table>
<tr><th>Set Data</th><th colspan="3">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>„Un“</td><td colspan="3">Head I/O address of the ETHERNET interface module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as „U10“)</td><td>0 to FE_H</td><td>User</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="8">s1</td><td colspan="5">Head number of the devices where control data for the execution of this instruction is stored</td><td rowspan="8">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s1)+0</td><td>System area</td><td>Used by the system</td><td rowspan="2">—</td><td rowspan="2">System</td></tr>
<tr><td>(s1)+1</td><td>Execution result of the instruction</td><td>Indicates whether an error has occured during execution of the instruction:
0000_H: No error
Any value other than 0000_H: An error has occured. The stored value is an error code which is explained either in the user’s manual of the ETHERNET interface module or in chapter 13 of this manual.</td></tr>
<tr><td>(s1)+2</td><td>Target of change</td><td>The bits 0 and 1 of this word device are used to specify the parameters to be changed:
• Bit 0: Change of the IP address of the local station (The new address is entered in (s1)+3 and (s1)+4.)
0: The IP address is not changed
1: Change the IP address
• Bit 1: Change of the operation settings (Enter the new settings in (s1)+5.)
0: Settings are not changed
1: The Settings are changed
Make sure to set all other bits (b2 to b15) to „0“.</td><td>0000_H
to
0003_H</td><td rowspan="3">User</td></tr>
<tr><td>(s1)+3
(s1)+4</td><td>IP address of the locale station</td><td>New IP address of the local station</td><td>00000001_H
to
FFFFFFFE_H</td></tr>
<tr><td>(s1)+5</td><td>Operation settings</td><td>The bits of this word device specify the operation settings:
• Bit 1: Communication data code setting
0: Communication in binary code
1: Communication in ASCII code
• Bit 5: Send frame setting
0: ETHERNET frame
1: IEEE802.3 frame
• Bit 6: Enable/disable writing of program when the CPU is in RUN mode
0: Writing disabled
1: Writing enabled
• Bit 8: Initial time setting
0: Do not wait for open (Communication is impossible when the CPU is stopped.)
1: Always wait for open (Communication is possible when the CPU is stopped.)
All other bits of this device must be reset (to „0“).</td><td>—</td></tr>
</table>

Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="4">d1</td><td colspan="5">Bit device which is set for one scan after completion of the UINI instruction. (d)+1 indicates that an error has occured during execution of the instruction.</td><td rowspan="4">Bit</td></tr>
<tr><td></td><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td></td><td>(d1)+0</td><td>Instruction completed</td><td>Indicates the completion of the UINI instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td></td><td>(d1)+1</td><td>Instruction completed with error</td><td>Indicates that an error has occured during the processing of the UINI instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

NOTE

When performing re-initial processing of the ETHERNET module only, i.e., without changing the local station IP address and operation settings, the control data should be specified so that the value (0H) is stored in (s1)+2, the specification of target of change, before executing the UINI instruction.
The ETHERNET module clears external device address information that it has been maintaining and performs re-initial processing in order to allow data communication to restart. (The initial normal completion signal (X19) is on.)

Functions

Re-initial processing of an ETHERNET interface module

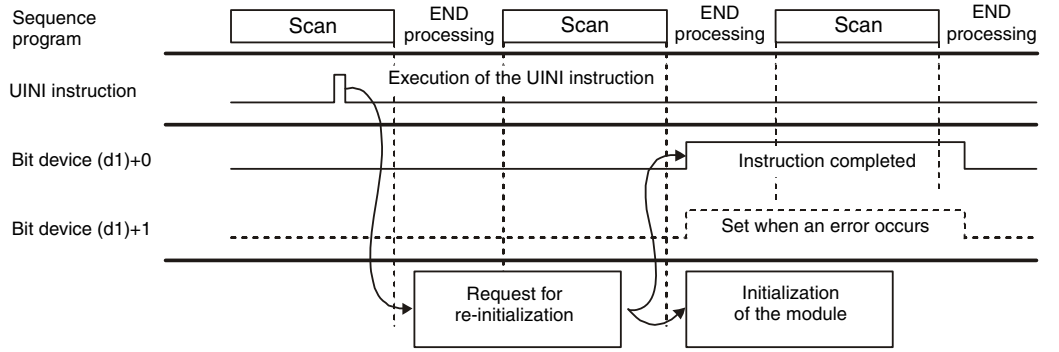
UINI Start re-initialization

The UINI instruction performs the re-initial processing of the ETHERNET module specified with Un.

Whether the execution of the UINI instruction has been finished can be checked with the devices (d1)+0 and (d1)+1:

- The bit device (d1)+0 turns ON at the END processing of the scan in which the UINI instruction has been completed and turns OFF at the next END processing.
- The bit device (d1)+1 indicates an error during execution of the UINI instruction. When the instruction has been completed normal, this device stays OFF. When an error occurs during execution of the UINI instruction, (d1)+1 turns ON at the END processing of the scan in which the UINI instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the UINI instruction is being executed:



NOTES

Please keep the following points in mind when reinitializing an ETHERNET module. (Failure to do so may cause errors in the data communication with the external devices.)

- Be sure to end all current data communication with external devices and close all connections before performing a re-initial process.

- Do not mix a re-initial processing done by writing directly into buffer memory, for instance by using a TO instruction, with a re-initial processing via UINI instruction.
Also, do not request another re-initial processing while an UINI instruction is already being executed.

- Be sure to reset external devices if the IP address of the ETHERNET module has been changed. (If an external device maintains the ETHERNET address of a device with which it communicates, the communication may not be continued after the IP address of the ETHERNET module has been changed.)

Operation Error

When an error occurs during the processing of the UINI instruction, the bit device (d1)+1 is set, and an error code is stored in (s1)+1. For more information about the error codes please refer to the following manuals:

- When the error code is 4FFFH or less, you will find more information in chapter 13 of this manual.
- When the error code is C001H or higher, please see the user's manual of the ETHERNET interface module.

Program Example

UINI

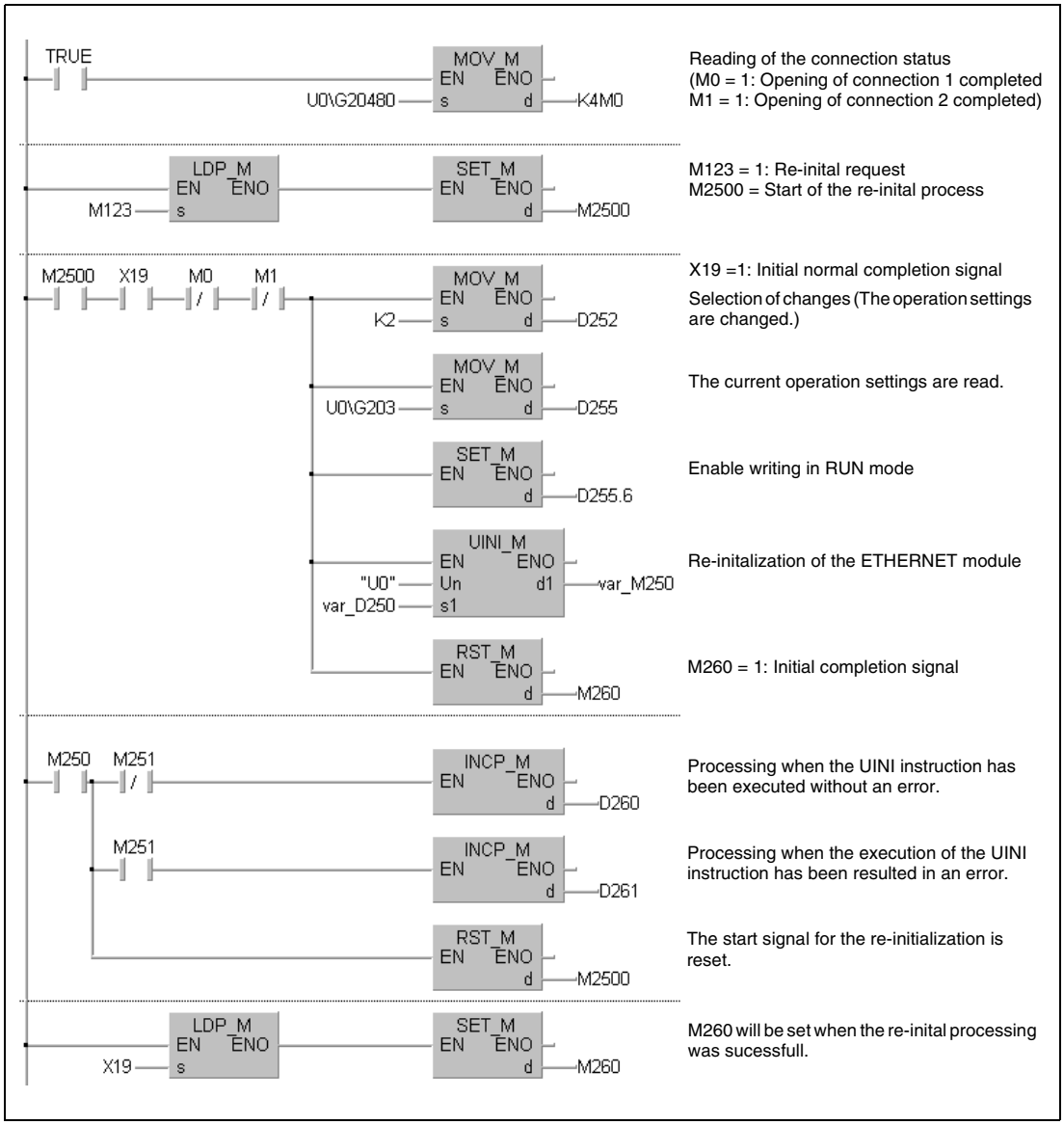
For the ETHERNET module with the head I/O address X/Y0 (Range from X/Y0 to X/Y1F) a re-initial process is performed.

NOTES

Only the connections 1 and 2 are used for this program example. When other connections are used the corresponding signals must be used.

For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.

- Ladder Diagram (GX IEC Developer)



- IEC Instruction List and MELSEC Instruction List
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

IEC Instruction List

```
LD      TRUE
MOV_M   U0\G20480, K4M0
PLS_M   M123
SET_M   M2500
AND     M2500
AND     X19
ANDN    M0
ANDN    M1
MOV_M   K2,        D252
MOV_M   U0\G203,   D255
SET_M   D255.6
UINI_M  "U0",      var_D250, var_M250
RST_M   M260
```

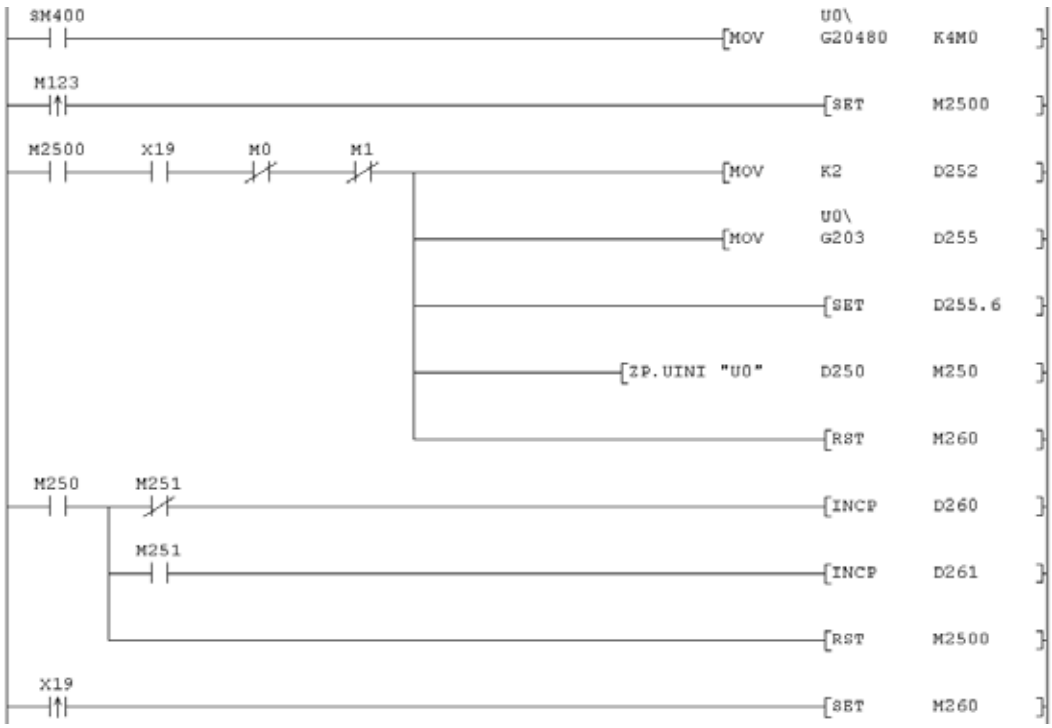
```
LD      M250
ANDN    M251
INCP_M  D260
LD      M250
AND     M251
INCP_M  D261
LD      M250
RST_M   M2500
PLS_M   X19
SET_M   M260
```

MELSEC Instruction List

```
MELSEC
LD      SM400
MOV     U0\G20480  K4M0
LDP     M123
SET     M2500
LD      M2500
AND     X19
ANI     M0
ANI     M1
MOV     K2         D252
MOV     U0\G203    D255
SET     D255.6
ZP.UINI "U0"       D250     M250
RST     M260
```

```
MELSEC
LD      M250
MPS
ANI     M251
INCP    D260
MRD
AND     M251
INCP    D261
MPP
RST     M2500
LDP     X19
SET     M260
```

- Ladder Diagram (GX Works2)
  The devices and instructions used are explained with the program example for the ladder diagram of the GX IEC Developer shown on the previous page.



12.4 Instructions for MELSECNET/H

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Setting of stations for duplex network	J.PAIRSET	PAIRSET_M

12.4.1 PAIRSET

CPU

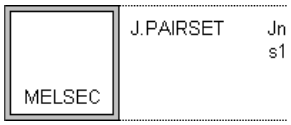
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

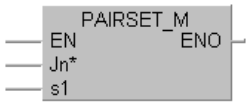
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—

GX IEC Developer

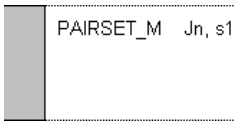
MELSEC Instruction List



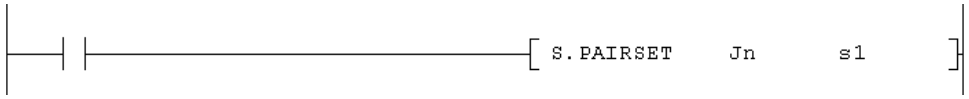
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Befehlswert	Data Type
Jn	Number of the network (1 to 239)	BIN 16-bit
s1	Head address of the device area where the settings for pairing are stored. File register (R, ZR) or the devices T, ST, C, D and W set in latch range can be used. When file registers are used, a memory card is required.	

Functions

Pairing setting of stations

PAIRSET Pairing setting instruction

This instruction specifies which station numbers are paired (duplexed). It is required to set up on the control station.

Structure of the device area storing the settings

- The setting of the stations in the devices designated by s1 cannot be done in a sequence program. It is necessary to load them in the PLC CPU by peripheral devices in advance.
- Four words are used regardless of the number of stations connected.
- It is only possible to pair two stations with neighbouring station numbers. For pairing, set in s1 the bit designating the station with the higher number.
- Each bit in the devices designated by (s1)+0 to (s1)+3 stands for a station number between 1 and 64:

Set Data	Bits															
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
(s1)+0	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
(s1)+1	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
(s1)+2	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33
(s1)+3	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49

NOTES

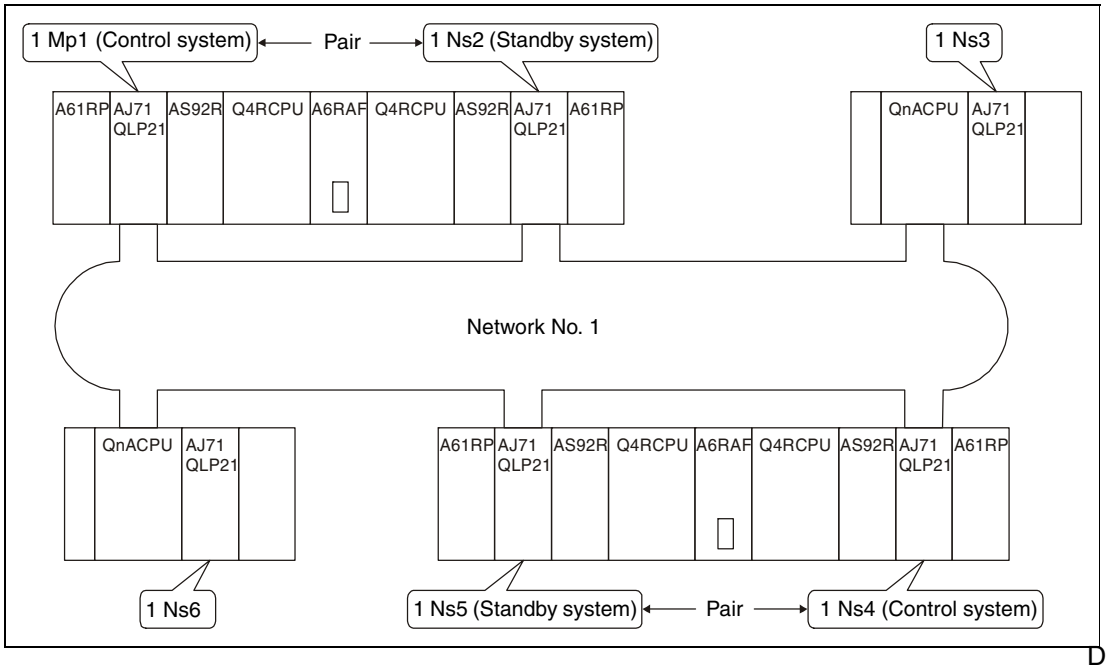
The pairing setting instruction is valid only on control stations. Any settings on normal stations are voided.

If in a redundant system consisting of Q4ARCPUs the control systems network module fails to data-link due to cable connection breakage, switching from control system to standby system is done only when pairing setting has been performed.

Program Example

PAIRSET

Pairing is performed for the stations 1 and 2 as well as for the stations 4 and 5 of a redundant system:

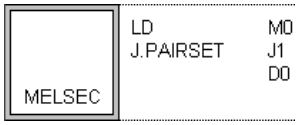


D

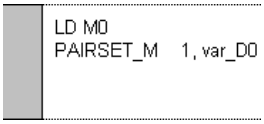
The settings are stored in the data registers D0 to D3. Bit 1 (b1) of D0 is set for the pairing of the stations 1 and 2 whereas b4 is set for the pairing of the stations 4 and 5:

Set Data	Bits															
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
D0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
D1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

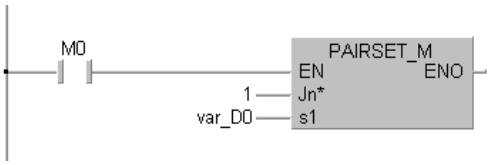
MELSEC Instruction List



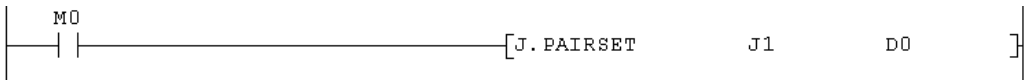
IEC Instruction List



Ladder Diagram (GX IEC Developer)



Ladder Diagram (GX Works2)



NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

12.5 Instructions for CC-Link

Function	MELSEC Instruction in MELSEC Editor	MELSEC Instruction in IEC Editor
Parameter setting for a CC-Link network (A series)	G.RLPA	RLPA_MD
	GP.RLPA	RLPA_P_MD
Parameter setting for a CC-Link network and start of the data link (System Q)	G.RLPASET	RLPASET_MD
	GP.RLPASET	RLPASET_P_MD
Setting of automatic refresh parameters (A series)	G.RRPA	RRPA_MD
	G.RRPA	RRPA_P_MD
Reading from the buffer memory of an intelligent device station or the device memory of the PLC CPU	G.RIRD	RIRD_MD
	GP.RIRD	RIRD_P_MD
Writing to the buffer memory of an intelligent device station or the device memory of the PLC CPU	G.RIWT	RIWT_MD
	GP.RIWT	RIWT_P_MD
Reading from the buffer memory of an intelligent device station (with handshake)	G.RIRCV	RIRCV_MD
	GP.RIRCV	RIRCV_P_MD
Writing to the buffer memory of an intelligent device station (with handshake)	G.RISEND	RISEND_MD
	GP.RISEND	RISEND_P_MD
Write to the automatic updated buffer memory	G.RITO	RITO_MD
	GP.RITO	RITO_P_MD
Read from the automatic updated buffer memory	G.RIFR	RIFR_MD
	GP.RIFR	RIFR_P_MD

12.5.1 RLPASET

CPU

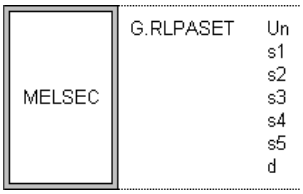
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

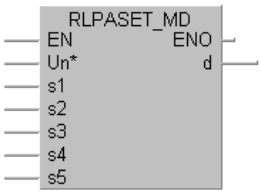
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
s3	—	●	●	—	—	—	—	—	—
s4	—	●	●	—	—	—	—	—	—
s5	—	●	●	—	—	—	—	—	—
d	●	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List



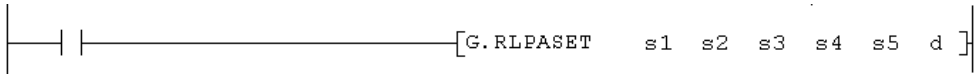
Ladder Diagram



IEC Instruction List

```
RLPASET_MD  Un, s1, s2, s3, s4, d
```

GX Works2



Variables

Set Data	Meaning	Range	Contents is stored by	Data Type
Un	Head I/O address of the CC-Link master module (Only the upper two digits of an address expressed as a 3-digit number are entered, e. g. the head address X/Y100 is set as 10H)	0 to FEH	User	BIN 16-bit
s1	Head number of the devices where control data for the execution of this instruction is stored.			Address

Set Data	Meaning	Description	Range	Contents is stored by
(s1)+0	Execution result of the instruction	Indicates whether an error has occured during execution of the instruction: 0000H: No error Any value other than 0000H: Error code	—	System
(s1)+1	Validation of the settings	The first four bits are used to specify whether the settings made in s2 to s5 are valid or invalid: Bit 0 = 1:Slave station settings (s2) Bit 1 = 1:Reserved station specifications (s3) Bit 2 = 1:Error invalid station specifications (s4) Bit 3 = 1:Send, receive and automatic refresh buffer assignment data (s5) For the settings marked as invalid the default parameters will be applied.	0 to F	User
(s1)+2	Number of connected modules	Set the number of connected slave stations (including the reserved stations)	1 to 64	
(s1)+3	Number of retries	Set the number of retries to a communication faulty station.	1 to 7	
(s1)+4	Number of automatic return modules	Set the number of slave modules which after a failure can be returned automatically to the link within one link scan.	1 to 10	
(s1)+5	Operation specification when the PLC CPU has stopped	Specifies the data link status when a master station PLC CPU error occurs. 0: Stop 1: Continue	0 or 1	
(s1)+6	Scan mode specification	Choose betwen the synchronous and the asynchrous mode 0: The data link is synchronous to the scan of the sequence program 1: The data link is asynchronous to the scan of the sequence program.	0 or 1	
(s1)+7	Delay time setting	Link scan intervall (Unit: 50 µs)	0 to 100	

Variables

<table>
<tr><th>Set Data</th><th colspan="3">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="8">s2</td><td colspan="5">Head device of the area where slave station settings are stored.</td><td rowspan="8">Address</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td colspan="2">Description</td><td>Contents is stored by</td></tr>
<tr><td>(s2)+0</td><td>Settings for station No. 1</td><td colspan="2" rowspan="5">See the table at page 93
Make the settings for as much modules as are specified in (s1)+2 as number of connected modules.</td><td rowspan="5">User</td></tr>
<tr><td>(s2)+1</td><td>Settings for station No. 2</td></tr>
<tr><td>•
•</td><td>•
•</td></tr>
<tr><td>(s2)+62</td><td>Settings for station No. 63</td></tr>
<tr><td>(s2)+63</td><td>Settings for station No. 64</td></tr>
<tr><td rowspan="6">s3</td><td colspan="5">Head device of the area where specifications for reserved stations are stored.
Perform the setting for all stations up to the largest station number set in s2.</td><td rowspan="6">Address</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td colspan="2">Description</td><td>Contents is stored by</td></tr>
<tr><td>(s3)+0</td><td>Setting for station No.'s 1 – 16</td><td colspan="2" rowspan="4">Specify a reserved station by setting the bit for the corresponding station number (see the table at page 93).
Specify only the head station number of a module that occupies 2 or more stations.

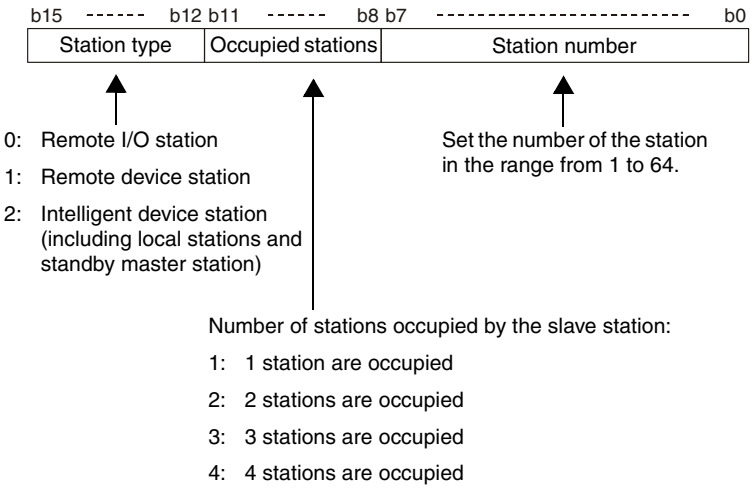
No station is reserved in the default parameter setting.</td><td rowspan="4">User</td></tr>
<tr><td>(s3)+1</td><td>Setting for station No.'s 17 – 32</td></tr>
<tr><td>(s3)+2</td><td>Setting for station No.'s 33 – 48</td></tr>
<tr><td>(s3)+3</td><td>Setting for station No.'s 49 – 64</td></tr>
<tr><td rowspan="6">s4</td><td colspan="5">Head device of the area where specifications for error invalid stations are stored.
Perform the setting for all stations up to the largest station number set in s2.</td><td rowspan="6">Address</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td colspan="2">Description</td><td>Contents is stored by</td></tr>
<tr><td>(s4)+0</td><td>Setting for station No.'s 1 – 16</td><td colspan="2" rowspan="4">When the error of a station should be ignored, set the bit for the corresponding station number (see the table at page 93). Specify only the head station number of a module that occupies 2 or more stations.
The reserved station number is given the higher priority if both error invalid station and reserved station specifications are made for the same station.
No error invalid station is set in the default parameter setting.</td><td rowspan="4">User</td></tr>
<tr><td>(s4)+1</td><td>Setting for station No.'s 17 – 32</td></tr>
<tr><td>(s4)+2</td><td>Setting for station No.'s 33 – 48</td></tr>
<tr><td>(s4)+3</td><td>Setting for station No.'s 49 – 64</td></tr>
</table>

Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="14">s5</td><td colspan="4">Head device of the area where settings for the buffer memory size are stored.
Perform the settings for stations specified in s2 as local stations and intelligent device stations. Start with the smallest station number.</td><td></td><td></td><td rowspan="14">Address</td></tr>
<tr><td>Set Data</td><td colspan="2">Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s5)+0</td><td rowspan="12">First module</td><td>Send buffer size</td><td rowspan="2">Specify the buffer size needed for communication between the master station and local stations or intelligent device stations.
The maximum size of the send and receive buffer together is 4096 words (1000H).
For the sending/receiving buffer size, specify a number 7 words larger than the size actually required for communication.</td><td>0H:
No buffer
40H to 1000H
(64 to 4096 words)
Default setting: 40H</td><td rowspan="12">User</td></tr>
<tr><td>(s5)+1</td><td>Receive buffer size</td><td>0H:
No buffer
40H to 1000H
(64 to 4096 words)
Default setting: 40H</td></tr>
<tr><td>(s5)+2</td><td>Automatic refresh buffer size</td><td>Number of points of the automatic refresh buffer used for communication between the master station and local stations or intelligent device stations.
The size of the automatic updating buffer must be equal to the size necessary for the individual intelligent device station.</td><td>0H:
No buffer
80H to 1000H
(128 to 4096 words)
Default setting: 80H</td></tr>
<tr><td>•
•</td><td>•
•</td><td>•
•</td><td>•
•</td><td>•
•</td></tr>
<tr><td>(s5)+75</td><td rowspan="3">Twenty sixth module</td><td>Send buffer size</td><td colspan="2" rowspan="3">The same as for the 1st module.</td></tr>
<tr><td>(s5)+76</td><td>Receive buffer size</td></tr>
<tr><td>(s5)+77</td><td>Automatic refresh buffer size</td></tr>
<tr><td rowspan="4">d</td><td colspan="4">Bit device which is set for one scan after completion of the RLPASET instruction. (d)+1 indicates that an error has occured during execution of the instruction.</td><td></td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td colspan="2">Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d)+0</td><td colspan="2">Instruction completed</td><td>Indicates the completion of the RLPASET instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>0 or 1</td><td rowspan="2">System</td></tr>
<tr><td>(d)+1</td><td colspan="2">Instruction completed with error</td><td>Indicates that an error has occured during the processing of the RLPASET instruction
ON: Abnormal completion
OFF: Normal completion</td><td>0 or 1</td></tr>
</table>

Slave station settings

For each station a word device ((s2)+0 to (s2)+63) is reserved which contains settings for this station:

Meaning	Description	Range
Settings for 1 to 64 modules		b0 to b7: 1 – 64 (01H – 40H) b8 to b11: 1 – 4 b12 to b15: 0 – 2

The default parameter settings for (s2)+0 to (s2)+63 are „0101H“ to „0140H“. (Station number 1 to 64, one station occupied, remote I/O station)

Designation of the station number in s3 and s4

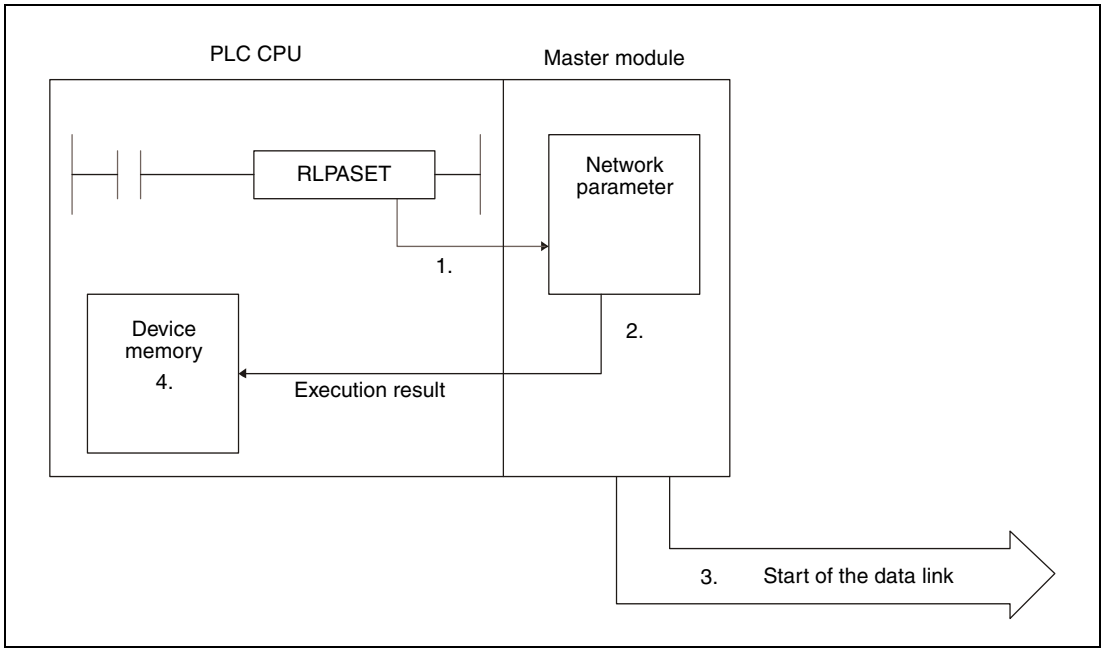
Each bit of the four word devices used for s3 and s4 represents one station:

Set Data	Bit															
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
(s3)+0 (s4)+0	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
(s3)+1 (s4)+1	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
(s3)+2 (s4)+2	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33
(s3)+3 (s4)+3	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49

The numbers 1 to 64 in the table indicate a station number. When a bit is set the corresponding station is selected.

Functions **Parameter setting for a CC-Link Network and start of the data link**

RLPASET Parameter setting instruction



1. The network parameters stored in (s1) to (s5) are send to master module of the CC-Link designated by Un using the RLPASET instruction.
2. The received settings are checked by the master module.
3. If the settings are correct, the data link is started.
4. The device specified by (d) is set.

It is only possible to execute one RLPASET instruction at a time.

Number of required devices

The following numbers of devices are required for the RLPASET instruction:

- s1: 8 word devices
- s2: 64 word devices
- s3: 4 word devices
- s4: 4 word devices
- s5: 78 word devices

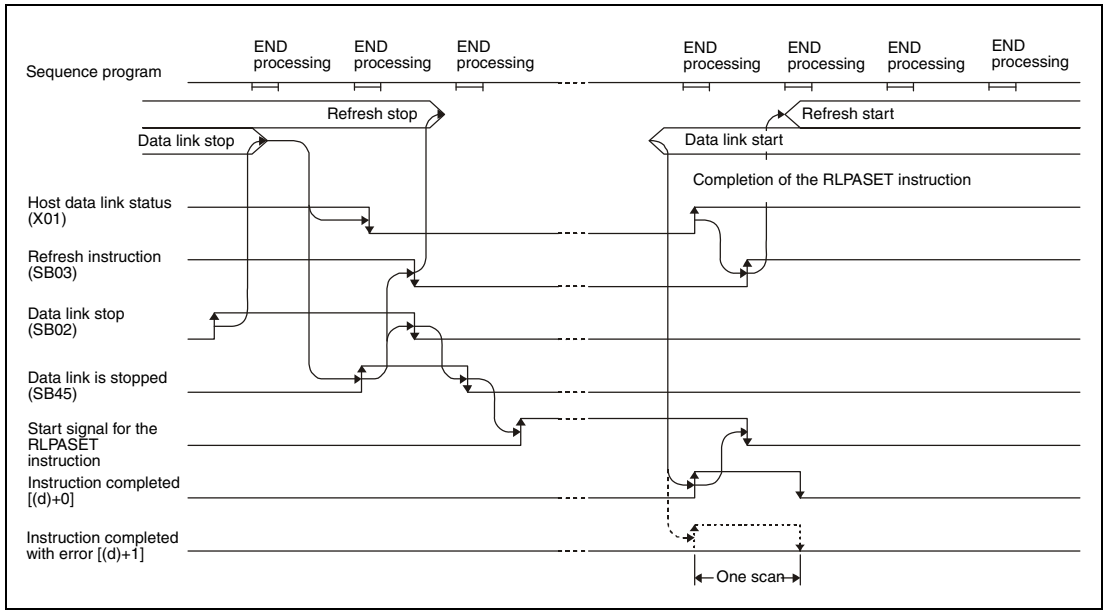
Please note the required areas for (s1) to (s5) during programming.

An example:
Four slave stations are connected to a master module. In the Q02CPU mounted in the PLC of the master station the data link registers D0 to D12287 are available. If D12284 is designated as head device for (s2) because there are only four slave stations, the execution of the RLPASET instruction will result in an error with the code 4101. This is because the PLC CPU always checks the range for 64 stations (D12284 to D12347 in this example) and in this case the available range is exceeded.

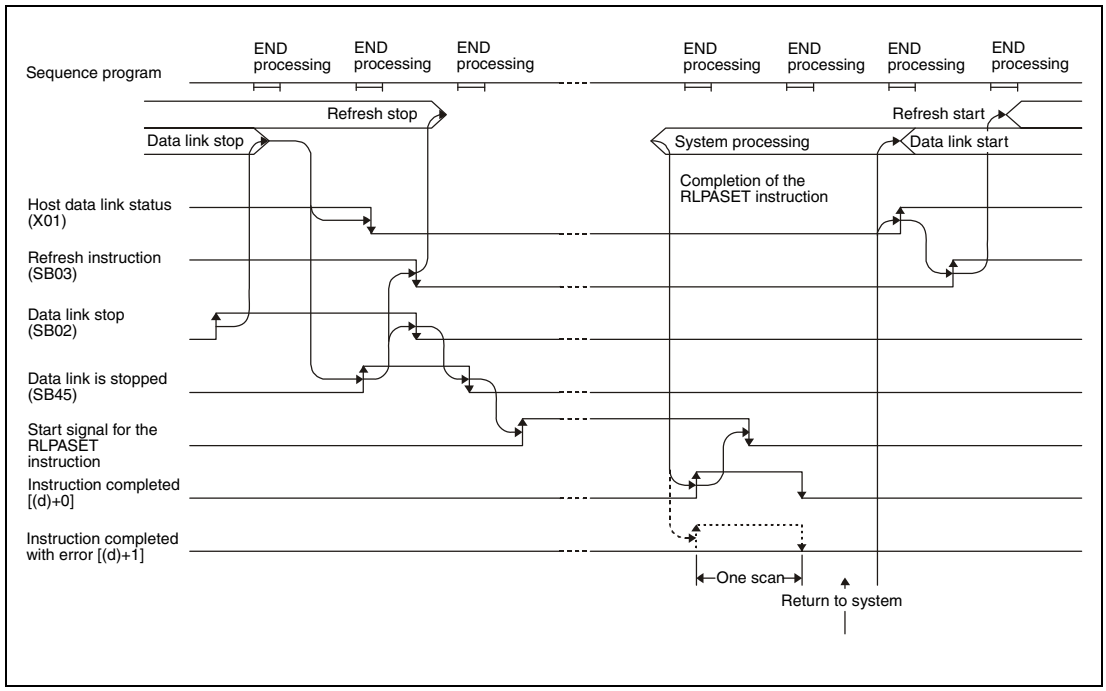
Whether the execution of the RLPASET instruction has been finished can be checked with the devices (d1)+0 and (d1)+1:

- The bit device (d1)+0 turns ON at the END processing of the scan in which the RLPASET instruction has been completed and turns OFF at the next END processing.
- The bit device (d1)+1 indicates an error during execution of the UINI instruction. When the instruction has been completed normal, this device stays OFF. When an error occurs during execution of the UINI instruction, (d1)+1 turns ON at the END processing of the scan in which the RLPASET instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the RLPASET instruction is executed and all stations are normal:



The timing for the RLPASET instruction in the case of a faulty station is shown below:



Operation Error

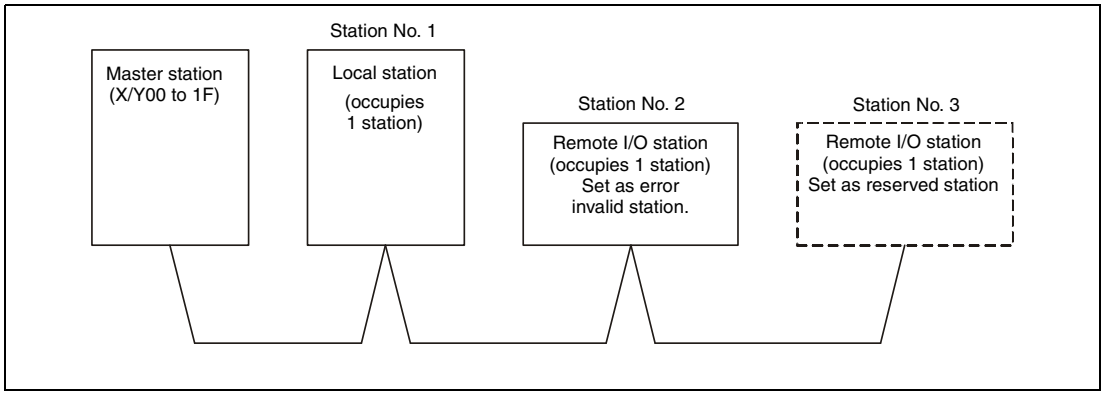
In the following cases an operation error occurs, the error flag SM0 is set, and an error code is stored in SD0:

- When the module designated by (Un) is not a special function module. (error code: 2112)
- When an attempt was made to execute an unsupported instruction. (error code: 4002)
- When the number of devices in the instruction is incorrect. (error code: 4003)
- When the instruction specifies a device that cannot be used. (error code: 4004)
- When the instruction contains data that cannot be used. (error code: 4100)
- When the number of points for data used in the instruction exceeds the available range, or storage data and constants of a device specified by the instruction exceeds the available range (including dummy devices). (error code: 4101)

Program Example

RLPASET

This program transfers the network parameter to the master station occupying the head I/O number X/Y000. The CC-Link network consists of three slave stations:



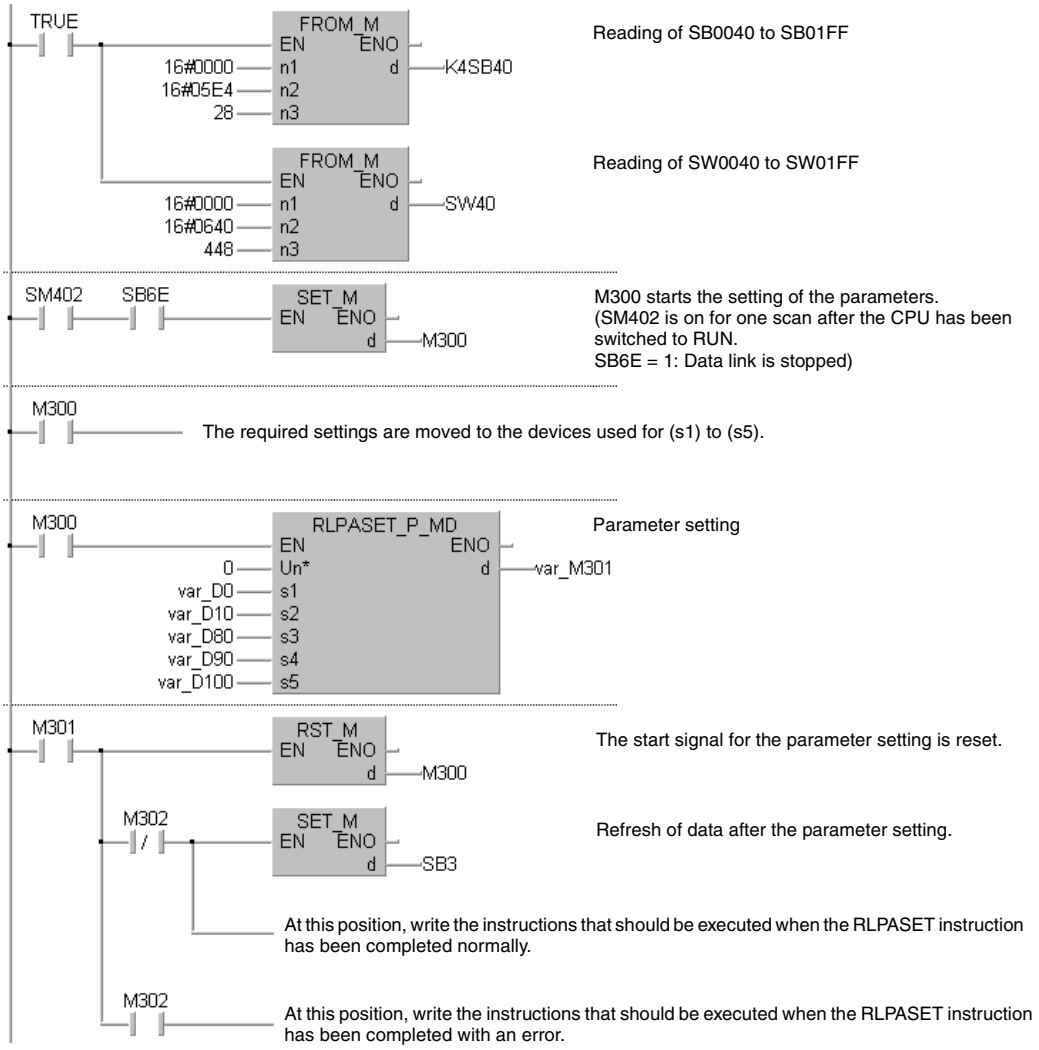
The devices designated by (s1) to (s5) are holding the following values:

Parameter			Setting	Set value	Allocated device
Control data for the execution of the instruction	(s1)+1	Validation of the settings	All settings are valid.	15	D1
	(s1)+2	Number of connected modules	3 slave modules	3	D2
	(s1)+3	Number of retries	3 times	3	D3
	(s1)+4	Number of automatic return modules	1 module	1	D4
	(s1)+5	Operation specification when the PLC CPU has stopped	Stop	0	D5
	(s1)+6	Scan mode specification	Asynchonous	0	D6
	(s1)+7	Delay time setting	0 µs	0	D7
Settings for slave stations	(s2)+0	Settings for the first station	Local station, occupies 1 station, Station No. 1	2101H	D10
	(s2)+1	Settings for the second station	Remote I/O station, occupies 1 station, Station No. 2	102H	D11
	(s2)+2	Settings for the third station	Remote I/O station, occupies 1 station, Station No. 3	103H	D12
Reserved stations	(s3)+0	Selection of reserved stations	Station No. 3 is reserved (bit 2 is set)	4	D80
	(s3)+1			0	D81
	(s3)+2			0	D82
	(s3)+3			0	D83
Error invalid stations	(s4)+0	Specification of error invalid stations	Station No. 2 (bit 1 is set)	2	D90
	(s4)+1			0	D91
	(s4)+2			0	D92
	(s4)+3			0	D93
Buffer sizes	(s5)+0	Send buffer of the first local station (Station No. 1)	100 words	64H	D100
	(s5)+1	Receive buffer of the first local station (Station No. 1)	100 words	64H	D101
	(s5)+2	Automatic refresh buffer of the first local station (Station No. 1)	Not used	0H	D102

The contents of the data registers D1 to D102 must be set according to the above table before the RLPA instruction is called.

- IEC editors (On the next page this program example is shown for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



IEC Instruction List

```
LD              TRUE
FROM_M          16#0000,    16#05E4,   28,     K4SB40
FROM_M          16#0000,    16#0640,   448,    SW40
```

```
LD              SM402
AND             SB6E
SET_M           M300
```

For an explanation of the devices and instructions used please see the program example on the previous page.

```
LD              M300
```

The required settings are written to the devices designated by (s1) to (s5) using MOV instructions.

```
LD              M300
RLPASET_P_MD    0, var_D0, var_D10, var_D80, var_D90, var_D100, var_M301
```

```
LD              M301
RST_M           M300
LD              M301
ANDN            M302
SET_M           SB3
```

An instruction at this position will be executed when the RLPASET instruction has been completed normally.

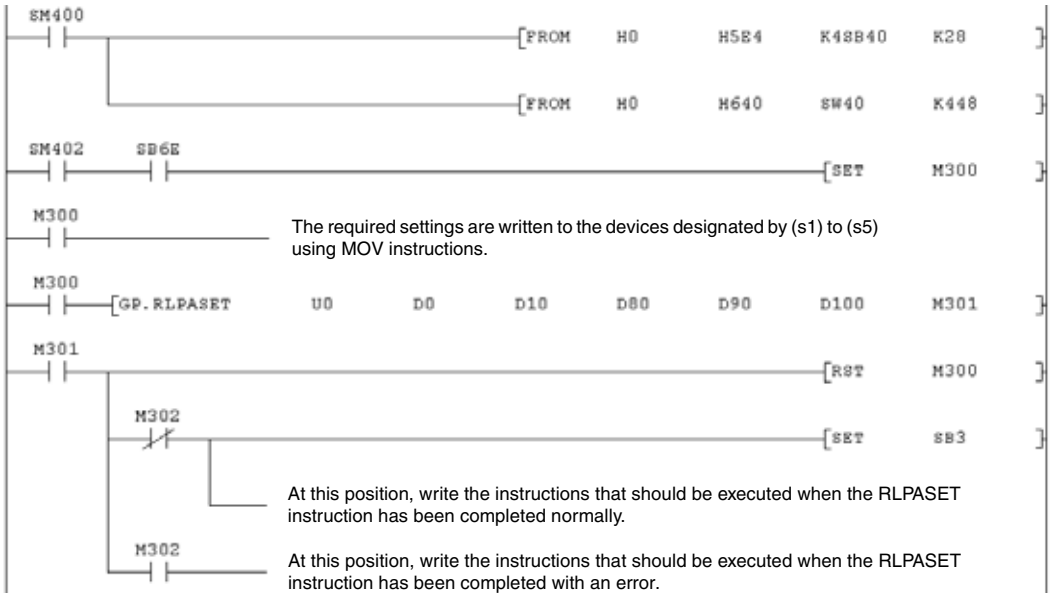
```
LD              M301
AND             M302
```

An instruction at this position will be executed when the processing of the RLPASET instruction has been completed with an error.

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer.

Ladder Diagram (GX Works2)



MELSEC Instruction List

```
MELSEC   LD          SM400
         FROM        H0
                     H5E4
                     K4SB40
                     K28
         FROM        H0
                     H640
                     SW40
                     K448

MELSEC   LD          SM402
         AND         SB6E
         SET         M300

MELSEC   LD          M300
The required settings are written to the devices designated by
(s1) to (s5) using MOV instructions.

MELSEC   LD          M300
         GP.RLPASET  U0
                     D0
                     D10
                     D80
                     D90
                     D100
                     M301

MELSEC   LD          M301
         RST         M300
         MPS
         ANI         M302
         SET         SB3
An instruction at this position will be executed when the RLPASET instruction has been completed normally.
         MPP
         AND         M302
An instruction at this position will be executed when the processing of the RLPASET instruction has been
completed with an error.
```

12.5.2 RIRD

CPU

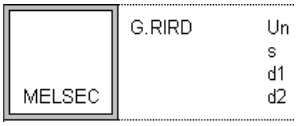
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

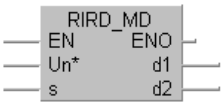
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d1	—	●	●	—	—	—	—	—	—
d2	●	●	●	—	—	—	—	—	—

GX IEC Developer

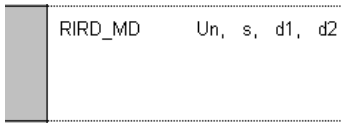
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>Un</td><td colspan="4">Head I/O address of the CC-Link master/local module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as 10H)</td><td>0 to FEH</td><td>User</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="10">s</td><td colspan="4">Head number of the devices where control data for the execution of this instruction is stored</td><td></td><td></td><td rowspan="10">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s)+0</td><td>Execution result of the instruction</td><td>Indicates whether an error has occured during execution of the instruction:
0000H: No error
Any value other than 0000H: An error has occured. The stored value is an error code which is explained in the user's manual of the CC-Link module.</td><td>—</td><td>System</td></tr>
<tr><td>(s)+1</td><td>Station number</td><td>Staton number of the remote station, where data is read from</td><td>0 to 64</td><td rowspan="5">User</td></tr>
<tr><td rowspan="2">(s)+2</td><td rowspan="2">Acces code

Device code and access code</td><td>• For a A/Q series master module with software version A to H
Set „0004H“ to access the buffer memory of an intelligent device station.
Set „2004H“ to access the buffer memory of a local station.</td><td>0004H
or
2004H</td></tr>
<tr><td>• For a A/Q series master module with software version J or higher or a module of System Q
A device code is stored in the upper 8 bits of this device.
The access code which, specifies whether to access the buffer memory of a CC-Link module (04H) or a CPU device (05H), is entered in the lower 8 bits.</td><td>Higher byte: see the table below

Lower byte:
04H
or
05H</td></tr>
<tr><td rowspan="2">(s)+3</td><td rowspan="2">Head address</td><td>• For a A/Q series master module with software version A to H
Head address of the buffer memory</td><td rowspan="2">Depends on the accessed station</td></tr>
<tr><td>• For a A/Q series master module with software version J or higher or a module of System Q
Head address of the buffer memory or first device number</td></tr>
<tr><td>(s)+4</td><td>Number of points to read</td><td>Specify the number of data (unit:words) to read-out.
This number depends on the type of CPU module mounted in the station where the data is read from:
AnU, QnA series, System Q: max. 480 words
All other CPUs: max. 32 words</td><td>1 to 480
1 to 32</td></tr>
<tr><td>d1</td><td colspan="4">Head address of the area where the read data is stored</td><td>User</td><td>BIN 16-bit</td></tr>
</table>

Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="4">d2</td><td colspan="5">Bit device which is set for one scan after completion of the RIRD instruction. (d2)+1 indicates that an error has occured during execution of the instruction.</td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d2)+0</td><td>Instruction completed</td><td>Indicates the completion of the RIRD instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d2)+1</td><td>Instruction completed with error</td><td>Indicates that an error has occured during the processing of the RIRD instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

From software version J of the master module two codes (both stored in s+2) are used to specify the data to read: The **access code** selects whether access is made to the buffer memory of a CC-Link module or the device memory in the CPU module. With the **device code** the area of the buffer memory or the device is designated:

- Access to the buffer memory of a CC-Link module (Access code: 04H)

Access to		Device code
Buffer memory in an intelligent device station		00H
Buffer memory in a master or local station	Random access buffer	20H
	Remote inputs	21H
	Remote outputs	22H
	Remote register	24H
	Link special relays	63H
	Link special register	64H

- Access to the device memory of a CPU module (Access code: 05H)
  Devices not indicated in the following table are not accessible. To access a bit device, specify „0“ or a multiple of „16“ as head device. Otherwise an error will occur.

Device		Device type		Unit	Device code
Name	Symbol	Bit	Word		
Inputs	X	●		Hexadeci-mal	00H
Outputs	Y	●			02H
Internal relays	M	●		Decimal	03H
Latch relays	L	●			83H
Link relays	B	●		Hex.	23H
Timer (contact)	T	●		Decimal	09H
Timer (coil)		●			0AH
Timer (present value)			●		0CH
Retentive Timer (contact)	ST	●			89H
Retentive Timer (coil)		●			8AH
Retentive Timer (present value)			●		8CH
Counter (contact)	C	●		Decimal	11H
Counter (coil)		●			12H
Counter (present value)			●		14H
Data register	D		●		04H
Link register	W		●	Hex.	24H
File register	R		●	Decimal	84H
Special link relay	SB	●		Hexadeci-mal	63H
Special link register	SW		●		64H
Special relay	SM	●		Decimal	43H
Special register	SD		●		44H

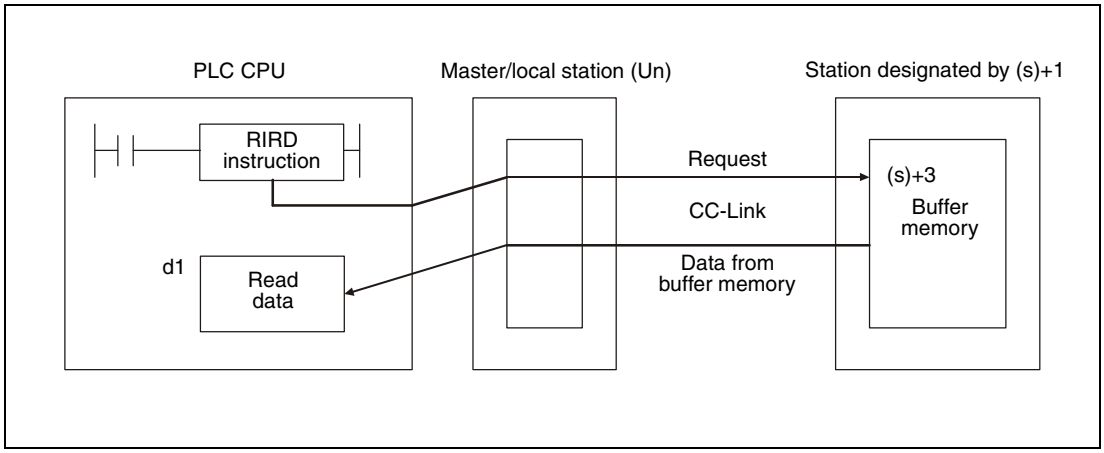
Functions **Read from buffer memory of intelligent device station or from device memory of PLC CPU**

RIRD Data read

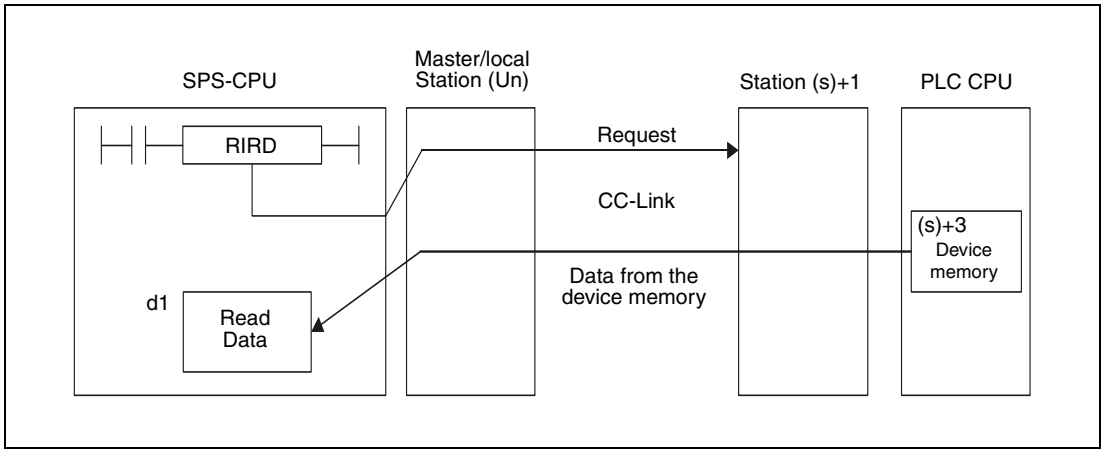
The RIRD instruction reads data from the buffer memory of an intelligent device connected to the CC-Link. When a master module with a software version from J onward or a CC-Link module of the MELSEC System Q is used, it is also possible to access the PLC CPU device memory of another station connected to the CC-Link network.

The head address of the buffer memory or the head device is designated by (s)+3. The station number of the other station is designated by (s)+1. This station is connected to the master/local station specified at Un. The read data is stored in the CPU which executes the RIRD instruction to the devices starting from d1. The number of data to read is designated by (s)+4.

- Accessing the buffer memory of an CC-Link module



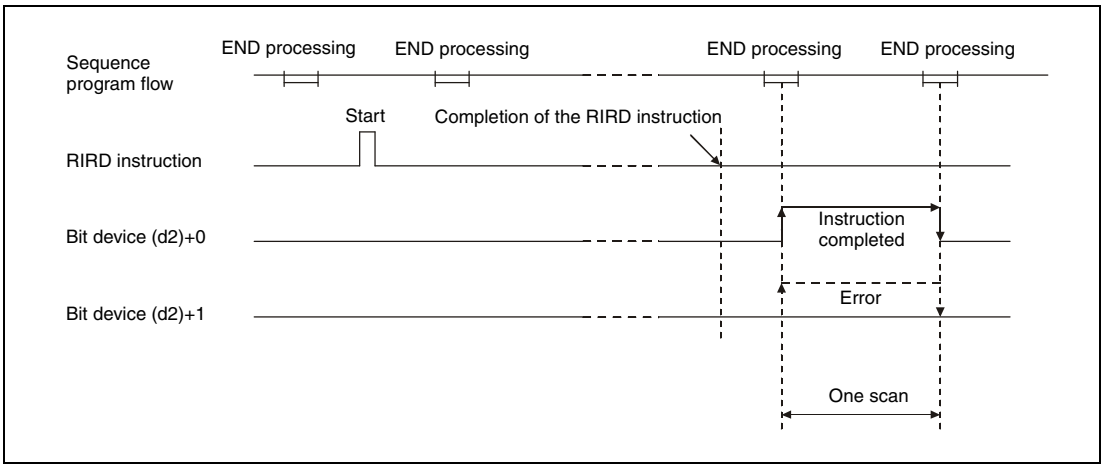
- Accessing the device memory in the PLC CPU of another station on CC-Link



Whether the execution of the RIRD instruction has been finished can be checked with the devices (d2)+0 and (d2)+1:

- The bit device (d2)+0 turns ON at the END processing of the scan in which the RIRD instruction has been completed and turns OFF at the next END processing.
- The bit device (d2)+1 indicates an error during execution of the RIRD instruction. When the instruction has been completed normal, this device stays OFF. When an error occurs during execution of the RIRD instruction, (d2)+1 turns ON at the END processing of the scan in which the RIRD instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the RIRD instruction is being executed:



It is possible to execute RIRD instructions for multiple stations at the same time, but it is not possible to access the same intelligent device station or local station simultaneously from more than one station.

Operation Error

In the following cases an operation error occurs, the error flag SM0 is set, and an error code is stored in SD0:

- When the module specified by Un is not an intelligent function module or a special function module. (Error code: 2112)
- When an attempt was made to execute an unsupported instruction. (Error code: 4002)
- When the number of devices in the instruction is incorrect. (Error code: 4003)
- When the instruction specifies a device that cannot be used. (Error code: 4004)
- When the area designated by s contains data that cannot be used. (Error code: 4100)
- When the number of data set to be used exceeds the allowable range. (Error code: 4101)
- When the storage data or constants of the device specified with the instruction exceeds the allowable range. (Error code: 4101)

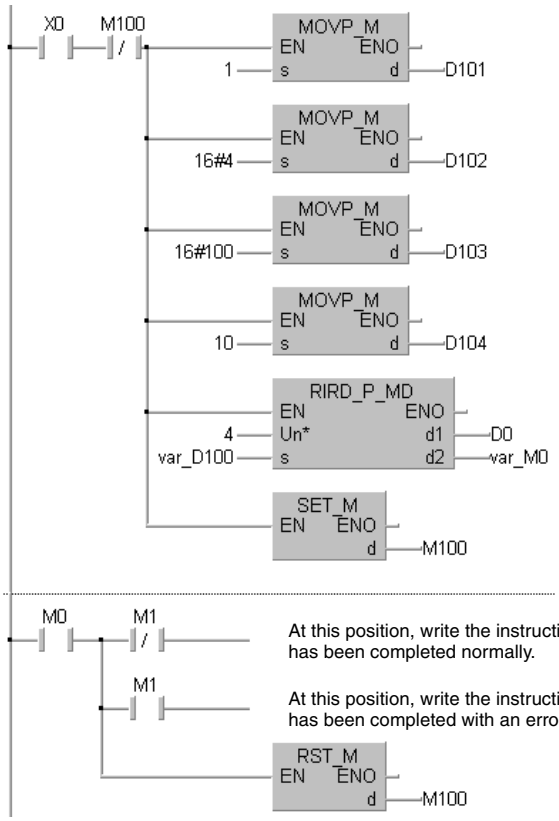
Program Example

RIRD

The following program is executed in the PLC CPU of the master station. When the input X0 is set the contents of 10 buffer memory addresses is read from the intelligent device station with the station number, starting with the buffer memory address 100H. The read data is stored in the PLC CPU from data register D0 onward. The head I/O number of the master module of CC-Link is X/Y40.

- IEC editors (On the next page this program example is shown for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



The station number is written to (s)+1

To access the buffer memory of an intelligent device station the code 4H is written to (s)+2.

The head address of the buffer memory is stored in (s)+3.

Number of words to read

Read instruction

M100 is set to indicate that data is being read.

At this position, write the instructions that should be executed when the RIRD instruction has been completed normally.

At this position, write the instructions that should be executed when the RIRD instruction has been completed with an error.

When the reading of data has been completed M100 is reset.

IEC Instruction List

```
LD          X0
ANDN        M100
MOVP_M      1,          D101
MOVP_M      16#4,       D102
MOVP_M      16#100,     D103
MOVP_M      10,         D104
RIRD_P_MD   4,          var_D100,   D0,   var_M0
SET_M       M100
```

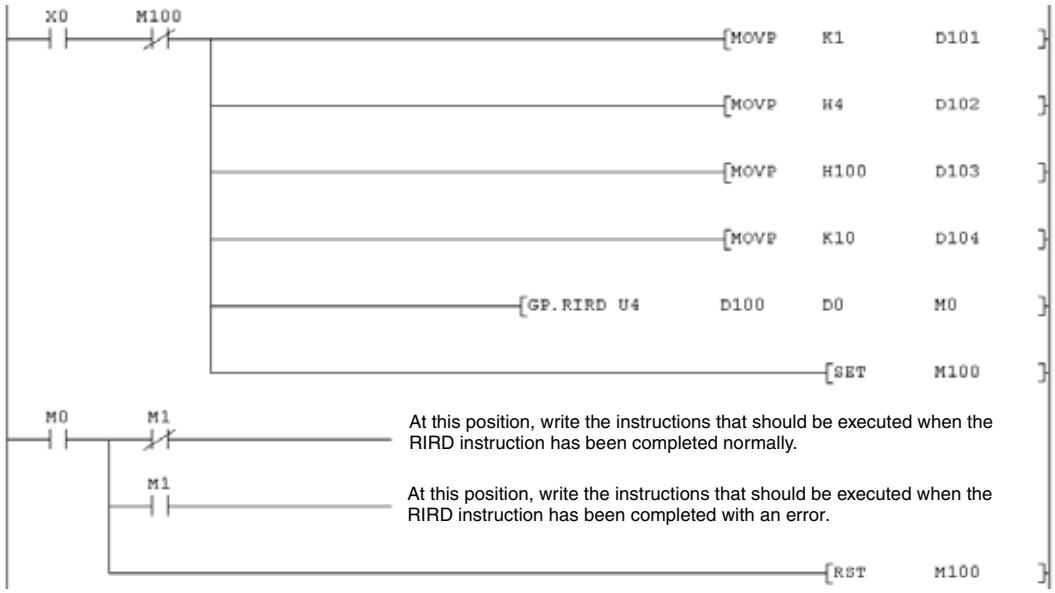
For an explanation of the devices and instructions used please see the above program example.

```
LD          M0
ANDN        M1
An instruction at this position will be executed when the RIRD instruction has been completed normally.
LD          M0
AND         M1
An instruction at this position will be executed when the RIRD instruction has been completed annormally.
LD          M0
RST_M       M100
```

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



MELSEC Instruction List

MELSEC	LD	X0	
	ANI	M100	
	MOVP	K1	D101
	MOVP	H4	D102
	MOVP	H100	D103
	MOVP	K10	D104
	GP.RIRD	U4	
		D100	
		D0	
		M0	
	SET	M100	

MELSEC	LD	M0	
	MPS		
	ANI	M1	
	An instruction at this position will be executed when the RIRD instruction has been completed normally.		
	MRD		
	AND M1		
	An instruction at this position will be executed when the RIRD instruction has been completed annormally.		
	MPP		
	RST	M100	

12.5.3 RIWT

CPU

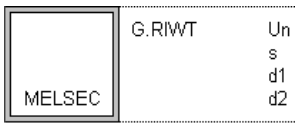
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

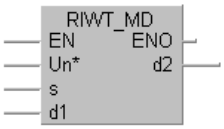
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s	—	●	●	—	—	—	—	—	—
d1	—	●	●	—	—	—	—	—	—
d2	●	●	●	—	—	—	—	—	—

GX IEC Developer

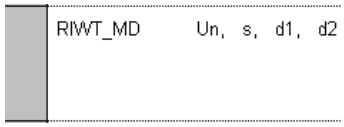
MELSEC Instruction List



Ladder Diagram



IEC Instruction List



GX Works2

Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>Un</td><td colspan="4">Head I/O address of the CC-Link master/local module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as 10H)</td><td>0 to FEH</td><td>User</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="9">s</td><td colspan="4">Head number of the devices where control data for the execution of this instruction is stored</td><td></td><td></td><td rowspan="9">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s)+0</td><td>Execution result of the instruction</td><td>Indicates whether an error has occured during execution of the instruction:
0000H: No error
Any value other than 0000H: An error has occured. The stored value is an error code which is explained in the user's manual of the CC-Link module.</td><td>—</td><td>System</td></tr>
<tr><td>(s)+1</td><td>Station number</td><td>Station number of the remote station, where data is written to.</td><td>0 to 64</td><td rowspan="5">User</td></tr>
<tr><td rowspan="2">(s)+2</td><td rowspan="2">Access code</td><td>• For a A/Q series master module with software version A to H
Set „0004H“ to write to the buffer memory of an intelligent device station.
Set „2004H“ to write to the buffer memory of a local station.</td><td>0004H
or
2004H</td></tr>
<tr><td rowspan="2">Device code and access code</td><td>• For a A/Q series master module with software version J or higher or a module of System Q
A device code is stored in the upper 8 bits of this device.
The access code, which specifies whether to access the buffer memory of a CC-Link module (04H) or a CPU device (05H), is entered in the lower 8 bits.</td><td>Higher byte: see the table below

Lower byte: 04H or 05H</td></tr>
<tr><td rowspan="2">(s)+3</td><td rowspan="2">Head address</td><td>• For a A/Q series master module with software version A to H
Head address of the buffer memory</td><td rowspan="2">Depends on the accessed station</td></tr>
<tr><td>• For a A/Q series master module with software version J or higher or a module of System Q
Head address of the buffer memory or head device</td></tr>
<tr><td>(s)+4</td><td>Datenlänge</td><td>Specify the number of data (unit:words) to write.
This number depends on the type of CPU module mounted in the station where the data is written to:
AnU, QnA series, System Q:
max. 480 words
All other CPUs: max. 32 words</td><td>1 to 480
1 to 10</td></tr>
<tr><td>d1</td><td colspan="4">Head address of the area where the write data is stored</td><td>User</td><td>BIN 16-bit</td></tr>
</table>

Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="4">d2</td><td colspan="5">Bit device which is set for one scan after completion of the RIWT instruction. (d2)+1 indicates that an error has occured during execution of the instruction.</td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d2)+0</td><td>Instruction completed</td><td>Indicates the completion of the RIWT instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d2)+1</td><td>Instruction completed with error</td><td>Indicates that an error has occured during the processing of the RIWT instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

From software version J of the master module two codes (both stored in (d1)+2) are used to specify the target for the data: The **access code** selects whether data is written to the buffer memory of a CC-Link module or the device memory in the CPU module. With the **device code** the area of the buffer memory or the devices, which will be overwritten, is designated:

- Access to the buffer memory of a CC-Link module (Access code: 04H)

Access to		Device code
Buffer memory in an intelligent device station		00H
Buffer memory in a master or local station	Random access buffer	20H
	Remote inputs	21H
	Remote outputs	22H
	Remote register	24H
	Link special relays	63H
	Link special register	64H

- Access to the device memory of a CPU module (Access code: 05H)
  Devices not indicated in the following table are not accessible. To access a bit device, specify „0“ or a multiple of „16“ as head device. Otherwise an error will occur.

Device		Device type		Unit	Device code
Name	Symbol	Bit	Word		
Inputs	X	●		Hexadeci-mal	00H
Outputs	Y	●			02H
Internal relays	M	●		Decimal	03H
Latch relays	L	●			83H
Link relays	B	●		Hex.	23H
Timer (contact)	T	●		Decimal	09H
Timer (coil)		●			0AH
Timer (present value)			●		0CH
Retentive Timer (contact)	ST	●			89H
Retentive Timer (coil)		●			8AH
Retentive Timer (present value)			●		8CH
Counter (contact)	C	●		Decimal	11H
Counter (coil)		●			12H
Counter (present value)			●		14H
Data register	D		●		04H
Link register	W		●	Hex.	24H
File register	R		●	Decimal	84H
Special link relay	SB	●		Hexadeci-mal	63H
Special link register	SW		●		64H
Special relay	SM	●		Decimal	43H
Special register	SD		●		44H

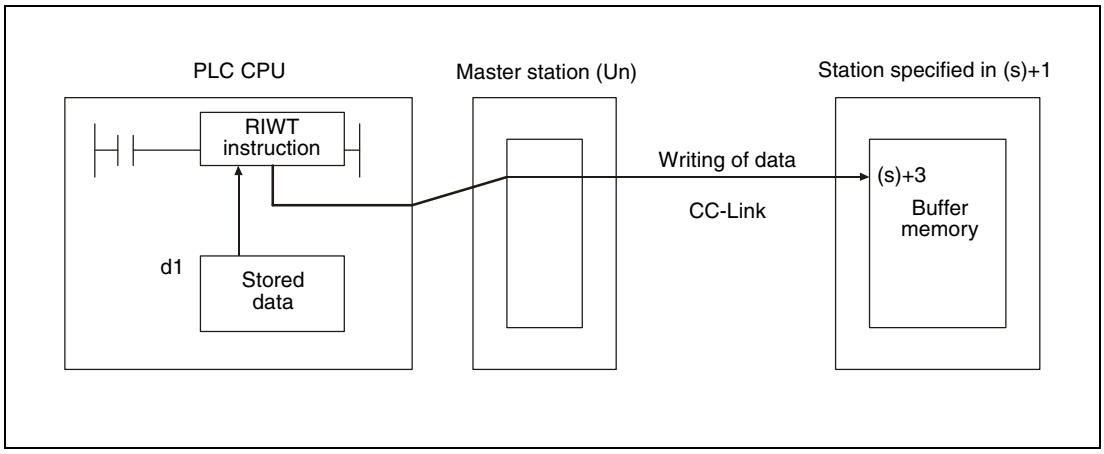
Functions **Write to buffer memory of intelligent device station or to device memory of PLC CPU**

RIWT Data write

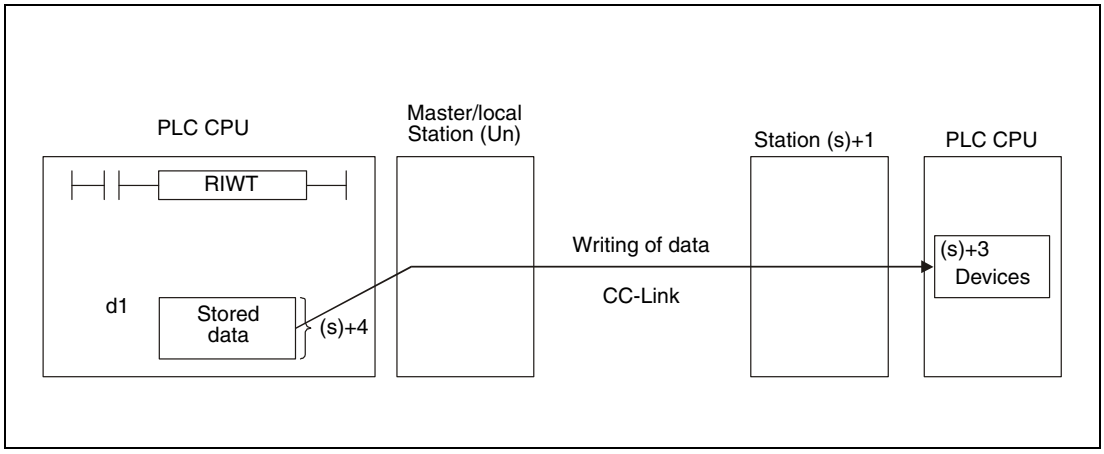
The RIWT instruction writes data to the buffer memory of an intelligent device connected to the CC-Link. When a master module with a software version from J onward or a CC-Link module of the MELSEC System Q is used, it is also possible to write to the PLC CPU device memory of another station connected to the CC-Link network.

The station number of the other station is designated by (s)+1. This station is connected to the master/local station specified at Un. Where the write data are is stored is designated by d1. At (s)+2 a code is stored which specifies whether to write to a buffer memory or to the device memory of a CPU module. The head address of the buffer memory or the head device is designated by (s)+3. The number of data to write is designated by (s)+4.

- Accessing the buffer memory of an CC-Link module



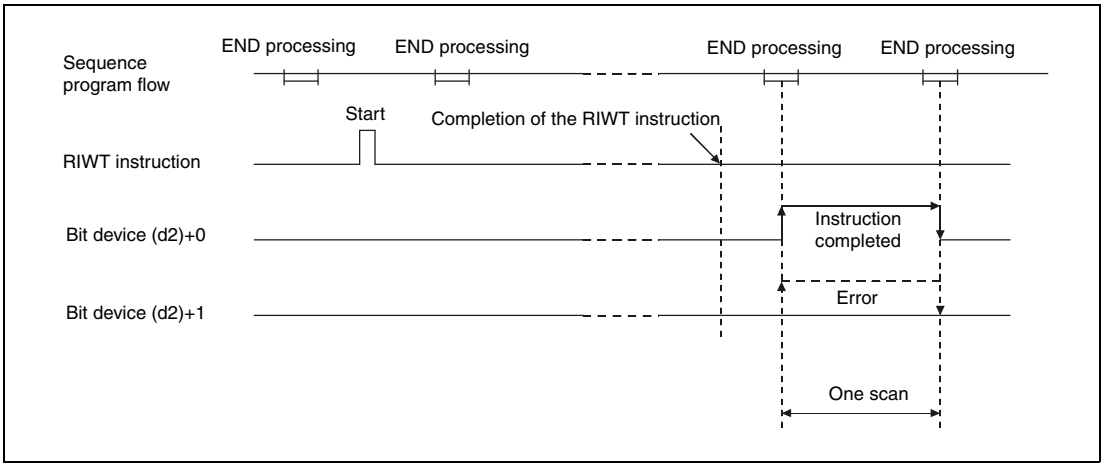
- Accessing the device memory in the PLC CPU of another station on CC-Link



Whether the execution of the RIWT instruction has been finished can be checked with the devices (d2)+0 and (d2)+1:

- The bit device (d2)+0 turns ON at the END processing of the scan in which the RIWT instruction has been completed and turns OFF at the next END processing.
- The bit device (d2)+1 indicates an error during execution of the RIWT instruction. When the instruction has been completed normal, this device stays OFF, but when an error occurs during execution of the RIWT instruction, (d2)+1 turns ON at the END processing of the scan in which the RIWT instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the RIWT instruction is being executed:



Please note, that it's possible to execute RIWT instructions for multiple stations at the same time, but the same intelligent device station or local station cannot be accessed simultaneously from more than one station.

Operation Error

In the following cases an operation error occurs, the error flag SM0 is set, and an error code is stored in SD0:

- When the module specified by Un is not an intelligent function module or a special function module. (Error code: 2112)
- When an attempt was made to execute an unsupported instruction. (Error code: 4002)
- When the number of devices in the instruction is incorrect. (Error code: 4003)
- When the instruction specifies a device that cannot be used. (Error code: 4004)
- When the area designated by s contains data that cannot be used. (Error code: 4100)
- When the number of data set to be used exceeds the allowable range. (Error code: 4101)
- When the storage data or constants of the device specified with the instruction exceeds the allowable range. (Error code: 4101)

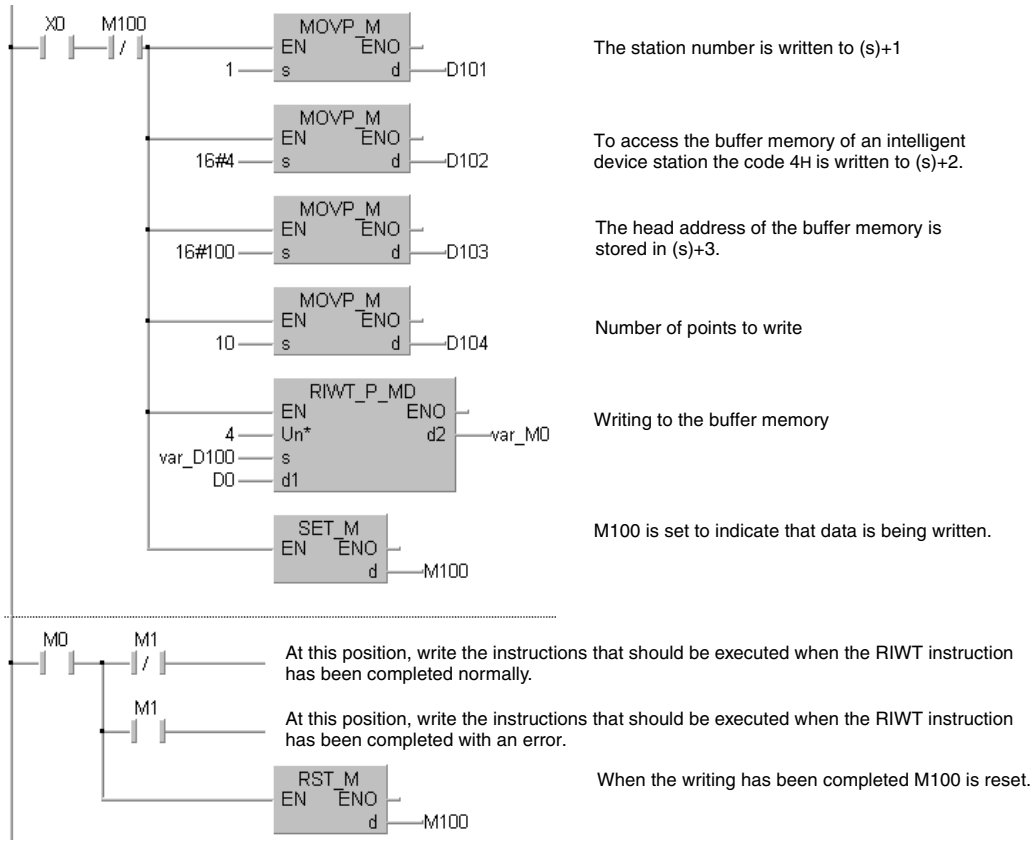
Program Example

RIWT

The following program is processed in the PLC CPU of the master station. When the input X0 is set, the contents of the data registers D0 to D9 is moved to the intelligent device station number 1 and stored to the buffer memory addresses 100H to 109H. The head I/O number of the master module of CC-Link is X/Y40.

- IEC editors
  (On the next page this program example is shown for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



IEC Instruction List

```
LD          X0
ANDN        M100
MOVP_M      1,          D101
MOVP_M      16#4,       D102
MOVP_M      16#100,     D103
MOVP_M      10,         D104
RIWT_P_MD   4,          var_D100,   D0,   var_M0
SET_M       M100
```

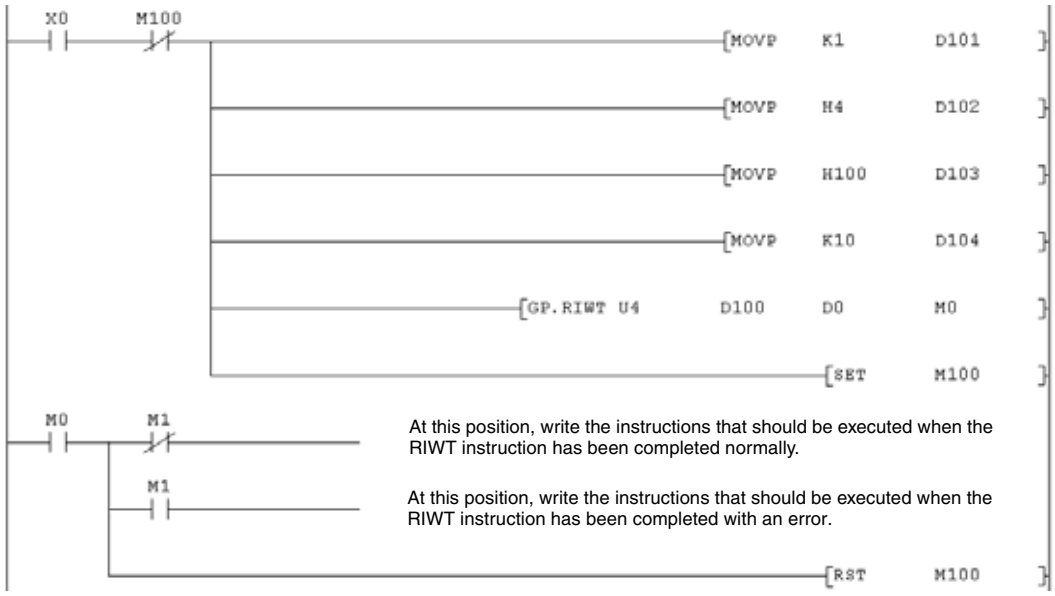
For an explanation of the devices and instructions used please see the above program example.

```
LD          M0
ANDN        M1
An instruction at this position will be executed when the RIWT instruction has been completed normally.
LD          M0
AND         M1
An instruction at this position will be executed when the RIWT instruction has been completed annormally.
LD          M0
RST_M       M100
```

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page

Ladder Diagram (GX Works2)



MELSEC Instruction List

MELSEC	LD	X0	
	ANI	M100	
	MOVP	K1	D101
	MOVP	H4	D102
	MOVP	H100	D103
	MOVP	K10	D104
	GP.RIWT	U4	
		D100	
		D0	
		M0	
	SET	M100	

MELSEC	LD	M0
	MPS	
	ANI	M1
	An instruction at this position will be executed when the RIWT instruction has been completed normally.	
	MRD	
	AND	M1
	An instruction at this position will be executed when the RIWT instruction has been completed annormally.	
	MPP	
	RST	M100

12.5.4 RIRCV

CPU

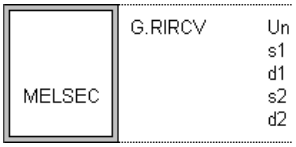
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

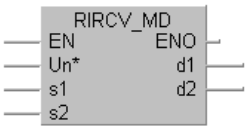
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s1	—	●	●	—	—	—	—	—	—
d1	—	●	●	—	—	—	—	—	—
d2	●	●	●	—	—	—	—	—	—

GX IEC Developer

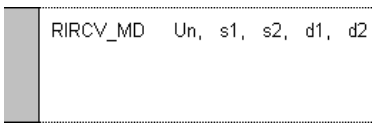
MELSEC Instruction List



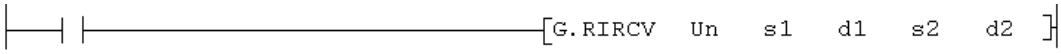
Ladder Diagram



IEC Instruction List



GX Works2



Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>Un</td><td colspan="4">Head I/O address of the CC-Link master module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as 10H)</td><td>0 to FEH</td><td>User</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="7">s1</td><td colspan="4">Head number of the devices where control data for the execution of this instruction is stored.</td><td></td><td></td><td rowspan="7">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s1)+0</td><td>Execution result of the instruction</td><td>Indicates whether an error has occured during execution of the instruction:
0000H: No error
Any value other than 0000H: An error has occured. The stored value is an error code which is explained in the user's manual of the CC-Link module.</td><td>—</td><td>System</td></tr>
<tr><td>(s1)+1</td><td>Station number</td><td>Station number of the intelligent device station where data is read from</td><td>0 to 64</td><td rowspan="4">User</td></tr>
<tr><td>(s1)+2</td><td>Access code</td><td>Enter the value „0004H“ (Read from the buffer memory of an intelligent device station.)</td><td>0004H</td></tr>
<tr><td>(s1)+3</td><td>Head address</td><td>Head address in the buffer memory (Address of the first data to read)</td><td>Depends on the accessed station</td></tr>
<tr><td>(s1)+4</td><td>Number of points to read</td><td>Specify how much data (in the unit „words“) should be read from the intelligent device station. Set a value within the intelligent device station buffer memory capacity and the parameter-set receiving buffer area of the master station.</td><td>1 to 480</td></tr>
<tr><td rowspan="6">s2</td><td colspan="4">Link devices used for handshaking</td><td></td><td rowspan="6">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td rowspan="2">(s2)+0</td><td rowspan="2">Remote output (RY) for data request</td><td>• Higher byte
Set the upper 8 bits to „0“.</td><td>0</td><td rowspan="4">User</td></tr>
<tr><td>• Lower byte
Specify a remote output (RY) of the intelligent device station</td><td>0 to 127</td></tr>
<tr><td rowspan="2">(s2)+1</td><td rowspan="2">Remote register (RWr) used as error code storage device
Remote input (RX) used as completion device.</td><td>• Higher byte
Specify a remote register (RWr) of the intelligent device station, in which the same error code as in (s1)+0 will be stored.</td><td>0 to 15 or FF (When FF is set, no number is specified.)</td></tr>
<tr><td>• Lower byte
Specify a remote input (RX) of the intelligent device station</td><td>0 to 127</td></tr>
<tr><td>(s2)+2</td><td>Completion mode</td><td>Specify, how the completion of the reading process should be indicated:
0: Using 1 device (RXn)
1: Using 2 devices (RXn, RXn+1) (RXn+1 will be set at abnormal completion.)</td><td>0 or 1</td></tr>
<tr><td>d1</td><td colspan="4">Head address of the devices where the read data is to be stored.</td><td>User</td><td>BIN 16-bit</td></tr>
</table>

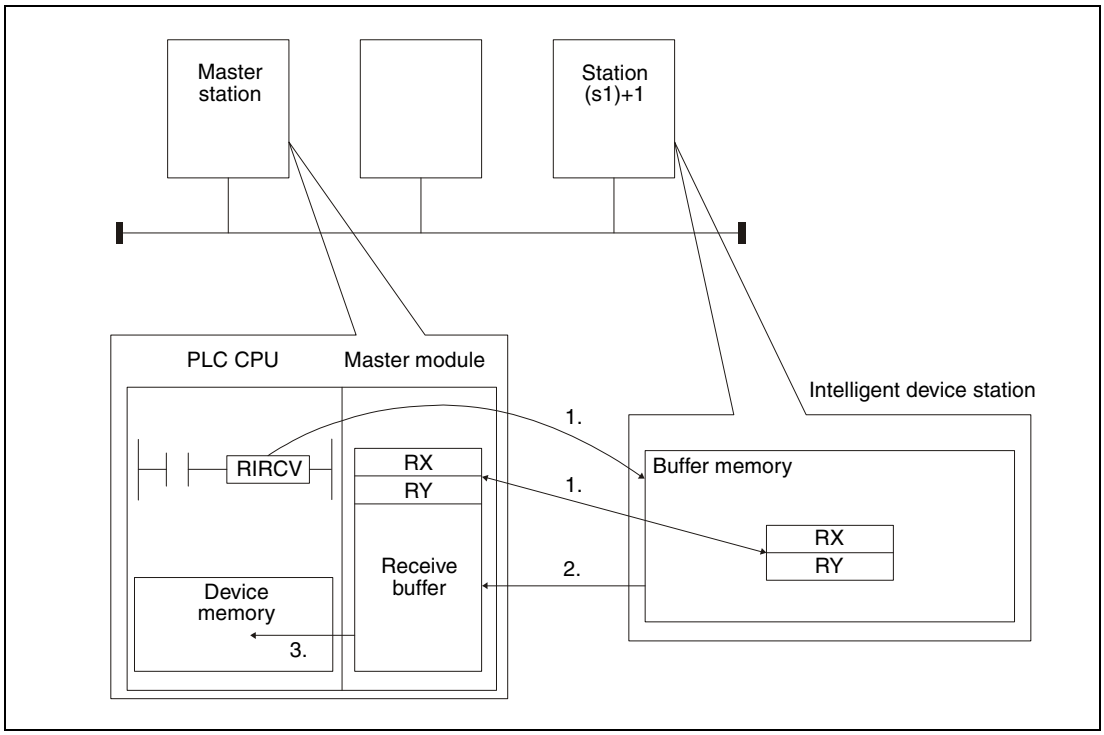
Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="4">d2</td><td colspan="4">Bit device which is set for one scan after completion of the RIRCV instruction.
(d2)+1 indicates that an error has occured during execution of the instruction.</td><td></td><td></td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d2)+0</td><td>Instruction completed</td><td>Indicates the completion of the RIRCV instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d2)+1</td><td>Instruction completed with error</td><td>Indicates that an error has occured during the processing of the RIRCV instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions **Reading of data from the buffer memory of an intelligent device station (with handshake)**

RIRCV Data read (with handshake)

The execution of a RIRCV instruction is only possible in the PLC CPU of the master station. This instruction is used to read data from the buffer memory on an intelligent device station. The data exchange is controlled by handshaking devices:

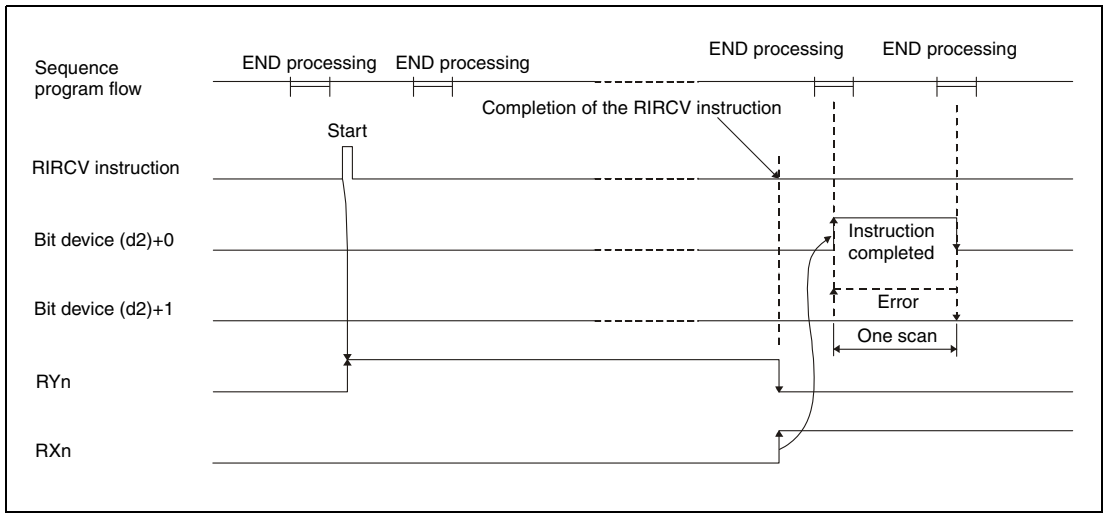


1. The buffer memory address specified by (s1)+3 of the station specified by (s1)+1 is accessed. The devices specified in s2 are used for the handshake.
2. The contents of the number of buffer memory addresses specified in (s1)+4 is read to the receive buffer of the master module.
3. The read data is stored in the PLC CPU to the devices starting with the one specified in d1. After that, the bit device specified in (d2)+0 is set for one scan.

Whether the execution of the RIRCV instruction has been finished can be checked with the devices (d2)+0 and (d2)+1:

- The bit device (d2)+0 turns ON at the END processing of the scan in which the RIRCV instruction has been completed and turns OFF at the next END processing.
- The bit device (d2)+1 indicates an error during execution of the RIRCV instruction. When the instruction has been completed normal, this device stays OFF. When an error occurs during execution of the RIRCV instruction, (d2)+1 turns ON at the END processing of the scan in which the RIRCV instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the RIRCV instruction is being executed:



Although it's possible to execute RIRCV instructions for multiple intelligent device stations at the same time, it's not possible to access the same intelligent device station simultaneously from more than one station.

Operation Error

In the following cases an operation error occurs, the error flag SM0 is set, and an error code is stored in SD0:

- When the module specified by Un is not an intelligent function module or a special function module. (Error code: 2112)
- When an attempt was made to execute an unsupported instruction. (Error code: 4002)
- When the number of devices in the instruction is incorrect. (Error code: 4003)
- When the instruction specifies a device that cannot be used. (Error code: 4004)
- When the area designated by s contains data that cannot be used. (Error code: 4100)
- When the number of data set to be used exceeds the allowable range. (Error code: 4101)
- When the storage data or constants of the device specified with the instruction exceeds the allowable range. (Error code: 4101)

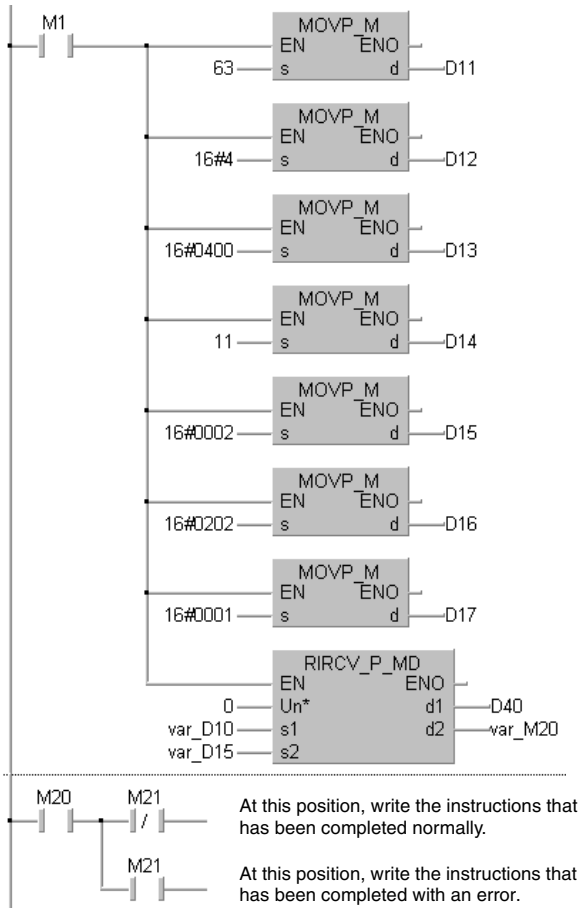
Program Example

RIRCV

The following program is executed in the PLC CPU of the master station. When M1 is set, the contents of 11 buffer memory addresses is read from the intelligent device station with the station number 63. Reading starts at the buffer memory address 400H. The data will be stored in the CPU module from data register D40 onward. To the master module of CC-Link the head I/O number X/Y00 is assigned. The remote devices RX2, RY2 and RWr2 are used for handshake. The completion of the reading is indicated by two devices. ((s2)+2 is set to „1“.)

- IEC editors
  (On the next page this program example is shown for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



Read from station number 63 (63 -> (s1)+1)

The access code 4H for access to the buffer memory of an intelligent device station is stored in (s1)+2.

In (s)+3 the head address in the buffer memory (400H) is stored.

Read 11 buffer memory addresses

RY2 is entered as handshake device in (s2)+1

RWr2 and RX2 are set as handshake devices in (s2)+2

Two devices are used to indicate the completion of the reading.

Reading of the buffer memory

At this position, write the instructions that should be executed when the RIRCV instruction has been completed normally.

At this position, write the instructions that should be executed when the RIRCV instruction has been completed with an error.

IEC Instruction List

```
LD          M1
MOVP_M      63,       D11
MOVP_M      16#0004,  D12
MOVP_M      16#0400,  D13
MOVP_M      11,       D14
MOVP_M      16#0002,  D15
MOVP_M      16#0202,  D16
MOVP_M      16#0001,  D17
RIRCV_P_MD  0,        var_D10,  var_d15,  D40,  var_M20
```

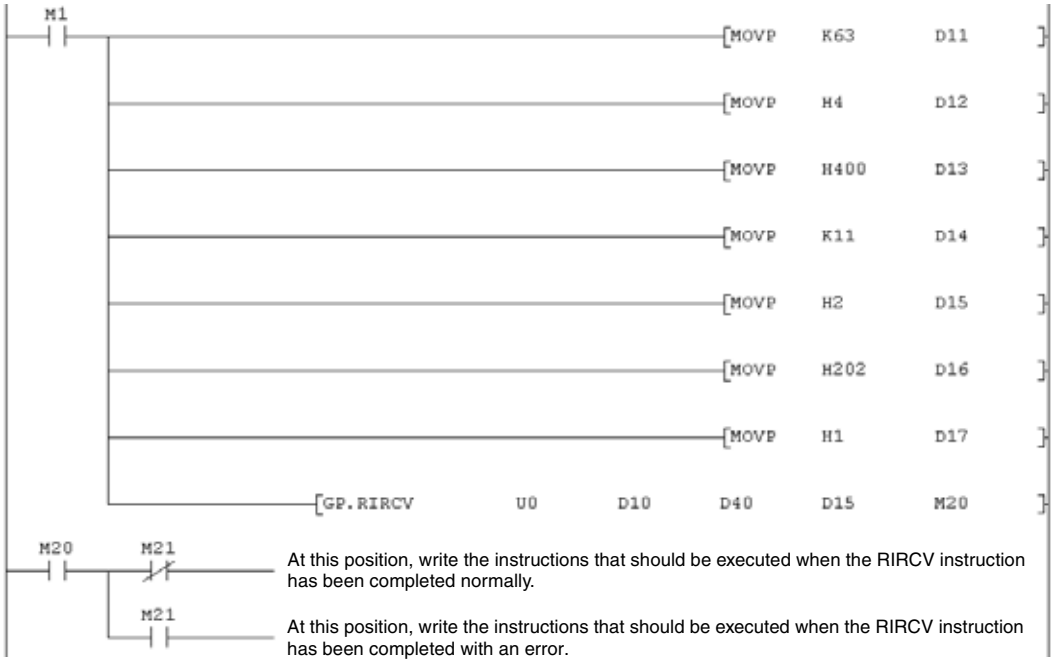
For an explanation of the devices and instructions used please see the above program example.

```
LD          M20
ANDN        M21
```
An instruction at this position will be executed when the RIRCV instruction has been completed normally.
```
LD          M20
AND         M21
```
An instruction at this position will be executed when the RIRCV instruction has been completed annormally

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



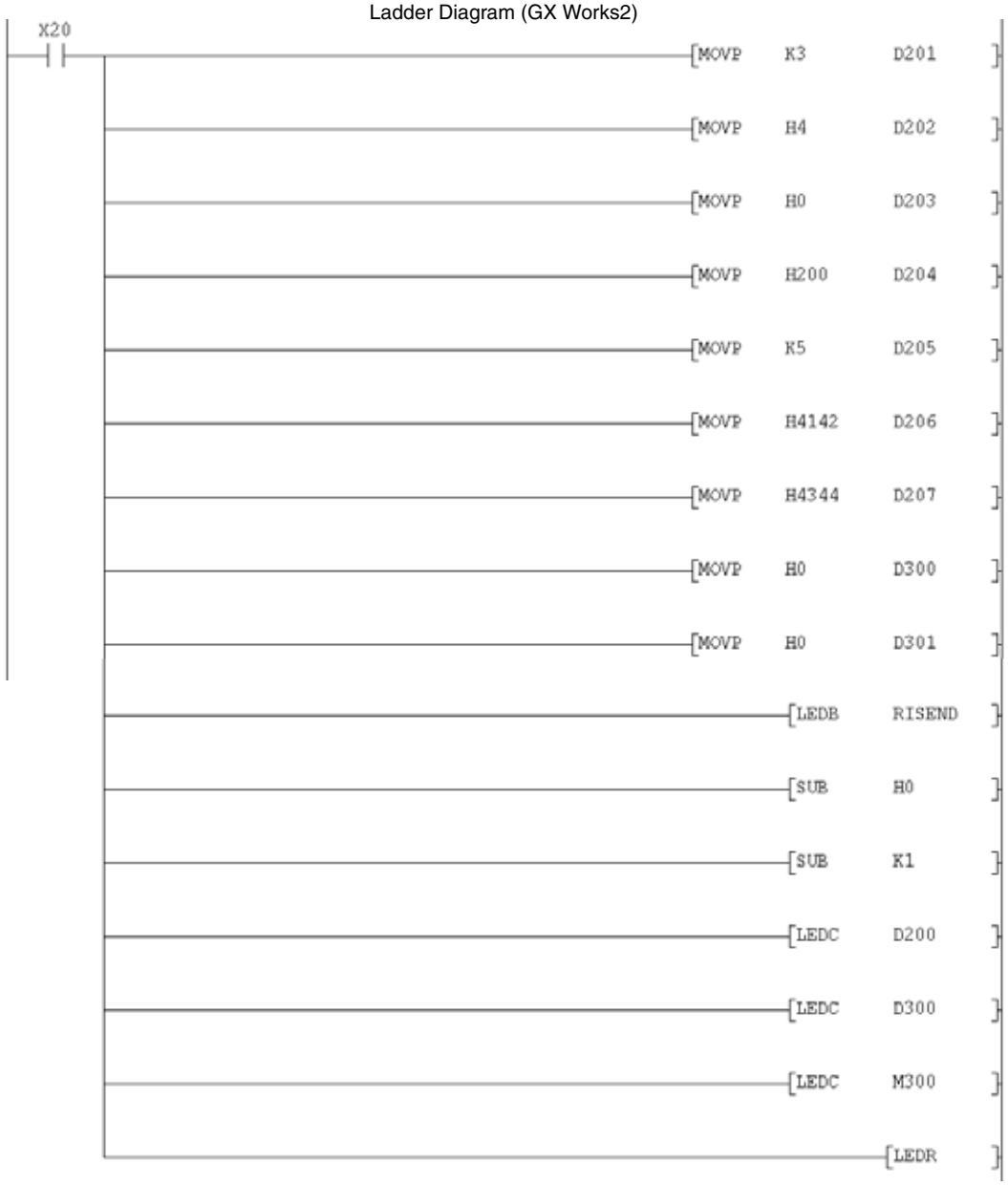
MELSEC Instruction List

MELSEC	LD	M1	
	MOVP	K63	D11
	MOVP	H0004	D12
	MOVP	H0400	D13
	MOVP	K11	D14
	MOVP	H0002	D15
	MOVP	H0202	D16
	MOVP	H0001	D17
	GP.RIRCV	U0	
		D10	
		D40	
		D15	
		M20	

MELSEC	LD	M20
	MPS	
	ANI	M21
	An instruction at this position will be executed when the RIRCV instruction has been completed normally.	
	MRD	
	AND	M21
	An instruction at this position will be executed when the RIRCV instruction has been completed annormally.	

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



MELSEC Instruction List

MELSEC

```
LD      X20
MOVP    K3      D201
MOVP    H0004   D202
MOVP    H0      D203
MOVP    H0200   D204
MOVP    K5      D205
MOVP    H4142   D206
MOVP    H4344   D207
MOVP    H0      D300
MOVP    H0      D301
LEDB    RISEND
SUB     H0
SUB     K1
LEDC    D200
LEDC    D300
LEDC    M300
LEDR
```

For further information about the programming of dedicated instructions using the MELSEC editors please refer to chapter 3.3 of this manual.

12.5.5 RISEND

CPU

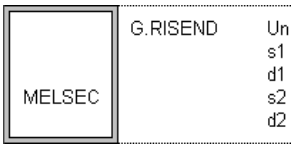
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

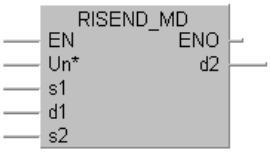
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
s1	—	●	●	—	—	—	—	—	—
s2	—	●	●	—	—	—	—	—	—
d1	—	●	●	—	—	—	—	—	—
d2	●	●	●	—	—	—	—	—	—

GX IEC Developer

MELSEC Instruction List



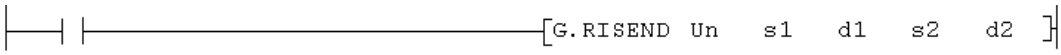
Ladder Diagram



IEC Instruction List

```
RISEND_MD   Un, s1, d1, s2, d2
```

GX Works2



Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td>Un</td><td colspan="4">Head I/O address of the CC-Link master module
(The upper two digits of an address expressed as a 3-digit number, e. g. the head address X/Y100 is set as 10H)</td><td>0 to FEH</td><td>User</td><td>BIN 16-bit</td></tr>
<tr><td rowspan="7">s1</td><td colspan="4">Head number of the devices where control data for the execution of this instruction is stored.</td><td></td><td></td><td rowspan="7">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(s1)+0</td><td>Execution result of the instruction</td><td>Indicates whether an error has occured during execution of the instruction:
0000H: No error
Any value other than 0000H: An error has occured. The stored value is an error code which is explained in the user's manual of the CC-Link module.</td><td>—</td><td>System</td></tr>
<tr><td>(s1)+1</td><td>Station number</td><td>Station number of the intelligent device station where the data is send to</td><td>0 to 64</td><td rowspan="4">User</td></tr>
<tr><td>(s1)+2</td><td>Access code</td><td>Enter the value „0004H“ (Write to the buffer memory of an intelligent device station.)</td><td>0004H</td></tr>
<tr><td>(s1)+3</td><td>Head address</td><td>Head address in the buffer memory (First address where data is written to)</td><td>Depends on the accessed station</td></tr>
<tr><td>(s1)+4</td><td>Number of points to write</td><td>Specify how much data (in the unit „words“) should be written to the intelligent device station.</td><td>1 to 480</td></tr>
<tr><td rowspan="6">s2</td><td colspan="4">Link devices used for handshaking</td><td></td><td rowspan="6">BIN 16-bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td rowspan="2">(s2)+0</td><td rowspan="4">Remote output (RY) to request the sending of data
Remote register (RWr) used as error code storage device
Remote input (RX) used as completion device.</td><td>• Higher byte
Set the upper 8 bits to „0“.</td><td>0</td><td rowspan="4">User</td></tr>
<tr><td>• Lower byte
Specify a remote output (RY) of the intelligent device station</td><td>0 to 127</td></tr>
<tr><td rowspan="2">(s2)+1</td><td>• Higher byte
Specify a remote register (RWr) of the intelligent device station, in which the same error code as in (s1)+0 will be stored.</td><td>0 to 15 or FF (When FF is set, no number is specified.)</td></tr>
<tr><td>• Lower byte
Specify a remote input (RX) of the intelligent device station</td><td>0 to 127</td></tr>
<tr><td>(s2)+2</td><td>Completion mode</td><td>Specify, how the completion of the reading process should be indicated:
0: Using 1 device (RXn)
1: Using 2 devices (RXn, RXn+1) (RXn+1 will be set at abnormal completion.)</td><td>0 or 1</td></tr>
<tr><td>d1</td><td colspan="4">First address of the area where the data for the intelligent device station is stored</td><td>User</td><td>BIN 16-bit</td></tr>
</table>

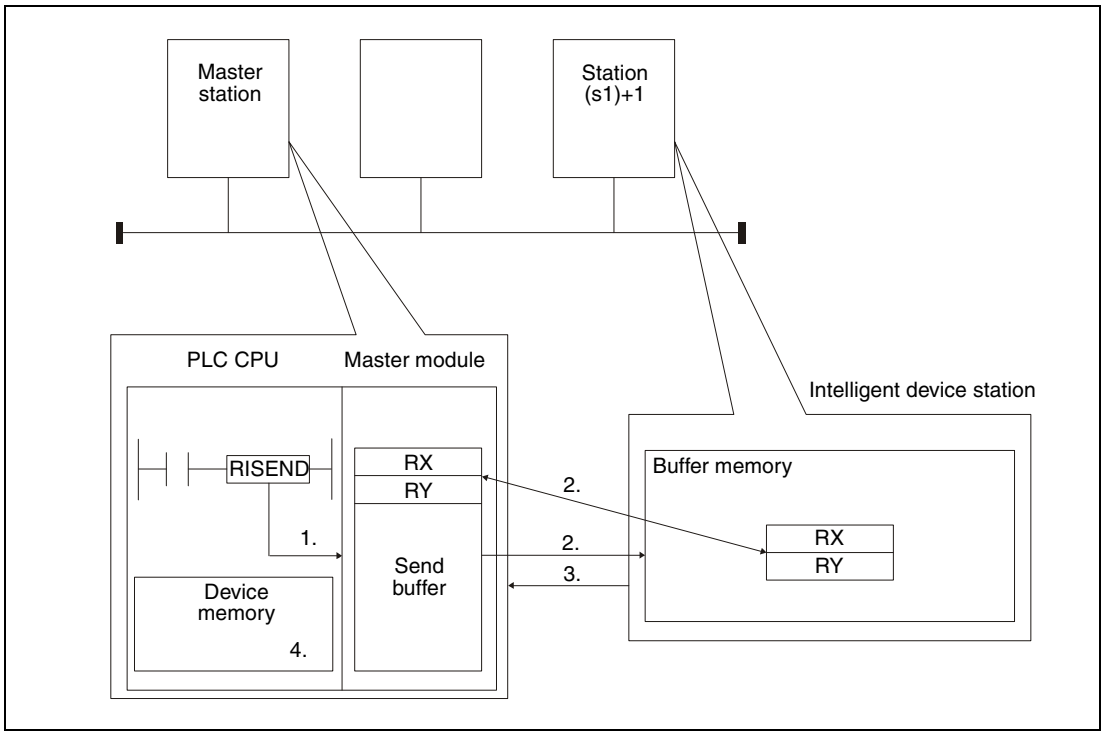
Variables

<table>
<tr><th>Set Data</th><th colspan="4">Meaning</th><th>Range</th><th>Contents is stored by</th><th>Data Type</th></tr>
<tr><td rowspan="4">d2</td><td colspan="4">Bit device which is set for one scan after completion of the RIRCV instruction.
(d2)+1 indicates that an error has occured during execution of the instruction.</td><td></td><td></td><td rowspan="4">Bit</td></tr>
<tr><td>Set Data</td><td>Meaning</td><td>Description</td><td>Range</td><td>Contents is stored by</td></tr>
<tr><td>(d2)+0</td><td>Instruction completed</td><td>Indicates the completion of the RISEND instruction
ON: Instruction completed
OFF: Instruction not completed</td><td>—</td><td rowspan="2">System</td></tr>
<tr><td>(d2)+1</td><td>Instruction completed with error</td><td>Indicates that an error has occured during the processing of the RIRCV instruction
ON: Abnormal completion
OFF: Normal completion</td><td>—</td></tr>
</table>

Functions **Write (with handshake) to the buffer memory of an intelligent decive station**

RISEND Sending of data (with handshake)

The RIRCV instruction can only be performed in the PLC CPU of the master station and is used to write data to the buffer memory on an intelligent device station. The data exchange is controlled by handshaking devices:

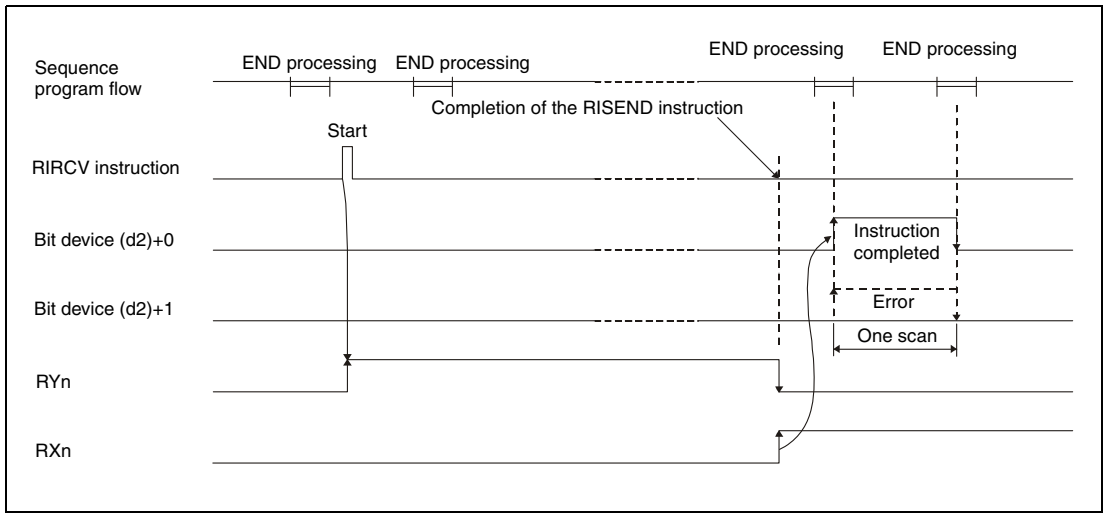


1. The data for the intelligent device station is moved to the send buffer of the master station.
2. The data is written to the buffer memory address specified by (s1)+3 of the station specified by (s1)+1. The devices specified in s2 are used for the handshake.
3. A write complete response is send to the master station.
4. The device specified in (d2)+0 is set.

Whether the execution of the RISEND instruction has been finished can be checked with the devices (d2)+0 and (d2)+1:

- The bit device (d2)+0 turns ON at the END processing of the scan in which the RISEND instruction has been completed and turns OFF at the next END processing.
- The bit device (d2)+1 indicates an error during execution of the RISEND instruction. When the instruction has been completed normal, this device stays OFF. When an error occurs during execution of the RISEND instruction, (d2)+1 turns ON at the END processing of the scan in which the RISEND instruction has been completed and turns OFF at the next END processing.

The following figure shows the timing when the RIRCV instruction is being executed:



Although it's possible to execute RISEND instructions for multiple intelligent device stations at the same time, it's not possible to access the same intelligent device station simultaneously from more than one station.

Operation Error

In the following cases an operation error occurs, the error flag SM0 is set, and an error code is stored in SD0:

- When the module specified by Un is not an intelligent function module or a special function module. (Error code: 2112)
- When an attempt was made to execute an unsupported instruction. (Error code: 4002)
- When the number of devices in the instruction is incorrect. (Error code: 4003)
- When the instruction specifies a device that cannot be used. (Error code: 4004)
- When the area designated by s contains data that cannot be used. (Error code: 4100)
- When the number of data set to be used exceeds the allowable range. (Error code: 4101)
- When the storage data or constants of the device specified with the instruction exceeds the allowable range. (Error code: 4101)

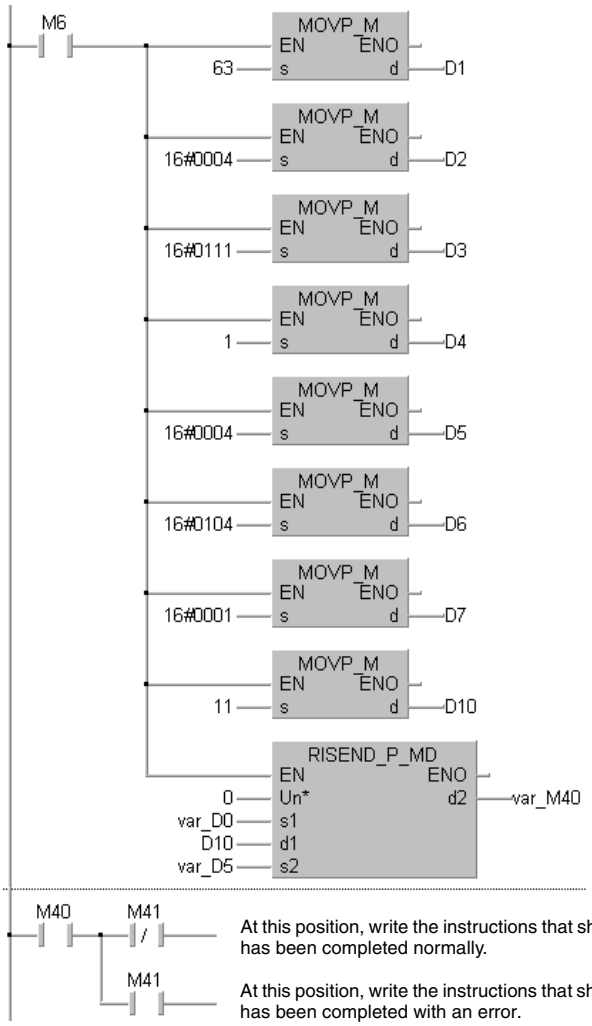
Program Example

RISEND

The following program, which is executed in the PLC CPU of the master station, writes 1 word of data to the buffer memory address 111H of the intelligent device station with the station number 63. To the master module of CC-Link, the head I/O number X/Y000 is assigned. The devices RX4, RY4 and RWr4 are used for handshaking. The completion of the reading is indicated by two devices. ((s2)+2 is set to „1“.)

- IEC editors (On the next page this program example is shown for the MELSEC instruction list and the ladder diagram of the GX Works2.)

Ladder Diagram (GX IEC Developer)



Write to station number 63 (63 -> (s1)+1)

The access code 4H for access to the buffer memory of an intelligent device station is stored in (s1)+2

The head address in the buffer memory (111H) is stored to (s)+3.

Write to only buffer memory address

RY4 is entered as handshake device in (s2)+1

RWr1 and RX4 are set as handshake devices in (s2)+2

Two devices are used to indicate the completion of the reading.

The value „11“ is stored in D10. The contents of D10 is moved to the buffer memory address 111H.

Write to buffer memory

At this position, write the instructions that should be executed when the RISEND instruction has been completed normally.

At this position, write the instructions that should be executed when the RISEND instruction has been completed with an error.

IEC Instruction List

```
LD              M6
MOVP_M          63,         D1
MOVP_M          16#0004,    D2
MOVP_M          16#0111,    D3
MOVP_M          1,          D4
MOVP_M          16#0004,    D5
MOVP_M          16#0104,    D6
MOVP_M          16#0001,    D7
MOVP_M          11,         D10
RISEND_P_MD     0,          var_D0,  D10,  var_D5,  var_M40
```

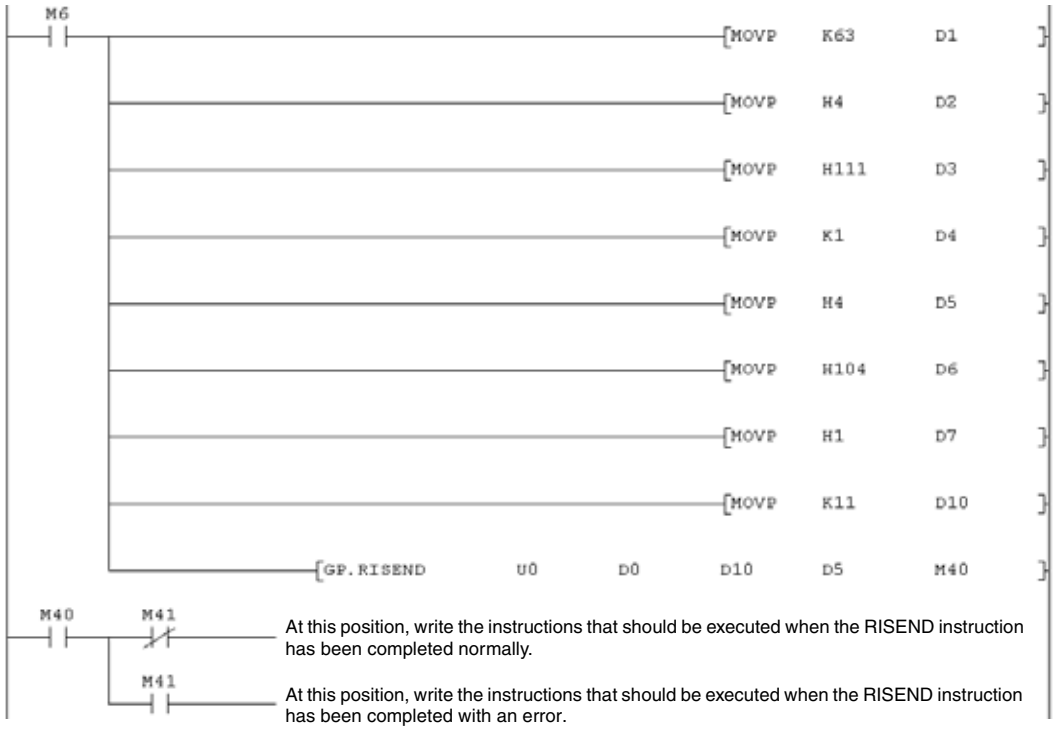
For an explanation of the devices and instructions used please see the above program example.

```
LD              M40
ANDN            M41
```
An instruction at this position will be executed when the RISEND instruction has been completed normally.
```
LD              M40
AND             M41
```
An instruction at this position will be executed when the RISEND instruction has been completed annormally

NOTE *For the IEC editors it is neccessary to define the variables in the header of the program organization unit (POU). Without variable definition it would cause compiler or checker error messages. For details see section 3.5.2 "Addressing of arrays and registers in the GX IEC Developer" of this manual.*

- MELSEC instruction list and ladder diagram of the GX Works2
  For explanation of the devices and instructions used please see the program example for the ladder diagram of the GX IEC Developer on the previous page.

Ladder Diagram (GX Works2)



MELSEC Instruction List

MELSEC	LD	M6	
	MOVP	K63	D1
	MOVP	H0004	D2
	MOVP	H0111	D3
	MOVP	K1	D4
	MOVP	H0004	D5
	MOVP	H0104	D6
	MOVP	H0001	D7
	MOVP	K11	D10
	GP.RISEND	U0	
		D0	
		D10	
		D5	
		M40	

```
LD   M40
MPS
ANI  M41
An instruction at this position will be executed when the RISEND instruction has been completed normally.
MPP
AND  M41
An instruction at this position will be executed when the RISEND instruction has been completed annormally.
```

12.5.6 RITO

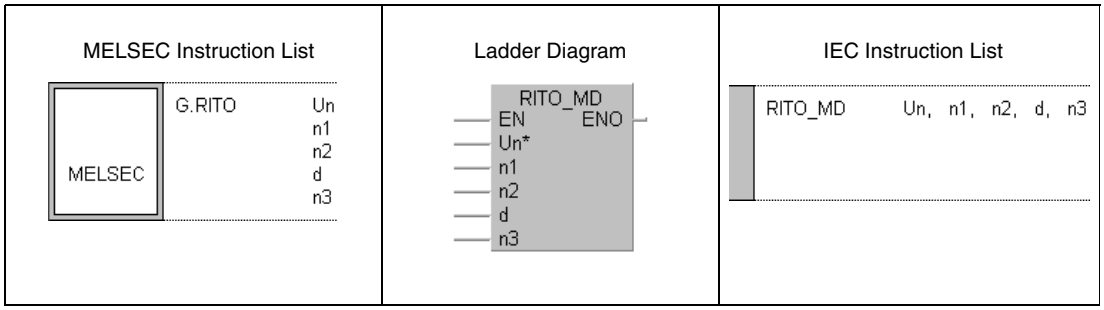
CPU

Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

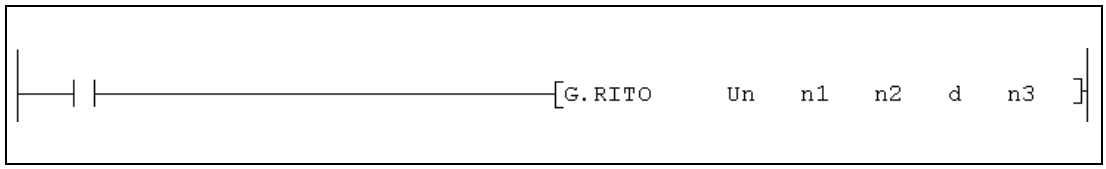
Devices MELSEC Q

	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
n1	●	●	●	—	—	—	—	●	—
n2	●	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—
n3	●	●	●	—	—	—	—	●	—

GX IEC Developer



GX Works2



Variables

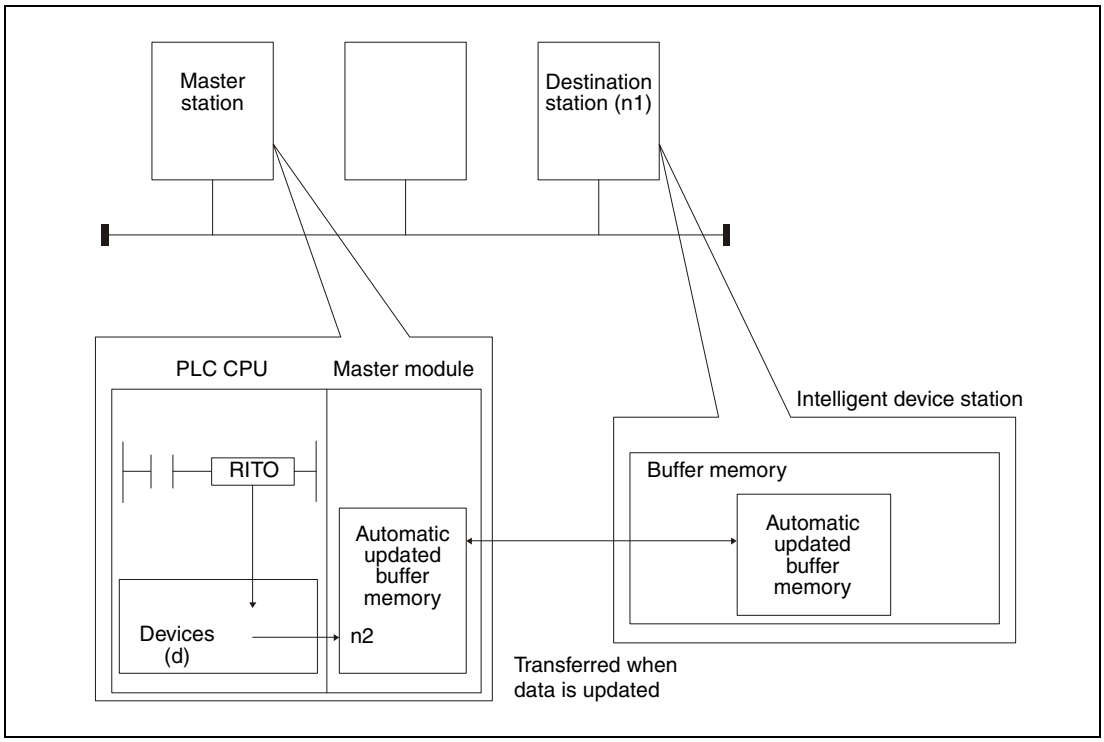
Set Data	Meaning	Range	Contents is stored by	Data Type
Un	Head I/O address of the CC-Link master module (Only the upper two digits of an address expressed as a 3-digit number are specified, e. g. the head address X/Y100 is set as 10H)	0 to FEH	User	BIN 16-bit
n1	Write destination • Specify the station number of the intelligent device station where data is written to. • Specify „FFH“ when data is to be moved to the random access buffer.	1 to 64 or FFH		
n2	The offset value of the automatic updated buffer of the intelligent device station specified by the master station or the random access buffer. The head address to write to is designated relative to the head address of the automatic updated buffer. An example: To write data to the address 356H of the buffer memory, which starts at address 350H, the value 6H must be specified at n2.	Between 0 and the max. value set in the parameters.		
d	First address of the area where the write data is stored.	Within the range of the specified device		Address
n3	Number of points to write (unit: words)	1 to 4096		BIN 16-bit

Functions **Write to automatic updating buffer memory**

RITO Data write

The RITO instruction moves data from the device memory of the PLC CPU to the automatic updating buffer memory in the master station. The data is than transferred to another station on CC-Link.
The data is specified by the head address (d) and the number of words (n3). The destination in the master-station is designated by n1 (equals the station number of the station where the data is finally send to) and n2 (head address of the automatic updating buffer memory in the master station). The head I/O number of the master station is specified in Un.

The function of the RITO instruction is explained in the following figure:



The RITO instruction cannot be executed at more than one station for the same intelligent device station.

Up to 4096 words may be written by the RITO instruction.

The assignment of the automatic updated buffers is performed using the „station information settings“ of the network parameters of the GX Works2 or GX IEC Developer.

Operation Error

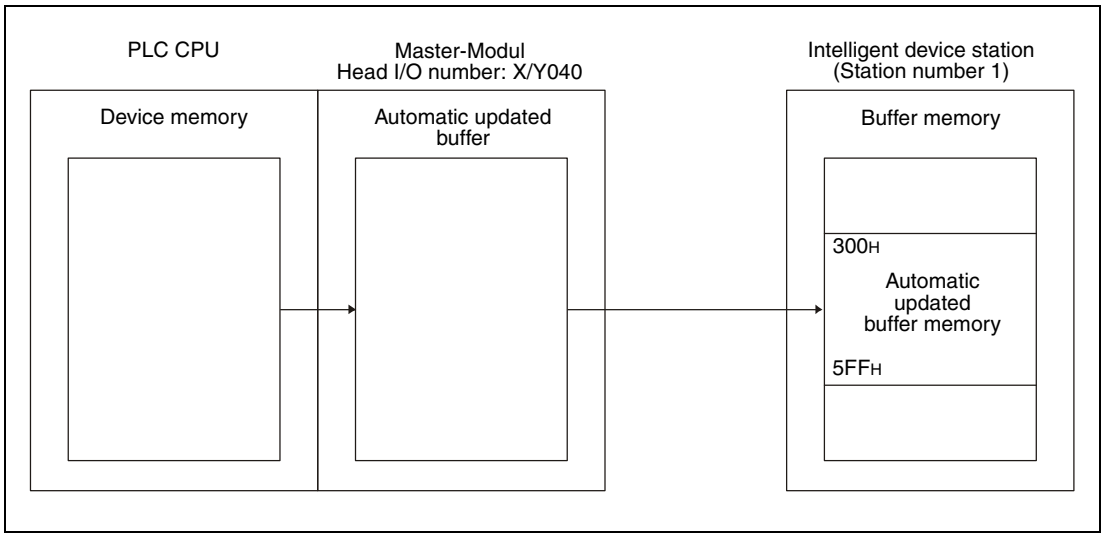
In the following cases an operation error occurs, the error flag SM0 is set, and an error code is stored in SD0:

- When the module specified by Un is not an intelligent function module or a special function module. (Error code: 2112)
- When an attempt was made to execute an unsupported instruction. (Error code: 4002)
- When the number of devices in the instruction is incorrect. (Error code: 4003)
- When the instruction specifies a device that cannot be used. (Error code: 4004)
- When the station number specified at n1 does not exist. (Error code: 4100)
- When the number of words to write specified in n3 is outside of the setting range. (Error code: 4100)

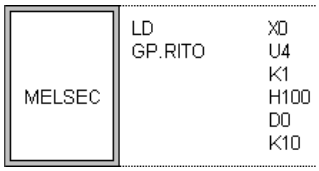
Program Example

RITO

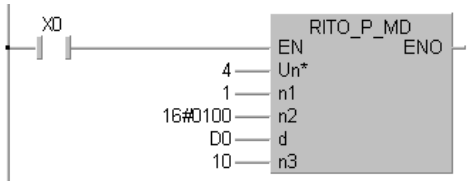
When the input X0 is set, the contents of 10 data registers (D0 to D10) is moved to the automatic updated buffer memory for the station set to station number 1 in the master module. This buffer begins at the address 300H. The data is stored from address 400H onward (offset = 100).



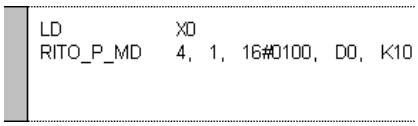
MELSEC Instruction List



Ladder Diagram (GX IEC Developer)



IEC Instruction List



Ladder Diagram (GX Works2)

12.5.7 RIFR

CPU

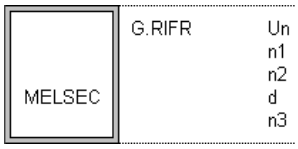
Basic	High Performance	Process	Redundant	Universal	LCPU
●	●	●	●	●	

Devices MELSEC Q

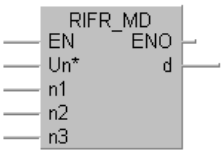
	Usable Devices								
	Internal Devices (System, User)		File-Register	MELSECNET/H Direct J□\□		Special Function Module U□\G□	Index Register Zn	Constants K, H (16#)	Other
	Bit	Word		Bit	Word				
n1	●	●	●	—	—	—	—	●	—
n2	●	●	●	—	—	—	—	●	—
n3	●	●	●	—	—	—	—	●	—
d	—	●	●	—	—	—	—	—	—

GX IEC Developer

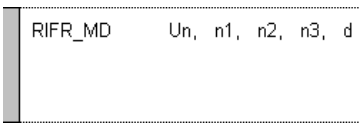
MELSEC Instruction List



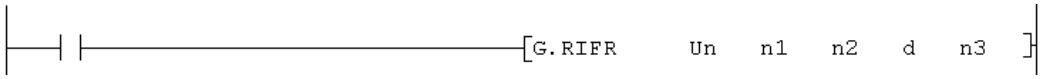
Ladder Diagram



IEC Instruction List



GX Works2



Variables

Set Data	Meaning	Range	Contents is stored by	Data Type
Un	Head I/O address of the CC-Link master module (Only the upper two digits of an address expressed as a 3-digit number are specified, e. g. the head address X/Y100 is set as 10H)	0 to FEH	User	BIN 16-bit
n1	Source of the data • Specify the station number of the intelligent device station where data is read from. • Specify „FFH“, when data is to be read from the random access buffer.	1 to 64 or FFH		
n2	The offset value of the automatic updated buffer of the intelligent device station specified by the master station or the random access buffer. The head address for the data to read is designated relative to the head address of the automatic updated buffer. An example: When reading should start at the address 356H of the buffer memory, which starts at address 350H, the value 6H must be specified at n2.	Between 0 and the max. value set in the parameters		
n3	Number of points to read (unit: words)	1 to 4096		
d	First address of the area where the read data will be stored.	Within the range of the specified device		Address

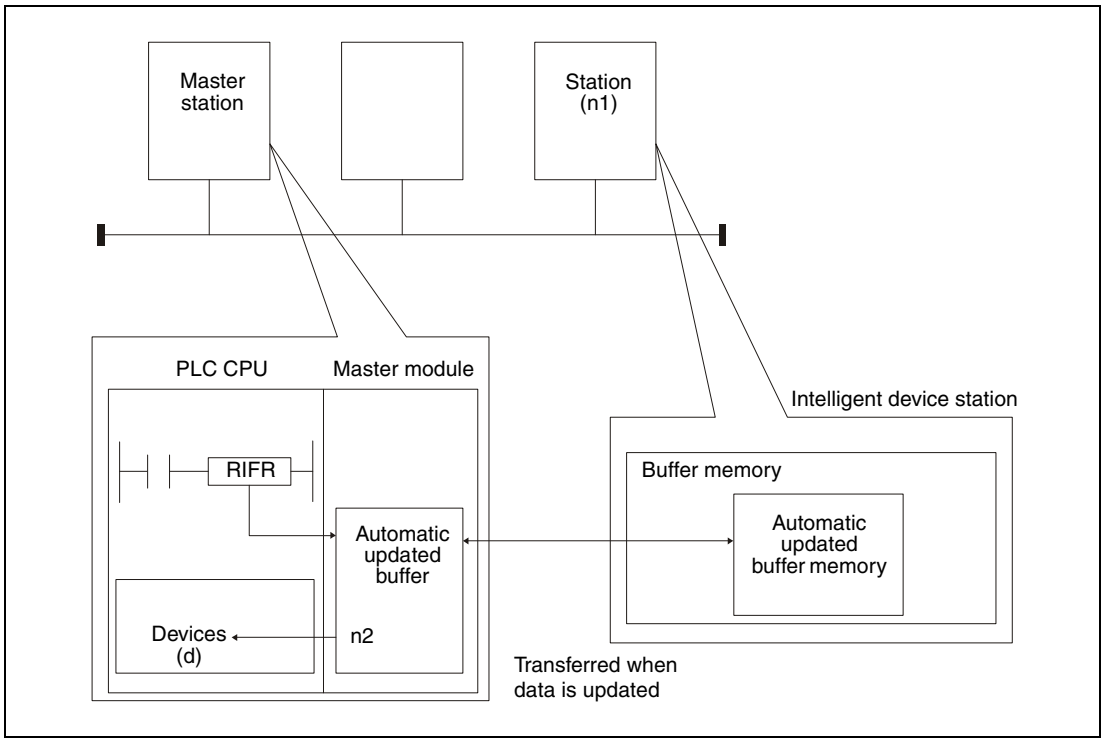
Functions

Read from to automatic updating buffer memory

RIFR Data read

The RIFR instruction moves data from the automatic updating buffer memory in the master station to the device memory of the PLC CPU. The storage area for this data is specified by the head address (d) and the number of words (n3). The source of the data is designated by the station number entered in n1 and the offset for the automatic updating buffer memory of the master station (n2). The head I/O number of the master station is specified in Un.

The function of the RIFR instruction is explained in the following figure:



The RIFR instruction cannot be executed at more than one station for the same intelligent device station.

Up to 4096 words may be read by the RIFR instruction.

The assignment of the automatic updated buffers is performed using the „station information settings“ of the network parameters of the GX Works2 or GX IEC Developer.

Operation Error

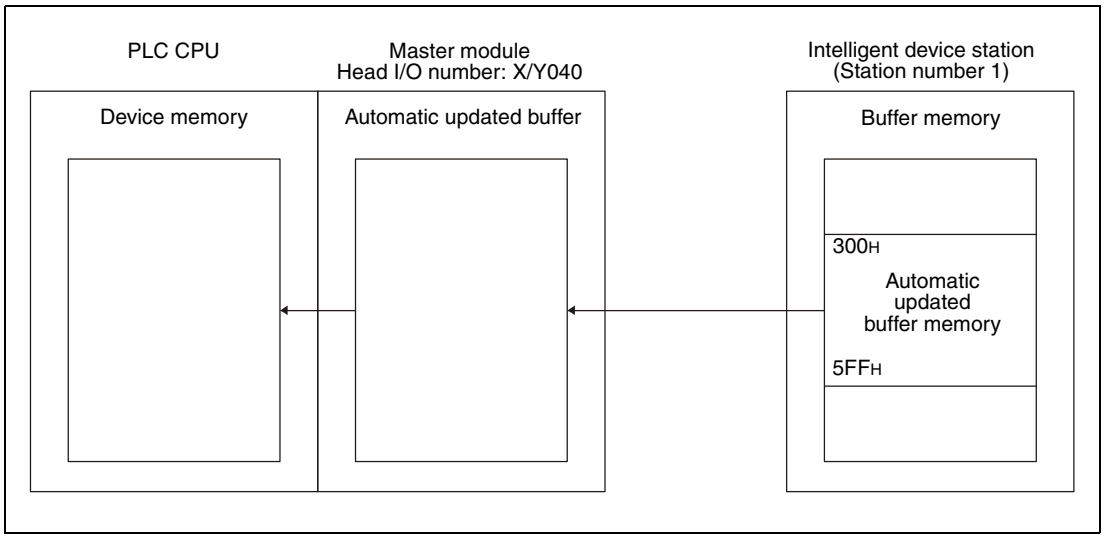
In the following cases an operation error occurs, the error flag SM0 is set, and an error code is stored in SD0:

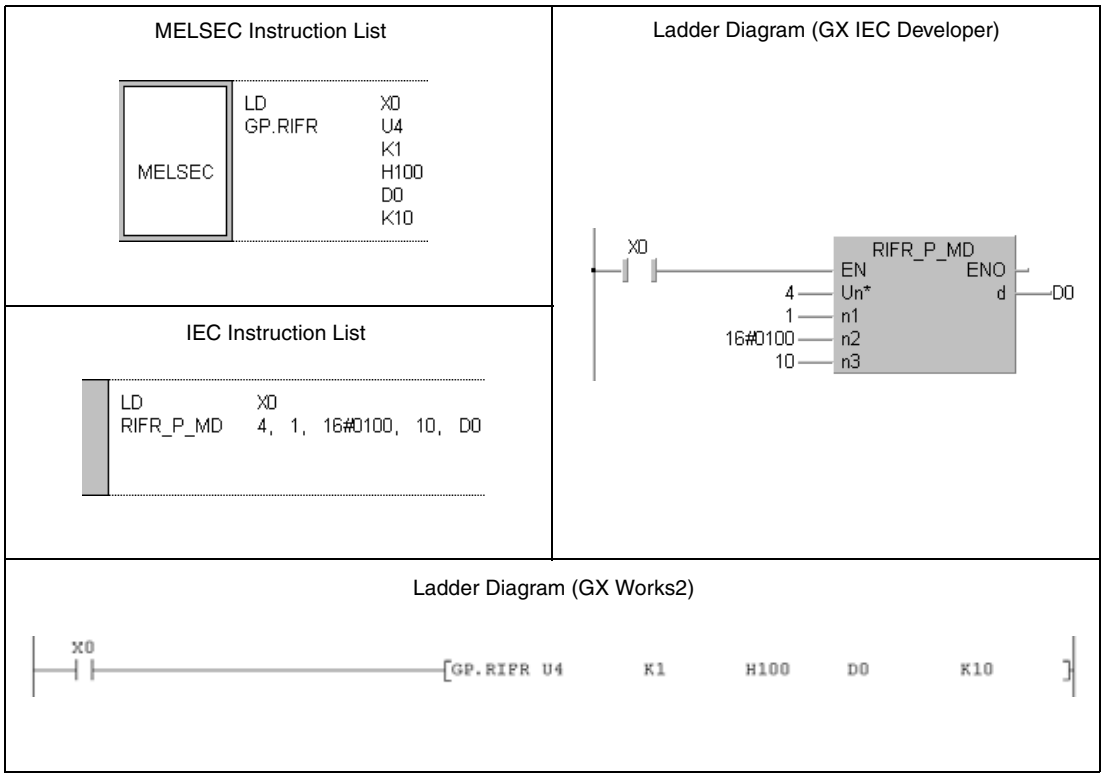
- When the module specified by Un is not an intelligent function module or a special function module. (Error code: 2112)
- When an attempt was made to execute an unsupported instruction. (Error code: 4002)
- When the number of devices in the instruction is incorrect. (Error code: 4003)
- When the instruction specifies a device that cannot be used. (Error code: 4004)
- When the station number specified at n1 does not exist. (Error code: 4100)
- When the number of words to read specified in n3 is outside of the setting range. (Error code: 4100)

Program Example

RIFR

When the input X0 is set, the following program reads the contents of 10 points of the automatic updated buffer set to station number 1 in the master module and stores this data in the PLC CPU to D0 and the successive registers. The automatic updated buffer begins at the address 300H. Reading starts at the address 400H (offset = 100). The master module of CC-Link is allocated to the I/O numbers X/Y040 to X/Y41F.





13 Error Codes

13.1 Error code list

If an error occurs when the PLC is turned ON, set into RUN mode, or during operation, the self-diagnostic functions of the CPU returns an error (LED indication or message on LED display) and store the error information in special relays (SM) and special registers (SD).

When an error occurs at communication request from a programming tool, intelligent function module, or network system to the CPU module, the CPU module returns the error code (4000H to 4FFFH) to the request source.

This section describes errors that may occur in the CPU module and corrective actions for the errors.

13.1.1 How to read the error code list

The following describes how to read section 13.2 "Error code list (1000 to 1999)" to section 13.8 "Error code list (7000 to 10000)". The list contains errors in QCPU and LCPU.

- Error code, common information, and individual information
  The error code is stored in SD0. The common information is stored in SD5 to SD15. The individual information is stored in SD16 to SD26.
- Corresponding CPU
  - QCPU: All the System Q series CPU modules
  - Q00J/Q00/Q01: Basic model QCPU
  - Qn(H): High Performance model QCPU
  - QnPH: Process CPU
  - QnPRH: Redundant CPU
  - QnU: Universal model QCPU
  - Q00UJ/Q00U/Q01U: Q00UJCPU, Q00UCPU, and Q01UCPU
  - LCPU: All the L series CPU modules
  - CPU module model: Only the specified model (Example: Q02UCPU, L26CPU-BT)

13.1.2 Types of error codes

There are two types of errors: errors detected by the self-diagnostic function of the CPU module and errors detected during communication with the CPU module.

The relation between the error detection pattern, error detection location and error code is shown in the following table.

Error detection pattern	Error detection location	Error code	Reference
Detection by the self diagnostics function of CPU module	CPU module	1000 to 1299 [1)]	Sections 13.2 to 13.8
		1300 to 10000 [2)]	
Detection at communication with CPU module	CPU module	4000H to 4FFFH	Section 13.9
	Serial communication module	7000H to 7FFFH	Manual of corresponding module
	CC-Link module (including built-in CC-Link function module)	B000H to BFFFH	
	ETHERNET module (including built-in Ethernet function module)	C000H to CFFFH	
	CC-Link IE field network module	D000H to DFFFH	
	CC-Link IE controller network	E000H to EFFFH	
	MELSECNET/H network module	F000H to FFFFH	

[1] Major error: Errors that may cause the CPU module to stop the operation, e.g. RAM error.

[2] Minor or moderate error: Errors that may allow the CPU module to continue the operation, e.g., battery error
or
Errors that may cause the CPU module to stop the operation, e.g., WDT error.

For determination of the error level (i.e. whether the operation can be continued or stopped) refer to column "CPU status" in the error code lists of sections 13.2 to 13.8).

13.1.3 Clearing an error

An error can be cleared as far as the CPU module continues its operation regardless of the error.

1. Remove the error cause.
2. Store the error code to be cleared in SD50.
3. Turn on SM50.
4. The error is cleared.

When the error in the CPU module is cleared, the special relay and special register or LEDs relating to the error return to the status before the error. If the same error occurs after clearing the error, the error will be registered to the error history again.

When multiple annunciators are detected, only the first annunciator detected can be cleared. For details on clearing errors, refer to the following manual:

User's manual (Function Explanation, Program Fundamentals) for the CPU module used

13.2 Error code list (1000 to 1999)

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1000	**MAIN CPU DOWN** Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU
	CPU UNIT DOWN Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always					LCPU
1001	**MAIN CPU DOWN** Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault – Universal model QCPU only: Accessed to outlying devices with the device range checks disabled (SM237 = 1). This error occurs only when BMOV, FMOV, and DFMOV instructions are executed. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative. – Universal model QCPU only: Check the devices specified by BMOV, FMOV, and DFMOV instructions and correct the device settings.	OFF	Flicker	Stop	QCPU
	CPU UNIT DOWN Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault – Accessed to outlying devices with the device range checks disabled (SM237 = 1). This error occurs only when BMOV, FMOV, and DFMOV instructions are executed. ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative. – Check the devices specified by BMOV, FMOV, and DFMOV instructions and correct the device settings.				LCPU
1002 1003 1004	**MAIN CPU DOWN** Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU
1002 1003 1004	**CPU UNIT DOWN** Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** Always					LCPU

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1005	**MAIN CPU DOWN** Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU
	MAIN CPU DOWN Boot operation was performed in the transfer destination without formatting. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power-on	– Before performing boot operation by the parameter, select "Clear program memory" to clear the program memory.				Qn(H) QnPH QnPRH
	CPU UNIT DOWN Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.				LCPU
1006	**MAIN CPU DOWN** Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always		OFF	Flicker	STOP	QCPU
	CPU UNIT DOWN Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** Always					LCPU
1007 1008	**MAIN CPU DOWN** Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1009	**MAIN CPU DOWN** – The voltage waveform that is outside the specification is applied to the power supply module, and an error is detected. – A failure is detected on the power supply module, CPU module, main base unit, extension base unit or extension cable. – When using the redundant base unit, the redundant power supply module failure in both systems and/or the redundant base unit failure are detected. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Correct the voltage waveform applied to the power supply module. – Reset the CPU module and RUN it again. If the same error is detected again, it is considered that the power supply module, CPU module, main base unit, extension base unit or extension cable is faulty. Replace the defective component. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) (first 5 digits of serial No. is 04101 or higher) QnPH QnPRH QnU
	CPU UNIT DOWN – A failure is detected on the power supply module or CPU module. – The voltage waveform that is outside the specification is applied to the power supply module, and an error is detected. ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** Always	– Correct the voltage waveform applied to the power supply module. – Reset the CPU module and run it again. If the same error code is displayed again, the cause is a hardware failure of the power supply module or CPU module. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	LCPU
1010	**END NOT EXECUTE** Entire program was executed without the execution of an END instruction. – When the END instruction is executed it is read as another instruction code, e.g. due to noise. – The END instruction has been changed to another instruction code somehow. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU LCPU
1020	**SFCP. END ERROR** – The SFC program cannot be normally terminated due to noise or any similar cause. – The SFC program cannot be normally terminated for any other reason. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When SFC program is executed	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnPH QnU LCPU
1035	**MAIN CPU DOWN** Runaway or failure of CPU module. – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnU
	CPU UNIT DOWN Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** Always					LCPU

Tab. 13-1: *Error code list (1000 to 1999)*

Error code list (1000 to 1999)

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1036	**MAIN CPU DOWN** Runaway or failure of CPU module – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q50UDEHCPU Q100UDEHCPU
1040 1041 1042	**CPU UNIT DOWN** Runaway or failure of CPU module (built-in I/O) – Malfunctioning due to noise or other reason – Hardware fault ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	LCPU
1101	**RAM ERROR** The sequence program storing program memory in the CPU module is faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At Reset/ When an END instruction executed	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU LCPU
1102	**RAM ERROR** – The work area RAM in the CPU module is faulty. – The standard RAM and extended RAM in the CPU module are faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At Reset/ When an END instruction executed		OFF	Flicker	Stop	QCPU LCPU
1103	**RAM ERROR** The device memory in the CPU module is faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Take noise reduction measures. – When indexing is performed, check the value of index register to see if it is within the device range. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU LCPU
	RAM ERROR – The device memory in the CPU module is faulty. – The device out of range is accessed due to indexing, and the device for system is over-written. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset/When an END instruction executed					Qn(H) (first 5 digits of serial No. is 08032 or higher) QnPH (first 5 digits of serial No. is 08032 or higher) QnPRH (first 5 digits of serial No. is 09012 or higher)
1104	**RAM ERROR** The address RAM in the CPU module is faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q00J/Q00/Q01 Qn(H) QnPH QnPRH

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1105	**RAM ERROR** The CPU memory in the CPU module is faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** **AT POWER ON/ AT RESET**	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q00J Q00 Q01 QnU
	RAM ERROR The CPU shared memory in the CPU module is faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset					Qn(H) (first 5 digits of serial No. is 04101 or higher) QnPH QnPRH QnU
1106	**RAM ERROR** – The program memory in the CPU module is faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** STOP –> RUN/When an END instruction executed	– Check the battery to see if it is dead or not. If dead, replace the battery. – Take noise reduction measures. – Format the program memory, write all files to the PLC, then reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Qn(H) QnPH (first 5 digits of serial No. is 07032 or higher) QnPRH
1107	**RAM ERROR** The work area RAM in the CPU module is faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	This suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH
1108						Qn(H) (first 5 digits of serial No. is 08032 or higher) QnPH (first 5 digits of serial No. is 08032 or higher) QnPRH (first 5 digits of serial No. is 09012 or higher)
1109	**RAM ERROR** The work area RAM in the CPU module is faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always					
1110	**TRK. CIR. ERROR** A fault was detected by the initial check of the tracking hardware. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	This suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH
1111	**TRK. CIR. ERROR** A tracking hardware fault was detected. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset					
1112	**TRK. CIR. ERROR** A tracking hardware fault was detected during running. – The tracking cable was disconnected and reinserted without the standby system being powered off or reset. – The tracking cable is not secured by the connector fixing screws. – The error occurred at a startup since the redundant system startup procedure was not followed. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** During running	– Start after checking that the tracking cable is connected. If the same error is displayed again, the cause is the hardware fault of the tracking cable or CPU module. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again. For details, refer to the manual of the redundant system.	OFF	Flicker	Stop	QnPRH
1113						

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1115	**TRK. CIR. ERROR** A fault was detected by the initial check of the tracking hardware. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	This suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH
1116	**TRK. CIR. ERROR** A tracking hardware fault was detected during running. – The tracking cable was disconnected and reinserted without the standby system being powered off or reset. – The tracking cable is not secured by the connector fixing screws. – The error occurred at a startup since the redundant system startup procedure was not followed. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** During running	– Start after checking that the tracking cable is connected. If the same error is displayed again, the cause is the hardware fault of the tracking cable or CPU module. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again. For details, refer to the manual of the redundant system.	OFF	Flicker	Stop	QnPRH
1150	**RAM ERROR** The memory of the CPU module in the Multiple CPU high speed transmission area is faulty. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
1160	**RAM ERROR** The program memory in the CPU module is overwritten. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At program execution	– Take noise reduction measures. – Format the program memory, write all files to the PLC, then reset the CPU module, and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU LCPU
1161	**RAM ERROR** The data of the device memory built in the CPU module is overwritten. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At program execution	– Take noise reduction measures. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.				
1163	**RAM ERROR** Data in the program memory of the CPU module were overwritten. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Take noise reduction measures. – For GX Works2, select "Transfer cache memory to program memory" in the Options dialog box. – Format the program memory, write all files to the CPU module, and run it again. If the same error code is displayed again, the cause is a hardware failure of the CPU module. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnU

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1164	**RAM ERROR** The destruction of the data stored in the standard RAM is detected. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Take noise reduction measures. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q10UD(E)H-, Q13UD(E)H-, Q20UD(E)H-, Q26UD(E)H-CPU L26CPU-BT
1166	**RAM ERROR** The internal memory in the CPU module is faulty. ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** Always					Q50UDEHCPU Q100UDEHCPU
1170	**RAM ERROR** The RAM of the CPU module (built-in I/O) is faulty. ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** At power ON/ At reset					LCPU
1171	**RAM ERROR** The RAM of the CPU module (built-in I/O) is faulty. ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** Always	Take noise reduction measures. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	
1172	**RAM ERROR** The RAM of the CPU module (built-in I/O) is faulty. ■ **Collateral information** • Common Information: — • Individual Information: Failure information ■ **Diagnostic Timing** At power ON/ At reset	Take noise reduction measures. If the same error is displayed again, this suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	LCPU
1200	**OPE. CIRCUIT ERR.** The operation circuit for index modification in the CPU module does not operate normally. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	This suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU LCPU
1201	**OPE. CIRCUIT ERR.** The hardware (logic) in the CPU module does not operate normally. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset					
1202	**OPE. CIRCUIT ERR.** The operation circuit for sequence processing in the CPU module does not operate normally. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset					

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1203	**OPE. CIRCUIT ERR.** The operation circuit for index modification in the CPU module does not operate normally. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	This suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH
1204	**OPE. CIRCUIT ERR.** The hardware (logic) in the CPU module does not operate normally. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed					
1205	**OPE. CIRCUIT ERR.** The operation circuit for sequence processing in the CPU module does not operate normally. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed					
1300	**FUSE BREAK OFF** There is an output module with a blown fuse. ■ **Collateral information** • Common Information: Module No.(Slot No.); For Remote I/O network: Network No./Station No. • Individual Information: — ■ **Diagnostic Timing** Always	– Check FUSE LED of the output modules and replace the module whose LED is lit. (The module with a blown fuse can also be identified using the programming tool. Check the special registers SD1300 to SD1331 to see if the bit corresponding to the module is "1".) – When a GOT is bus-connected to the main base unit or extension base unit, check the connection status of the extension cable and the earth status of the GOT.	OFF/ON	Flicker/ ON	Stop/ Continue (CPU operation can be set in the parameters at error occurrence)	Qn(H) QnPH QnPRH QnU
		Check ERR. LED of the output modules and replace the module whose LED is lit. (The module with a blown fuse can also be identified using the programming tool. Check the special registers SD130 to SD137 to see if the bit corresponding to the module is "1".)				Q00J/Q00/Q01
1310	**I/O INT ERROR** An interruption has occurred although there is no interrupt module. ■ **Collateral information** • Common Information: – • Individual Information: — ■ **Diagnostic Timing** During interrupt	Any of the mounted modules is experiencing a hardware fault. Therefore, check the mounted modules and change the faulty module. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU
	I/O INT ERROR An interruption has occurred although there is no interrupt module. ■ **Collateral information** • Common Information: – • Individual Information: — ■ **Diagnostic Timing** During interrupt	Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				LCPU

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1311	**I/O INT ERROR** An interrupt request from other than the interrupt module was detected. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** During interrupt	Take action so that an interrupt will not be issued from other than the interrupt module.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnU
	I/O INT ERROR An interrupt request from the module where interrupt pointer setting has not been made in the PLC parameter dialog box was detected. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** During interrupt	– Correct the interrupt pointer setting in the PLC system setting of the PLC parameter dialog box. – Take measures so that an interrupt is not issued from the module where the interrupt pointer setting in the PLC system setting of the PLC parameter dialog box has not been made. Correct the interrupt setting of the network parameter. Correct the interrupt setting of the intelligent function module buffer memory. Correct the basic program of the QD51.				Q00J/Q00/Q01 (Version A) QnPRH QnU
		– Correct the interrupt pointer setting in the PLC System tab of the PLC Parameter dialog box. – Take measures not to issue an interruption from the modules where the interrupt pointer setting is not configured in the PLC System tab of the PLC Parameter dialog box. – Correct the Interrupt Setting of the network parameter. – Correct the interrupt setting of the intelligent function module buffer memory. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				LCPU
1320 1321	**LAN CTRL.DOWN** The H/W self-diagnostics detected a LAN controller failure. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	This suggests a CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnU (with Built-in Ethernet port) LCPU

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1401	**SP. UNIT DOWN** – There was no response from the intelligent function module/special function module in the initial processing. – The size of the buffer memory of the intelligent function module/special function module is invalid. – An unsupported module is mounted. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset/When intelligent function module is accessed	– When the unsupported module is mounted, remove it. – When the corresponding module is supported, this suggests a hardware fault of the intelligent function module/special function module, CPU module and/or base unit. Contact your local Mitsubishi representative.	OFF/ON	Flicker/ ON	Stop/ Continue (can be selected for each intelligent function module by the parameters)	QCPU
	SP. UNIT DOWN – There was no response from the intelligent function module in the initial processing. – The size of the buffer memory of the intelligent function module is invalid. – There was no response from the intelligent function module. – The start I/O No. of the targeted intelligent function module is stored as a common information upon error. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset/When intelligent function module is accessed	Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				LCPU
1402	**SP. UNIT DOWN** The intelligent function module/special function module was accessed in the program, but there was no response. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When an intelligent function module access instruction is executed	This suggests a hardware fault of the intelligent function module/special function module, CPU module and/or base unit. Contact your local Mitsubishi representative.	OFF/ON	Flicker/ ON	Stop/ Continue (can be selected for each intelligent function module by the parameters)	QCPU
	SP. UNIT DOWN The intelligent function module was accessed in the program, but there was no response. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When an intelligent function module access instruction is executed	Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				LCPU

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1403	**SP. UNIT DOWN** An unsupported module is mounted. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** When an END instruction is executed	– When the unsupported module is mounted, remove it. – When the corresponding module is supported, this suggests a hardware fault of the intelligent function module/special function module, CPU module and/or base unit. Contact your local Mitsubishi representative. Contact your local Mitsubishi representative.	OFF/ON	Flicker/ ON	Stop/ Continue (can be selected for each intelligent function module by the parameters)	QCPU
	SP. UNIT DOWN – There was no response from the intelligent function module/special function module when the END instruction is executed. – An error is detected at the intelligent function module/special function module. – The I/O module (intelligent function module/ special function module) is nearly removed, completely removed, or mounted during running. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** Always	The CPU module, base module and/or the intelligent function module/special function module that was accessed is experiencing a hardware fault. Contact your local Mitsubishi representative.				
	SP. UNIT DOWN – There was no response from the intelligent function module when the END instruction is executed. – An error is detected at the intelligent function module. – The I/O module (intelligent function module/ special function module) is nearly removed, completely removed, or mounted during running. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** Always	Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				LCPU
1411	**CONTROL-BUS ERR.** When performing a parameter I/O allocation the intelligent function module/special function module could not be accessed during initial communications. (On error occurring, the head I/O number of the corresponding intelligent function module/special function module is stored in the common information.) ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module/special function module, CPU module or base unit is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU
1412	**CONTROL-BUS ERR.** The FROM/TO instruction is not executable, due to a control bus error with the intelligent function module/special function module. (On error occurring, the program error location is stored in the individual information.) ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** During execution of FROM/TO instruction set	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module/special function module, CPU module or base unit is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1413	**CONTROL-BUS ERR.** In a multiple CPU system, a CPU module incompatible with the multiple CPU system is mounted. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Remove the CPU module incompatible with the multiple CPU system from the main base unit, or replace the CPU module with a CPU module compatible with the multiple CPU system. – The intelligent function module, CPU module or base unit is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH
	CONTROL-BUS ERR. – Self-diagnostic error in the system bus – Self-diagnostic error in the CPU module – In a multiple CPU system, the control CPU setting of other CPUs, configured in the I/O Assignment tab of the PLC Parameter dialog box, differs from that of CPU No.1. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. Contact your local Mitsubishi representative. – Reconfigure the control CPU setting of other CPUs so that it can be the same as that of CPU No.1.				QCPU
1414	**CONTROL-BUS ERR.** – Fault of a loaded module was detected. – In a multiple CPU system, a CPU module incompatible with the multiple CPU system is mounted. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** Always	– Remove the CPU module incompatible with the multiple CPU system from the main base unit, or replace the CPU module with a CPU module compatible with the multiple CPU system. – Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU
	CONTROL-BUS ERR. An error is detected on the system bus. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** Always	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. Contact your local Mitsubishi representative.				Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPH QnPRH QnU
1415	**CONTROL-BUS ERR.** Fault of the main or extension base unit was detected. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q00J/Q00/Q01 Qn(H) (Function version is B or later) QnPH QnPRH QnU
	CONTROL-BUS ERR. Fault of the main or extension base unit was detected. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset/When an END instruction executed					Qn(H) (first 5 digits of serial No. is 08032 or higher) QnPH (first 5 digits of serial No. is 08032 or higher)

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1416	**CONTROL-BUS ERR.** System bus fault was detected at power-on or reset. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnU
	CONTROL-BUS ERR. In a multiple CPU system, a bus fault was detected at power-on or reset. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset					Q00/Q01 (Function version is B or later) QnU
1417	**CONTROL-BUS ERR.** A reset signal error was detected on the system bus. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH
1418	**CONTROL-BUS ERR.** In the redundant system, at power-on/reset or switching system, the control system cannot access the extension base unit since it failed to acquire the access right. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset/At Switching execution	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module, the Q6□WRB, or hardware of extension cable is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH (first 5 digits of serial No. is 09012 or higher)

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1430	**MULTI-C.BUS ERR.** The error of host CPU is detected in the Multiple CPU high speed bus. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
1431	**MULTI-C.BUS ERR.** The communication error with other CPU is detected in the Multiple CPU high speed bus. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Take noise reduction measures. – Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative.				
1432	**MULTI-C.BUS ERR.** The communication time out with other CPU is detected in the Multiple CPU high speed bus. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative.				
1433 1434 1435	**MULTI-C.BUS ERR.** The communication error with other CPU is detected in the Multiple CPU high speed bus. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** Always	– Take noise reduction measures. – Check the main base unit mounting status of the CPU module. – Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative.				
1436	**MULTI-C.BUS ERR.** The error of the Multiple CPU high speed main base unit is detected. (The error of the Multiple CPU high speed bus is detected.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative.				
1437		– Take noise reduction measures. – Check the main base unit mounting status of the CPU module. – Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative.				
1439	**MULTI-C.BUS ERR.** An error of the multiple CPU high speed main base unit was detected. (An error of the multiple CPU high speed bus was detected.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
1500	**AC DOWN** A momentary power supply interruption has occurred. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	Check the power supply.	ON	OFF	Continue	QCPU LCPU

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1510	**SINGLE PS. DOWN** The power supply voltage of either of redundant power supply modules on the redundant base unit dropped. ■ **Collateral information** • Common Information: Base No. / Nr. des Netzteils • Individual Information: — ■ **Diagnostic Timing** Always	Check the power supplied to the redundant power supply modules mounted on the redundant base unit.	ON	ON	Continue	Qn(H) (first 5 digits of serial No. is 04101 or higher) QnPH (first 5 digits of serial No. is 04101 or higher) QnPRH QnU (except Q00UJ-, Q00U- and Q01UCPU)
1520	**SINGLE PS. ERROR** On the redundant base unit, a damaged redundant power supply module was detected. ■ **Collateral information** • Common Information: Base No. / Nr. des Netzteils • Individual Information: — ■ **Diagnostic Timing** Always	Hardware fault of the redundant power supply module. Contact your local Mitsubishi representative.				
1600	**BATTERY ERROR** – The battery voltage in the CPU module has dropped below stipulated level. – The lead connector of the CPU module battery is not connected. – The lead connector of the CPU module battery is not securely engaged. ■ **Collateral information** • Common Information: Drive Name • Individual Information: — ■ **Diagnostic Timing** Always	– Change the battery. – If the battery is for program memory, standard RAM or for the back-up power function, install a lead connector. – Check the lead connector of the CPU module for looseness. Firmly engage the connector if it is loose. NOTE: When this error occurs, the BAT. LED of the CPU module is lit too.	ON	OFF	Continue	QCPU LCPU
1601	**BATTERY ERROR** Voltage of the battery on memory card has dropped below stipulated level. ■ **Collateral information** • Common Information: Drive Name • Individual Information: — ■ **Diagnostic Timing** Always	Change the battery. NOTE: When this error occurs, the BAT. LED of the CPU module is lit too.				Qn(H) QnPH QnPRH QnU (except Q00UJCPU, Q00UCPU, and Q01UCPU)
1610	**FLASH ROM ERROR]** The number of writing to flash ROM (standard ROM and system securement area) exceeds 100,000 times. (Number of writings = 100,000 times max.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When writing to ROM	Change the CPU module.	ON	ON	Continue	QnU LCPU

Tab. 13-1: *Error code list (1000 to 1999)*

Error code	Error Contents and Cause	Corrective Action	LED Status		CPU Status	Corresponding CPU
			RUN	ERR.		
1700	**BUS TIMEOUT ERROR** An error was detected on the system bus. – Self-diagnosis error of the system bus – Self-diagnosis error of the CPU module ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.	OFF	Flicker	STOP	LCPU
1710	**UNIT BUS ERROR** – An error was detected on the system bus. – An error was detected in the connected module. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** Always					
1720	**END COVER ERROR** A failure was detected on the END cover. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power-ON/At reset/When an END instruction executed	– Replace the END cover – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				
1730	**SYSTEM RST ERROR** An error was detected on the system bus. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power-ON/At reset					

Tab. 13-1: *Error code list (1000 to 1999)*

13.3 Error code list (2000 to 2999)

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2000	**UNIT VERFIY ERR.** In a multiple CPU system, a CPU module incompatible with the multiple CPU system is mounted. ■ **Collateral information** • Common Information: Module No. (Slot No.); For Remote I/O network: Network No./Station No. • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	Replace the CPU module incompatible with the multiple CPU system with a CPU module compatible with the multiple CPU system.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) (Function version is B or later) QnPH
	UNIT VERFIY ERR. The I/O module status is different from the I/O module information at power ON. I/O module (or intelligent function module) is not installed properly or installed on the base unit. ■ **Collateral information** • Common Information: Module No. (Slot No.); For Remote I/O network: Network No./Station No. • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	– Read the error common information at the programming tool, and check and/or change the module that corresponds to the numerical value (module number) there. – Alternatively, monitor special registers SD150 to SD157 using the programming tool, and check and replace the module where the bit of its data is "1".				Q00J/Q00/Q01
	UNIT VERFIY ERR. The I/O module status is different from the I/O module information at power ON. I/O module (or intelligent function module/special function module) not installed properly or installed on the base unit. ■ **Collateral information** • Common Information: Module No. (Slot No.); For Remote I/O network: Network No./Station No. • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	– Read the error common information at the programming tool, and check and/or change the module that corresponds to the numerical value (module number) there. – Alternatively, monitor special registers SD1400 to SD1431 using the programming tool and change the output module whose bit has a value of "1". – When a GOT is bus-connected to the main base unit or extension base unit, check the connection status of the extension cable and the grounding status of the GOT.				Qn(H) QnPH QnPRH QnU
2001	**UNIT VERFIY ERR.** During operation, a module was mounted on the slot where the empty setting of the CPU module was made. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	During operation, do not mount a module on the slot where the empty setting of the CPU module was made.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Q00J/Q00/Q01 (Function version is B or later) QnU
2010	**BASE LAY ERROR** – More than applicable number of extension base units have been used. – When a GOT was bus-connected, the CPU module was reset while the power of the GOT was OFF. ■ **Collateral information** • Common Information: Base No. • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Use the allowable number of extension base units or less. – Power on the Progammable Controller and GOT again.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnPRH Q00UJ-, Q00U-, Q01U- and Q02UCPU)
2011	**BASE LAY ERROR** The QA1S6□B, QA6□B or QA6ADP+A5□B/ A6□B was used as the base unit. ■ **Collateral information** • Common Information: Base No. • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Do not use the QA1S6□B, QA6□B and QA6ADP+A5□B/A6□B as the base unit.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnPH QnPRH QnU

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2012	**BASE LAY ERROR** – The GOT is bus-connected to the main base unit of the redundant system. The following errors are detected in the CPU redundant system compatible with the extension base unit. – The base unit other than the Q6□WRB is connected to the extension stage No.1. – The base unit is connected to any one of the extension stages No.2 to No.7, although the Q6□WRB does not exist in the extension stage No.1. – The other system CPU module is incompatible with the extension base unit. – The Q5□B, QA1S6□B, QA6□B or QA6ADP+A5□B/A6□B is connected. – The number of slots of the main base unit for both systems is different. – Information of the Q6□WRB cannot be read correctly. ■ **Collateral information** • Common Information: Base No. • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Remove a bus connection cable for GOT connection connected to the main base unit. – Use the Q6□WRB (fixed to the extension stage No.1) – Use the CPU module compatible with the extension base unit for the other system. – Do not use the Q5□B, QA1S6□B, QA6□B or QA6ADP+A5 B/A6□B for the base unit. – Use the main base unit which has the same number of slots. – Hardware failure of the Q6□WRB. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH (first 5 digits of serial No. is 09012 or higher)
2013	**BASE LAY ERROR** Stage number of the Q6□WRB is recognized as other than extension stage No.1 in the redundant system. ■ **Collateral information** • Common Information: Base No. • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Hardware failure of the Q6□WRB. Contact your local Mitsubishi representative.				
2020	**EXT.CABLE ERR.]** The following errors are detected in the redundant system. – At power-on/reset, the standby system has detected the error in the path between the control system and the Q6□WRB. – The standby system has detected the error in the path between the host system CPU and the Q6□WRB at END processing. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset/When an END instruction executed	Check to see if the extension cable between the main base unit and the Q6□WRB is connected correctly. If not, connect it after turning OFF the main base unit where the extension cable will be connected. If the cable is connected correctly, hardware of the CPU module, Q6□WRB, or extension cable is faulty. Contact your local Mitsubishi representative.				
2030	**NO END COVER** No end cover. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Attach an END cover – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.	OFF	Flicker	STOP	LCPU
2031	**NO END COVER** No end cover. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed					

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2040	**UNIT BAD CONNECT** – The I/O module status is different from that obtained at power-on. – The I/O module (including the intelligent function module) is nearly disconnected or is completely disconnected during running. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** Always	– Read common information of the error using the programming tool to identify the numeric value (module No.). Check the module corresponding to the value and replace it as necessary. – Monitor SD1400 to SD1431 using the programming tool to identify the module of which data bit is "1". Check the module and replace it as necessary. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.	OFF	Flicker	STOP	LCPU
2100	**SP. UNIT LAY ERR.** The slot to which the QI60 is mounted is set to other than Inteli (intelligent function module) or Interrupt (interrupt module) in the I/O assignment of PLC parameter. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Make setting again to match the PLC parameter I/O assignment with the actual loading status.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH
	SP. UNIT LAY ERR. Wrong I/O assignment setting of PLC parameter: – In the I/O assignment setting of PLC parameter, Inteli (intelligent function module) was allocated to an I/O module or vice versa. – In the I/O assignment setting of PLC parameter, a module other than CPU (or nothing) was allocated to the location of a CPU module or vice versa. – In the I/O assignment setting of the PLC parameter, switch setting was made to the module that has no switch setting. – In the I/O assignment setting of the PLC parameter dialog box, the number of points assigned to the intelligent function module is less than the number of points of the mounted module. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Make the PLC parameter's I/O assignment setting again so it is consistent with the actual status of the intelligent function module and the CPU module. – Delete the switch setting in the I/O assignment setting of the PLC parameter.				Qn(H) QnPH QnPRH QnU
	SP. UNIT LAY ERR. Wrong I/O assignment setting of PLC parameter: – In the I/O assignment setting of PLC parameter, Inteli (intelligent function module) was allocated to an I/O module or vice versa. – In the I/O assignment setting of PLC parameter, a module other than CPU (or nothing) was allocated to the location of a CPU module or vice versa. – In the I/O assignment setting of the PLC parameter dialog box, the number of points assigned to the intelligent function module is less than the number of points of the mounted module. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the parameter I/O allocation setting to conform to the actual status of the intelligent function module and the CPU module.				Q00J/Q00/Q01

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2100	**SP. UNIT LAY ERR.** Wrong I/O assignment setting of PLC parameter: – In the I/O assignment setting of PLC parameter, an intelligent function module was allocated to an I/O module or vice versa. – In the I/O assignment setting of the PLC parameter dialog box, the number of points assigned to the intelligent function module is less than the number of points of the mounted module. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Set the parameter again in the I/O Assignment tab of the PLC Parameter dialog box according to the CPU module mounted. – Delete the switch setting. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	LCPU
2101	**SP. UNIT LAY ERR.** 13 or more A-series special function modules (except for the A1SI61) that can initiate an interrupt to the CPU module have been installed. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reduce the A series special function modules (except the A1SI61) that can make an interrupt start to the CPU module to 12 or less.	OFF	Flicker	Stop	Qn(H)
2102	**SP. UNIT LAY ERR.** Seven or more A1SD51S have been installed. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Keep the number of A1SD51S to six or fewer.	OFF	Flicker	Stop	Qn(H)
2103	**SP. UNIT LAY ERR.** – Two or more QI60/A1SD51S modules are mounted in a single CPU system. – Two or more QI60/A1SD51S modules are set to the same control CPU in a multiple CPU system. – Two or more A1SD51S modules are loaded in a multiple CPU system. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the number of QI60/A1SD51S modules mounted in the single CPU system to one. – Change the number of QI60/A1SD51S modules set to the same control CPU to only one in the multiple CPU system. – Reduce the number of A1SD51S modules to only one in the multiple CPU system. When using an interrupt module with each QCPU in a multiple CPU system, replace it with the QI60. (Use one A1SI61 module + max. three QI60 modules or only the QI60 modules.)	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH
	SP. UNIT LAY ERR. Two or more QI60, A1SD51S interrupt modules have been mounted. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Install only one QI60, A1SD51S module.	OFF	Flicker	Stop	Qn(H) QnPRH
	SP. UNIT LAY ERR. Two or more QI60 modules are mounted. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Install only one QI60 module.	OFF	Flicker	Stop	Q00J/Q00/Q01 (first 5 digits of serial No. is 04101 or higher)
	SP. UNIT LAY ERR. Two or more QI60 modules where interrupt pointer setting has not been made are mounted. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Install only one QI60 module. – Make interrupt pointer setting to the second QI60 module and later.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnU

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2106	**SP. UNIT LAY ERR.** – Two or more MELSECNET/H and CC-Link IE controller network modules are mounted. – Two or more Ethernet modules are mounted. ■ **Collateral information** • Common Information: Module No. • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the number of MELSECNET/H or CC-Link IE controller network modules to one. – Reduce the number of Ethernet modules to one.	OFF	Flicker	Stop	Q00UJ
	SP. UNIT LAY ERR. – Two or more MELSECNET/H and CC-Link IE controller network modules are mounted in the entire system. – Two or more Ethernet modules are mounted in the entire system. ■ **Collateral information** • Common Information: Module No. • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the number of MELSECNET/H or CC-Link IE controller network modules to one in the entire system. – Reduce the number of Ethernet modules to one in the entire system.	OFF	Flicker	Stop	Q00U/Q01U
	SP. UNIT LAY ERR. – Three or more MELSECNET/H and CC-Link IE controller network modules in total are mounted in the entire system. – Three or more Ethernet interface modules are mounted in the entire system. ■ **Collateral information** • Common Information: Module No. • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the MELSECNET/H and CC-Link IE controller network modules up to two or less in the entire system. – Reduce the Ethernet interface modules up to two or less in the entire system.	OFF	Flicker	Stop	Q02U
	SP. UNIT LAY ERR. – Five or more MELSECNET/H and CC-Link IE controller network modules in total are mounted in the entire system. – Five or more Ethernet interface modules are mounted in the entire system. ■ **Collateral information** • Common Information: Module No. • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the MELSECNET/H and CC-Link IE controller network modules up to four or less in the entire system. – Reduce the Ethernet interface modules up to four or less in the entire system.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U and, Q02UCPU)
	SP. UNIT LAY ERR. – Three or more CC-Link IE controller network modules are mounted in the entire system. – Five or more MELSECNET/H and CC-Link IE controller network modules in total are mounted in the entire system. ■ **Collateral information** • Common Information: Module No. • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the CC-Link IE controller network modules up to two or less in the entire system. – Reduce the total number of the MELSECNET/H and CC-Link IE controller network modules up to four or less in the entire system.	OFF	Flicker	Stop	Qn(H) (first 5 digits of serial No. is 10042 or higher) QnPH (first 5 digits of serial No. is 10042 or higher) QnPRH (first 5 digits of serial No. is 10042 or higher)
	SP. UNIT LAY ERR. – Five or more MELSECNET/H modules have been installed. – Five or more Ethernet interface modules have been installed. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the number of MELSECNET/H modules to four or less. – Reduce the number of Ethernet modules to four or less.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2106	**SP. UNIT LAY ERR.** – Two or more MELSECNET/H modules were installed. – Two or more Ethernet modules were installed. – Three or more CC-Link modules were installed. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the MELSECNET/H modules to one. – Reduce the Ethernet modules to one. – Reduce the CC-Link modules to two or less.	OFF	Flicker	Stop	Q00J/Q00/Q01
	SP. UNIT LAY ERR. – The same network number or same station number is duplicated in the MELSECNET/H network system. ■ **Collateral information** • Common Information: Module No. (Steckplatz) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Check the network number and station number.	OFF	Flicker	Stop	Q00J/Q00/Q01 Qn(H) QnPH QnPRH
2107	**SP. UNIT LAY ERR.** The start X/Y set in the PLC parameter's I/O assignment settings is overlapped with the one for another module. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Make the PLC parameter's I/O assignment setting again so it is consistent with the actual status of the intelligent function module/special function modules.	OFF	Flicker	Stop	QCPU
	SP. UNIT LAY ERR. The start X/Y set in the PLC parameter's I/O assignment settings is overlapped with the one for another module. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Configure the start X/Y again in the I/O Assignment tab of the PLC Parameter dialog box according to the intelligent function module and I/O modules connected. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				LCPU
2108	**SP. UNIT LAY ERR.** • Network module A1SJ71LP21, A1SJ71BR11, A1SJ71AP21, A1SJ71AR21, or A1SJ71AT21B dedicated for the A2USCPU has been installed. • Network module A1SJ71QLP21 or A1SJ71QBR11 dedicated for the Q2ASCPU has been installed. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Replace the network module for the A2USCPU or the network module for the Q2ASCPU with the MELSECNET/H module.	OFF	Flicker	Stop	Qn(H)

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2110	**SP UNIT ERROR** – The location designated by the FROM/TO instruction set is not the intelligent function module/special function module. – The module that does not include buffer memory has been specified by the FROM/TO instruction. – The intelligent function module/special function module, Network module being accessed is faulty. – Station not loaded was specified using the instruction whose target was the CPU share memory. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed	– Read the individual information of the error using the programming tool, check the FROM/TO instruction that corresponds to that numerical value (program error location), and correct when necessary. – The intelligent function module/special function module that was accessed is experiencing a hardware fault. Therefore, change the faulty module. Alternatively, contact your local Mitsubishi representative.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Q00J/Q00/Q01 Qn(H) (Function version is B or later) QnPH QnPRH QnU
	SP UNIT ERROR – The location designated by the FROM/TO instruction set is not the intelligent function module. – The module that does not include buffer memory has been specified by the FROM/TO instruction. – The intelligent function module being accessed is faulty. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed	– Read the individual information of the error using the programming tool, check the FROM/TO instruction that corresponds to that numerical value (program error location), and correct when necessary. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				LCPU
2111	**SP UNIT ERROR** – The location designated by a link direct device (J□\□) is not a network module. – The I/O module (intelligent function module/ special function module) was nearly removed, completely removed, or mounted during running. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed.	– Read the individual information of the error using the programming tool, check the FROM/TO instruction that corresponds to that numerical value (program error location), and correct when necessary. – The intelligent function module/special function module that was accessed is experiencing a hardware fault. Therefore, change the faulty module. Alternatively, contact your local Mitsubishi representative.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QCPU
2112	**SP UNIT ERROR** – The module other than intelligent function module/special function module is specified by the intelligent function module/special function module dedicated instruction. Or, it is not the corresponding intelligent function module/special function module. – There is no network No. specified by the network dedicated instruction. Or the relay target network does not exist. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed/STOP → RUN	Read the individual information of the error using the programming tool, and check the intelligent function module /special function module dedicated instruction (network instruction) that corresponds to the value (program error part) to make modification.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QCPU
	SP UNIT ERROR – The module other than intelligent function module is specified by the intelligent function module dedicated instruction. Or, it is not the corresponding intelligent function module. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed/STOP → RUN	– Read the individual information of the error using the programming tool to identify the numeric value (program error location). Check the intelligent function module dedicated instruction corresponding to the value and correct it as necessary. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				LCPU

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2113	**SP UNIT ERROR** The module other than network module is specified by the network dedicated instruction. ■ **Collateral information** • Common Information: FFFFH (fixed) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed/STOP → RUN	Read the individual information of the error using the programming tool, and check the intelligent function module /special function module dedicated instruction (network instruction) that corresponds to the value (program error part) to make modification.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) QnPH
2114	**SP UNIT ERROR** An instruction, which on execution specifies other stations, has been used for specifying the host CPU. (An instruction that does not allow the host CPU to be specified). ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed/STOP → RUN	Read the individual information of the error using the programming tool, check the program corresponding that value (program error location), and make correction.	OFF/ON	Flicker/ ON	Stop/ Continue	Q00J/Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU
2115	**SP UNIT ERROR** An instruction, which on execution specifies the host CPU, has been used for specifying other CPUs. (An instruction that does not allow other stations to be specified). ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed/STOP → RUN	Read the individual information of the error using the programming tool, check the program corresponding that value (program error location), and make correction.	OFF/ON	Flicker/ ON	Stop/ Continue	Q00J/Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH
2116	**SP UNIT ERROR** – An instruction that does not allow the intelligent function module under the control of another CPU to be specified is being used for a similar task. – Instruction was executed for the A or QnA module under control of another CPU. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed/STOP → RUN	Read the individual information of the error using the programming tool, check the program corresponding that value (program error location), and make correction.	OFF/ON	Flicker/ ON	Stop/ Continue	Q00J/Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU
2117	**SP UNIT ERROR** A CPU module that cannot be specified in the instruction dedicated to the multiple CPU system was specified. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed/STOP → RUN	Read the individual information of the error using the programming tool, check the program corresponding that value (program error location), and make correction.	OFF/ON	Flicker/ ON	Stop/ Continue	
2118	**SP UNIT ERROR** When the online module change setting is set to be "enabled" in the PLC parameter in a multiple CPU system, intelligent function module controlled by other CPU using the FROM instruction/ intelligent function module device (U□\G□) is specified. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed	– When performing the online module change in a multiple CPU system, correct the program so that access will not be made to the intelligent function module controlled by the other CPU. – When accessing the intelligent function module controlled by the other CPU in a multiple CPU system, set the online module change setting to be "disabled" by parameter.	OFF/ON	Flicker/ ON	Stop/ Continue	Qn(H) (Function version is B or later) QnPH QnU (except Q00UJ-, Q00U-, Q01U and Q02UCPU)

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2120	**SP. UNIT LAY ERR.** The locations of an extension base unit is improper. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Check the location of the base unit.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Version A) Qn(H) QnPH
2121	**SP. UNIT LAY ERR.** The CPU module is installed to other than the CPU slot and slots 0 to 2. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Check the loading position of the CPU module and reinstall it at the correct slot.	OFF	Flicker	Stop	Qn(H) QnPH
2122	**SP. UNIT LAY ERR.** The QA1S6□B/QA6□B or QA6ADP+A5□B/A6□B are used for the main base unit. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Replace the main base unit with a usable one.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2124	**SP. UNIT LAY ERR.** – A module is mounted on the 65th slot or later slot. – A module is mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting. – A module is mounted on the slot whose number of I/O points exceeds 4096 points. – A module is mounted on the slot whose number of I/O points strides 4096 points. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Remove the module mounted on the 65th slot or later slot. – Remove the module mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting. – Remove the module mounted on the slot whose number of I/O points exceeds 4096 points. – Replace the module with the one whose number of occupied points does not exceed 4096 points.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH QnU (except Q00UJ-, Q00U-, Q01U and Q02UCPU)
	SP. UNIT LAY ERR. – A module is mounted on after the 25th slot (or after the 17th slot for the Q00UJCPU). – A module is mounted on the slot whose number is later than the one set in the "Base setting" on the I/O assignment tab of PLC parameter in theprogramming tool. – A module is mounted on the slot for which I/O points greater than 1024 (greater than 256 for the Q00UJCPU) is assigned. – A module is mounted on the slot for which I/O points is assigned from less than 1024 to greater than 1024 (from less than 256 to greater than 256 for the Q00UJCPU). ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Remove the module mounted on after the 25th (or after the 17th slot for the Q00UJCPU). – Remove the module mounted on the slot whose number is later than the one set in the "Base setting" on the I/O assignment tab of PLC parameter in the programming tool. – Remove the module mounted on the slot for which I/O points greater than 1024 (greater than 256 for the Q00UJCPU) is assigned. – Replace the end module with the one whose number of occupied points is within 1024 (within 256 for the Q00UJCPU).	OFF	Flicker	Stop	Q00UJ Q00U/Q01U
	SP. UNIT LAY ERR. – A module is mounted on the 37th slot or later slot. – A module is mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting. – A module is mounted on the slot whose number of I/O points exceeds 2048 points. – A module is mounted on the slot whose number of I/O points strides 2048 points ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Remove the module mounted on the 37th slot or later slot. – Remove the module mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting. – Remove the module mounted on the slot whose number of I/O points exceeds 2048 points. – Replace the module with the one whose number of occupied points does not exceed 2048 points.	OFF	Flicker	Stop	Q02U
	SP. UNIT LAY ERR. – A module is mounted on the 25th slot or later slot. (The 17th slot or later slot for the Q00JCPU.) – A module is mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting. – A module is mounted on the slot whose number of I/O points exceeds 1024 points. (256 points for the Q00JCPU.) – A module is mounted on the slot whose number of I/O points strides 1024 points. (256 points for the Q00JCPU.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Remove the module mounted on the 25th slot or later slot. (The 17th slot or later slot for the Q00JCPU.) – Remove the module mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting. – Remove the module mounted on the slot whose number of I/O points exceeds 1024 points. (256 points for the Q00J.) – Replace the module with the one whose number of occupied points does not exceed 1024 points. (256 points for the Q00J.) überschreitet, gegen eines mit weniger E/As	OFF	Flicker	Stop	Q00J Q00/Q01

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2124	**SP. UNIT LAY ERR.** – The number of connectable modules has exceeded 10. – A module is installed exceeding the I/O points of 4096. – A module is installed crossing the I/O points of 4096. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the number of connectable modules to 10. – Remove the module whose number of points exceeds 4096 points. – Replace the module installed to at end with the one whose number of occupied points does not exceed 4096 points. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	L26CPU-BT
	SP. UNIT LAY ERR. – The number of connectable modules has exceeded 10. – A module is installed exceeding the I/O points of 1024. – A module is installed crossing the I/O points of 1024. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reduce the number of connectable modules to 10. – Remove the module whose number of points exceeds 1024 points. – Replace the module installed to at end with the one whose number of occupied points does not exceed 1024 points. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				L02CPU
2125	**SP. UNIT LAY ERR.** – A module which the QCPU cannot recognise has been installed. – There was no response from the intelligent function module/special function module. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Install a usable module The intelligent function module/special function module is experiencing a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU
	SP. UNIT LAY ERR. – A module which the LCPU cannot recognise has been connected. – There was no response from the intelligent function module. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Connect an applicable module. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.				LCPU
2126	**SP. UNIT LAY ERR.** CPU module locations in a multiple CPU system are either of the following. – There are empty slots between the QCPU and QCPU/motion controller. – A module other than the High performance model QCPU/Process CPU (including the motion controller) is mounted on the left-hand side of the High performance model QCPU/Process CPU. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: ■ **Diagnostic Timing** At power ON/ At reset	– Mount modules on the available slots so that the empty slots will be located on the right-hand side of the CPU module. – Remove the module mounted on the left-hand side of the High performance model QCPU/ Process CPU, and mount the High performance model QCPU/Process CPU on the empty slot. Mount the motion CPU on the right-hand side of the High performance model QCPU/ Process CPU.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH
2128	**SP. UNIT LAY ERR.** An unusable module is mounted on the extension base unit in the redundant system. ■ **Collateral information** • Common Information: Module No. • Individual Information: ■ **Diagnostic Timing** At power ON/ At reset	Remove the unusable module from the extension base unit.	OFF	Flicker	Stop	QnPRH (first 5 digits of serial No. is 09012 or later)

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2150	**SP.UNIT VER. ERR.** In a multiple CPU system, the control CPU of the intelligent function module incompatible with the multiple CPU system is set to other than CPU No.1. ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset/At writing to progurammable controller	– Change the intelligent function module for the one compatible with the multiple CPU system. – Change the setting of the control CPU of the intelligent function module incompatible with the multiple CPU system to CPU No.1.	OFF	Flicker	Stop	Q00J/Q00/Q01 QnPH QnU (except Q00UJCPU)
2151	**SP. UNIT LAY ERR.** Either of the following modules incompatible with the redundant system has been mounted in a redundant system. • CC-Link IE controller network modules • MELSECNET/H modules • Ethernet modules ■ **Collateral information** • Common Information: Module No. (Slot No.) • Individual Information: ■ **Diagnostic Timing** At power ON/ At reset/At writing to progurammable controller	Use a module compatible with the redundant system.	OFF	Flicker	Stop	QnPRH
2170	**SYSTEM LAY ERR.** • A module which the LCPU cannot recognize is connected. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Disconnect the module that cannot be recognized. – Reset the CPU module and RUN it again. If the same error is displayed again, the cause is a hardware failure of the CPU module, I/O module, intelligent function module, or END cover. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	LCPU
2200	**MISSING PARA.** There is no parameter file in the drive specified as valid parameter drive by the DIP switches. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Check and correct the valid parameter drive settings made by the DIP switches. – Set the parameter file to the drive specified as valid parameter drive by the DIP switches.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH
	MISSING PARA. There is no parameter file at the program memory. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Set the parameter file to the program memory.				Q00J/Q00/Q01
	MISSING PARA. Parameter file does not exist in all drives where parameters will be valid. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Set a parameter file in a drive to be valid.				QnU
2200	**MISSING PARA.** There is no parameter file in the program memory. – When using a parameter file in an SD memory card, the SD memory card is being disabled by SM606 (SD memory card forced disable instruction). ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Write parameter files to the program memory of the CPU module. – Cancel the SD memory card forced disable instruction.	OFF	Flicker	Stop	LCPU

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2210	**BOOT ERROR** The contents of the boot file are incorrect. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Check the boot setting.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPH QnPRH QnU
2211	**BOOT ERROR** File formatting is failed at a boot. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reboot. – CPU module hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Qn(H) QnPRH QnU
2213	**BOOT ERROR** The file was booted from the SD memory card to the program memory or standard ROM but it was not booted to the CPU module due to either of the following reasons. – The passwords for the password 32 do not match between transfer source file and destination file. – The password 32 is not configured for the transfer source file while it is configured for the destination file. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Check the setting of the password 32 for the transfer source file and destination files. – Delete the boot setting from the parameter file of the SD memory card.	OFF	Flicker	Stop	LCPU
2220	**RESTORE ERROR** The device information (number of points) backuped by the device data backup function is different from the number of device points of the PLC parameter. After this error occurred, perform restore per power-on/reset until the number of device points is identical to the number of device points in the PLC parameter, or until the backup data is deleted. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Set the number of device points at the time of backup to the device point setting in [PLC parameter]. Then, turn ON from OFF power supply, or reset the CPU and cancel reset. – Delete the backuped data, and turn ON from OFF power supply, or reset the CPU and cancel reset.	OFF	Flicker	Stop	QnU LCPU
2221	**RESTORE ERROR** The device information backuped by the device data backup function is incomplete. (Turning power supply OFF or reset is suspected.) Do not return the data when this error occurs. Also, delete the incomplete device information at the time of this error occurrence. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the CPU module and run it again.	OFF	Flicker	Stop	QnU LCPU
2225	**RESTORE ERROR** The model name of the restoration destination CPU module is different from the one of the backup source CPU module. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Execute a restore for the CPU module whose name is same as the backup source CPU module.	OFF	Flicker	Stop	QnU (first five digits of the serial number is 04101 or higher) LCPU

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2226	**RESTORE ERROR** – The backup data file is destroyed. (The content of the file is different from the check code.) – Reading the backup data from the SRAM memory card is not successfully completed. – Since the write protect switch of the SRAM card is set to on (write inhibited), the checked "Restore for the first time only" setting cannot be performed. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Execute a restore of other backup data because the backup data may be destructed. – Set the write protect switch of the SRAM card to off (write enabled).	OFF	Flicker	Stop	QnU (first five digits of the serial number is 04101 or higher)
	RESTORE ERROR – The backup data file is destroyed. (The content of the file is different from the check code.) – Reading the backup data from the SD memory card is not successfully completed. – Since the write protect switch of the SD card is set to on (write prohibited), the checked "Restore for the first time only" setting cannot be performed. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Restore with any other backup data because the backup data may have been corrupted. – Set the write protect switch of the SD memory card to off (write-enabled).	OFF	Flicker	Stop	LCPU
2227	**RESTORE ERROR** Writing the backup data to the restoration destination drive is not successfully completed. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Execute a restore for the other CPU module too because the CPU module may be damaged.	OFF	Flicker	Stop	QnU (first five digits of the serial number is 04101 or higher) LCPU
2300	**ICM. OPE. ERROR** – A memory card was removed without turning on SM609 (Memory card remove/insert enable flag). – A memory card was removed while SM600 (Memory card usable flags) is on. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** When memory card is inserted or removed	– Turn on SM609 (Memory card remove/insert enable flag) and then remove the memory card. – Check that SM600 (Memory card usable flags) is off and then remove the memory card.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) QnPH QnPRH QnU (except Q00UJ-, Q00U- and Q01UCPU)
	ICM. OPE. ERROR – A memory card was removed without turning off the SD memory card lock switch. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** When memory card is inserted or removed	– Turn off the SD memory card lock switch first and then remove the memory card.				LCPU

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2301	**ICM. OPE. ERROR** – The memory card has not been formatted. – Memory card format status is incorrect. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** When memory card is inserted or removed.	– Format memory card. – Reformat memory card. – If the memory card is a flash card, write data to the flash card in any of the following methods.: 1)Write program memory to the ROM 2)Write data to the CPU module (flash ROM 3) Back up data to the flash card 4)Write image data to an external device, such as a memory card writer. – If the same error code is displayed again, the cause is a failure of the memory card. Contact your local Mitsubishi representative.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) QnPH QnPRH QnU (except Q00UJ-, Q00U- and Q01UCPU) LCPU
	ICM. OPE. ERROR – The QCPU file does not exist in the Flash card. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** When memory card is inserted or removed.	– Write the QCPU file to the Flash card				Qn(H) QnPH QnPRH QnU (except Q00UJ-, Q00U- and Q01UCPU)
	ICM. OPE. ERROR – SRAM card failure is detected. (It occurs when automatic format is not set.) – Writing parameters was performed during setting file registers. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** When memory card is inserted or removed.	– Format SRAM card after changing battery of SRAM card. – Write a parameter, which sets the file register at "Not available", in CPU, and then perform the operation.				QnU (except Q00UJ-, Q00U- and Q01UCPU)
2302	**ICM. OPE. ERROR** – A memory card that cannot be used with the CPU module has been installed. ■ **Collateral information** • Common Information: Drive name • Individual Information: — ■ **Diagnostic Timing** When memory card is inserted or removed.	– Format memory card. – Reformat memory card. – Check memory card.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) QnPH QnPRH QnU (except Q00UJ-, Q00U- and Q01UCPU)

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2400	**FILE SET ERROR** Automatic write to standard ROM was performed on the CPU module that is incompatible with automatic write to standard ROM. (Memory card where automatic write to standard ROM was selected in the boot file was fitted and the parameter enable drive was set to the memory card.) ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/ At reset/At writing to programmable controller	– Execute automatic write to standard ROM on the CPU module which is compatible with automatic write to standard ROM. – Using a programming tool, perform write of parameters and programs to standard ROM. – Change the memory card for the one where automatic write to standard ROM has not been set, and perform boot operation from the memory card.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH
	FILE SET ERROR The file designated at the PLC file settings in the parameters cannot be found. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/ At reset/At writing to programmable controller/STOP → RUN	– Read the individual information of the error using a programming tool, check to be sure that the parameter drive name and file name correspond to the numerical values there (parameter number), and correct. – Create a file created using parameters, and load it to the CPU module.				QCPU
	FILE SET ERROR – The file specified with a parameter does not exist. – When using a file in an SD memory card, the SD memory card is being disabled by SM606 (SD memory card forced disable instruction). ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/ At reset/At writing to programmable controller/STOP → RUN	– Read the individual information of the error using the programming tool to identify the numeric value (parameter No.). Check the drive name and file name of the parameter corresponding to the value, and correct it as necessary. Create the specified file and write it to the CPU module. – Cancel the SD memory card forced disable instruction.				LCPU

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2401	**FILE SET ERROR** Program memory capacity was exceeded by performing boot operation or automatic write to standard ROM. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/ At reset/At writing to programmable controller	– Check and correct the parameters (boot setting). – Delete unnecessary files in the program memory. – Choose "Clear program memory" for boot in the parameter so that boot is started after the program memory is cleared.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH
	FILE SET ERROR Program memory capacity was exceeded by performing boot operation. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/ At reset/At writing to programmable controller					QnU LCPU
	FILE SET ERROR The file specified by parameters cannot be created. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/ At reset/At writing to programmable controller	– Read the individual information of the error using a programming tool, check to be sure that the parameter drive name and file name correspond to the numerical values there (parameter number), and correct. – Check the space remaining in the memory card.	OFF	Flicker	Stop	QCPU LCPU
	FILE SET ERROR – Although setting is made to use the device data storage file, there is no empty capacity required for creating the device data storage file in the standard ROM. – When the latch data backup function (to standard ROM) is used, there is no empty capacity required for storing backup data in standard ROM. (The parameter number "FFFFH" is displayed for the error individual information.) ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/ At reset/At writing to programmable controller/STOP → RUN	Secure the empty capacity of the standard ROM.	OFF	Flicker	Stop	QnU LCPU
2406	**FILE SET ERROR** – When the extended data register and extended link register are configured in the File Register Extended Setting in the Device tab of the PLC Parameter dialog box, the size of the file register file is smaller than that specified in the PLC File tab. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** STOP → RUN	– Correct the size for the file register file in the PLC File tab of the PLC Parameter dialog box. – Correct the setting for the File Register Extended Setting in the Device tab of the PLC Parameter dialog box.	OFF	Flicker	Stop	QnU LCPU
2410	**FILE OPE. ERROR** – The specified program does not exist in the program memory. This error may occur when the ECALL, EFCALL, PSTOP, PSCAN, POFF or PLOW instruction is executed. – The specified file does not exist. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed	– Read the individual information of the error using a programming tool, check to be sure that the program corresponds to the numerical values there (program location), and correct. – Create a file created using parameters, and load it to the CPU module. – In case a specified file does not exist, write the file to a target memory and/or check the file specified with the instruction again.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) QnPH QnPRH QnU LCPU

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2411	**FILE OPE. ERROR** – The file is one which cannot be specified by the sequence program (such as comment file). – The specified program exists in the program memory, but has not been registered in the program setting of the Parameter dialog box. This error may occur when the ECALL, EFCALL, PSTOP, PSCAN or POFF instruction is executed. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed	Read the individual information of the error using a programming tool, check to be sure that the program corresponds to the numerical values there (program location), and correct.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) QnPH QnPRH QnU LCPU
2412	**FILE OPE. ERROR** The SFC program file is one that cannot be designated by the sequence program. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed	Read the individual information of the error using a programming tool, check to be sure that the program corresponds to the numerical values there (program location), and correct.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) QnPH QnPRH QnU LCPU
2413	**FILE OPE. ERROR** Check to ensure that the designated file has not been write protected. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Program error location ■ **Diagnostic Timing** When instruction executed	– Read the individual information of the error using a programming tool, check to be sure that the program corresponds to the numerical values there (program location), and correct. – Check to ensure that the designated file has not been write protected.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) QnPH QnPRH
2500	**CAN'T EXE. PRG.** – There is a program file that uses a device that is out of the range set in the PLC parameter device setting. – After the PLC parameter setting is changed, only the parameter is written into the PLC. – Although an SFC program exists, the step relay points is set to "0" in the Device tab of the PLC Parameter dialog box. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Read the common information of the error using a programming tool, check to be sure that the parameter device allocation setting and the program file device allocation correspond to the numerical values there (file name), and correct if necessary. – Whenever a device setting is changed, write both the parameter and program file to the CPU module. – To use the SFC program, set the step relay points to 8 k.	OFF	Flicker	Stop	QCPU LCPU
	CAN'T EXE. PRG. After the index modification of the PLC parameter is changed, only the parameter is written to the PLC. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	When the index modification of the PLC parameter is changed, batch-write the parameter and program file into the PLC.	OFF	Flicker	Stop	QnU LCPU

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2501	**CAN'T EXE. PRG.** There are multiple program files although "none" has been set at the PLC parameter program settings. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Edit the PLC parameter program setting to "yes". – Alternatively, delete unneeded programs.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH QnU LCPU
	CAN'T EXE. PRG. – There are three or more program files. – The program name differs from the program contents. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Delete unnecessary program files. – Match the program name with the program contents.	OFF	Flicker	Stop	Q00J/Q00/Q01
2502	**CAN'T EXE. PRG.** – The program file is incorrect. – Alternatively, the file contents are not those of a sequence program. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Check whether the program version is ***.QPG, and check the file contents to be sure they are for a sequence program.	OFF	Flicker	Stop	QCPU LCPU
	CAN'T EXE. PRG. The program file is not the one for the redundant CPU. – Alternatively, the file contents are not those of a sequence program. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Create a program using GX Developer, GX IEC Developer or PX Developer for which the PLC type has been set to the redundant CPU (Q12PRH/Q25PRH), and write it to the CPU module.	OFF	Flicker	Stop	QnPRH
2503	**CAN'T EXE. PRG.** There are no program files at all. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Check program configuration. – Check parameters and program configuration.	OFF	Flicker	Stop	QCPU LCPU
2504	**CAN'T EXE. PRG.** Two or more SFC normal programs or control programs have been designated. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Check program configuration. – Check parameters and program configuration.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH QnU LCPU
	CAN'T EXE. PRG. There are two or more SFC programs. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Reduce the SFC programs to one.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later)

Tab. 13-2: *Error code list (2000 to 2999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
2700	**REMOTE PASS.FAIL** The count of remote password mismatches reached the upper limit. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	Check for illegal accesses. If any illegal access is identified, take actions such as disabling communication of the connection. If no illegal access is identified, clear the error and perform the following. (Clearing the error also clears the count of remote password mismatches.) – Check if the remote password sent is correct. – Check if the remote password has been locked. – Check if concurrent access was made from multiple devices to one connection by UDP. – Check if the upper limit of the remote password mismatch count is too low.	ON	ON	Continue	QnU with Built-in Ethernet port LCPU
2710	**SNTP OPE.ERROR** Time setting failed when the programmable controller was powered ON or reset. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When time setting function is executed.	– Check if the time setting function is set up correctly. – Check if the specified SNTP server is operating normally, or if any failure has occurred on the network connected to the specified SNTP server computer.	OFF/ON	Flicker/ ON	Stop/ Continue	
2900	**DISPLAY ERROR** The display unit was attached or detached while the CPU module is on. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Do not detach the display unit during operation. – Ensure that the display unit is securely attached to the CPU module. – Reset the CPU module and run it again. If the same error is displayed again, the CPU module or display unit is faulty. Contact your local Mitsubishi representative.	ON	ON	Continue	LCPU
2901	**DISPLAY ERROR** A failure was detected in the display unit (in an initial processing). ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset					
2902	**DISPLAY ERROR** A failure was detected in the display unit (during operation). ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Ensure that the display unit is securely attached to the CPU module. – Reset the CPU module and run it again. If the same error is displayed again, the CPU module or display unit is faulty. Contact your local Mitsubishi representative.				

Tab. 13-2: *Error code list (2000 to 2999)*

13.4 Error code list (3000 to 3999)

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3000	**PARAMETER ERROR** In a multiple CPU system, the intelligent function module under control of another CPU is specified in the interrupt pointer setting of the PLC parameter. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Specify the head I/O number of the intelligent function module under control of the host CPU. – Delete the interrupt pointer setting of the parameter.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)
	PARAMETER ERROR The PLC parameter settings for timer time limit setting, the RUN-PAUSE contact, the common pointer number, general data processing, number of empty slots, system interrupt settings, baud rate setting, and service processing setting are outside the range that can be used by the CPU module. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Read the individual information of the error using the programming tool, check the parameter item corresponding to the numerical value (parameter No.), and correct it. – Rewrite corrected parameters to the CPU module, reload the CPU power supply and/or reset the module. – If the same error occurs, it is thought to be a hardware error. Contact your local Mitsubishi representative.				QCPU
	PARAMETER ERROR In a program memory check, the check capacity has not been set within the range applicable for the CPU module. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller					QnPH QnPRH (first 5 digits of serial No. is 07032 or higher)
	PARAMETER ERROR The parameter settings in the error individual information (special register SD16) are illegal. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller					QCPU
	PARAMETER ERROR The ATA card is set to the memory card slot when the specified drive for the file register is set to "memory card (ROM)" and [Use the following file] or [Use the same file name as the program] (either one is allowed) is set in the PLC file setting. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller					QnU (except Q00UJ-, Q00U- and Q01UCPU)

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3000	**PARAMETER ERROR** Any of the values for the Timer Limit Setting, RUNPAUSE Contacts, Common Pointer No., Points Occupied by Empty Slot, System Interrupt Setting, or Service Processing Setting option configured in the PLC Parameter dialog box are outside the range of the CPU module. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Read the individual information of the error using the programming tool, check the parameter item corresponding to the numerical value (parameter No.), and correct it. – If the same error occurs, the cause is a failure of the program memory of the CPU module, standard RAM, or SD memory card. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	LCPU
3001	**PARAMETER ERROR** The parameter settings are corrupted. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Read the individual information of the error using the programming tool, check the parameter item corresponding to the numerical value (parameter No.), and correct it. – Rewrite corrected parameters to the CPU module, reload the CPU power supply and/or reset the module. – If the same error occurs, it is thought to be a hardware error. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU LCPU
3002	**PARAMETER ERROR** When "Use the following file" is selected for the file register in the PLC file setting of the PLC parameter dialog box, the specified file does not exist although the file register capacity has been set. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Read the individual information of the error using the programming tool, check the parameter item corresponding to the numerical value (parameter No.), and correct it. – Rewrite corrected parameters to the CPU module, reload the CPU power supply and/or reset the module. – If the same error occurs, it is thought to be a hardware error. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH
	PARAMETER ERROR When "Use the following file" is set for the file register in the PLC file setting of the PLC parameter dialog box and the capacity of file register is not set, the file register file does not exist in the specified target memory. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller					QnU (except Q00UJCPU) LCPU
	PARAMETER ERROR When "Use the following file" is set for the device data storage file in [PLC file] of [PLC parameter], and [Capacity] is not set, the device data storage file does not exist in the target memory. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller					QnU LCPU

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3003	**PARAMETER ERROR** The automatic refresh range of the multiple CPU system exceeded the file register capacity. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** When an END instruction executed	Change the file register file for the one refreshenabled in the whole range.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)
	PARAMETER ERROR The number of devices set at the PLC parameter device settings exceeds the possible CPU module range. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Read the individual information of the error using the programming tool, check the parameter item corresponding to the numerical value (parameter No.), and correct it. – If the error is still generated following the correction of the parameter settings, the possible cause is the memory error of the CPU module's program memory or the memory card. Contact your local Mitsubishi representative.				QCPU LCPU
3004	**PARAMETER ERROR** The parameter file is incorrect. Alternatively, the contents of the file are not parameters. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Check whether the parameter file version is ***.QPA, and check the file contents to be sure they are parameters.	OFF	Flicker	Stop	QCPU LCPU
3005	**PARAMETER ERROR** The contents of the parameter are damaged. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Read the individual information of the error using the programming tool, check the parameter item corresponding to the numerical value (parameter No.), and correct it. – Write the modified parameter items to the CPU module again, and power-on the Programmable Controller or reset the CPU module. – When the same error occurs again, the hardware is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Qn(H) (first 5 digits of serial No. is 09012 or higher) QnPH (first 5 digits of serial No. is 10042 or higher) QnPRH (first 5 digits of serial No. is 10042 or higher)
3006	**PARAMETER ERROR** – The high speed interrupt is set in a Q02CPU. – The high speed interrupt is set in a multiple CPU system. – The high speed interrupt is set for a not applicable base unit. – No module is installed at the I/O address designated by the high speed interrupt. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Delete the setting of the Q02CPU's high speed interrupt. To use high speed interrupts, change the CPU module to one of the Q02H/Q06H/Q12H/Q25HCPU. – To use a multiple CPU system, delete the setting of the high-speed interrupt. To use high speed interrupts, change the system to a single CPU system. – Use applicable base units. – Re-examine the I/O address designated by the high speed interrupt setting.	OFF	Flicker	Stop	Qn(H) (first 5 digits of serial No. is 04012 or higher)
3007	**MISSING PARA.** The parameter file in the drive specified as valid parameter drive by the DIP switches is inapplicable for the CPU module. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Create parameters using the programming tool, and write them to the drive specified as valid parameter drive by the DIP switches.	OFF	Flicker	Stop	QnPRH

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3009	**PARAMETER ERROR** In a multiple CPU system, the modules for AnS, A, Q2AS and QnA have been set to multiple control CPUs. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Re-set the parameter I/O assignment to control them under one CPU module. Change the parameters of all CPUs in the multiple CPU system.	OFF	Flicker	Stop	Qn(H) (Function version is B or later)
3010	**PARAMETER ERROR** The parameter-set number of CPU modules differs from the actual number in a multiple CPU system. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Match the number of (CPU modules in multiple CPU setting) - (CPUs set as empty in I/O assignment) with that of actually mounted CPU modules.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH
3012	**PARAMETER ERROR** Multiple CPU setting or control CPU setting differs from that of the reference CPU settings in a multiple CPU system. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Match the multiple CPU setting or control CPU setting in the PLC parameter with that of the reference CPU (CPU No. 1) settings.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnU

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3013	**PARAMETER ERROR** In a multiple CPU system, the multiple CPU auto refresh setting is any of the following: – When a bit device is specified as a refresh device, a number other than a multiple of 16 is specified for the refresh-starting device. – The device specified is other than the one that may be specified. – The number of send points is an odd number. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Check the following in the multiple CPU auto refresh setting and make correction. – When specifying the bit device, specify 0 or a multiple of 16 for the refresh starting device. – Specify the device that may be specified for the refresh device. – Set the number of send points to an even number.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH
	PARAMETER ERROR In a multiple CPU system, the multiple CPU auto refresh setting is any of the following: – The total number of transmission points is greater than the maximum number of refresh points. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Check the following in the multiple CPU auto refresh setting and make correction. – The total number of transmission points is within the maximum number of refresh points.				Q00/Q01 (Function version is B or later)
	PARAMETER ERROR In a multiple CPU system, the multiple CPU auto refresh setting is any of the following: – The device specified is other than the one that may be specified. – The number of send points is an odd number. – The total number of send points is greater than the maximum number of refresh points. – The setting of the refresh range crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W). – No device is set in the host CPU send range. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Check the following in the multiple CPU auto refresh setting and make correction. – Specify the device that may be specified for the refresh device. – Set the number of send points to an even number. – Set the total number of send points within the range of the maximum number of refresh points. – Set the refresh range so that it does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W). – For the send range of the host CPU, refresh target device must be specified. If a send range is not necessary, delete the applicable send range.				QnU (except Q00UJ)
3014	**PARAMETER ERROR** – In a multiple CPU system, the online module change parameter (multiple CPU system parameter) settings differ from those of the reference CPU 1. – In a multiple CPU system, the online module change setting is enabled although the CPU module mounted does not support online module change. – In a multiple CPU system, online module change parameter was corrected and then it was written to the CPU module. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Match the online module change parameter with that of the reference CPU 1. – If the CPU module that does not support online module change is mounted, replace it with the CPU module that supports online module change.	OFF	Flicker	Stop	Qn(H) QnPH QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3015	**PARAMETER ERROR** In a multiple CPU system configuration, the CPU verified is different from the one set in the parameter setting. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No./CPU No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Read the individual information of the error using the programming tool, check the parameter item corresponding to the numerical value (parameter No./CPU No.) and parameter of target CPU, and correct them.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
3016	**PARAMETER ERROR** The CPU module incompatible with multiple CPU synchronized boot-up is set as the target for the synchronized boot-up in the [Multiple CPU synchronous startup setting]. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No./CPU No. ■ **Diagnostic Timing** At power ON/At reset/At writing to programmable controller	Delete the CPU module incompatible with multiple CPU synchronized boot-up from the setting.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
3040	**PARAMETER ERROR** The parameter file is damaged. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	With the programming tool, write [PLC parameter/Network parameter/Remote password] to a valid drive then reload the power supply for system and/or reset the CPU module. If the same error occurs, it is thought to be a hardware error. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Qn(H) (first 5 digits of serial No. is 07032 or higher) QnPH (first 5 digits of serial No. is 07032 or higher) QnPRH (first 5 digits of serial No. is 07032 or higher)
3041	**PARAMETER ERROR** Parameter file of intelligent function module is damaged. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	With the programming tool, write [Intelligent function module parameter] to a valid drive then reload the power supply for system and/or reset the CPU module. If the same error occurs, it is thought to be a hardware error. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	
3042	**PARAMETER ERROR** The system file that stored the remote password setting information is damaged. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	– With the programming tool, write [PLC parameter/Network parameter/Remote password] to a valid drive then reload the power supply for system and/or reset the CPU module. If the same error occurs, it is thought to be a hardware error. Contact your local Mitsubishi representative. – When a valid drive for parameter is set to other than [program memory], set the parameter file (PARAM) at the boot file setting to be able to transmit to the program memory. With the programming tool, write [PLC parameter/Network parameter/Remote password] to a valid drive then reload the power supply for system and/or reset the CPU module. If the same error occurs, it is thought to be a hardware error. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3100	**LINK PARA. ERROR** In a multiple CPU system, the CC-Link IE controller network module controlled by another CPU is specified as the head I/O number of the CC-Link IE controller network module. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Delete the network parameter of the CC-Link IE controller network module controlled by another CPU. – Change the setting to the head I/O number of the CC-Link IE controller network module controlled by host CPU.	OFF	Flicker	Stop	Qn(H) (first 5 digits of serial No. is 09012 or higher) QnPH (first 5 digits of serial No. is 10042 or higher) QnU
	LINK PARA. ERROR The network parameter of the CC-Link IE controller network operating as the normal station is overwritten to the control station. Or, the network parameter of the CC-Link IE controller network operating as the control station is overwritten to the normal station. (The network parameter is updated on the module by resetting.) ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Reset the CPU module.	OFF	Flicker	Stop	Qn(H) (first 5 digits of serial No. is 09012 or higher) QnPH (first 5 digits of serial No. is 10042 or higher) QnPRH (first 5 digits of serial No. is 10042 or higher) QnU
	LINK PARA. ERROR – The number of modules actually mounted is different from that is set in Network parameter for CC-Link IE controller network. – The head I/O number of the actually mounted module is different from the one set in the network parameter of the CC-Link IE controller network. – Data cannot be handled in the parameter existing. – The network type of CC-Link IE controller network is overwritten during power-on. (When changing the network type, switch RESET to RUN.) ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Check the network parameters and actual mounting status, and if they differ, make them match. When network parameters are modified, write them to the CPU module. – Check the setting of extension base unit stage number. – Check the connection status of extension base unit and extension cables. When the GOT is busconnected to the main base unit or extension base unit, also check its connection status. If the error occurs even after the above checks, the possible cause is a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	
	LINK PARA. ERROR – The CC-Link IE controller network module is specified for the head I/O number of network parameter in the MELSECNET/H. – The MELSECNET/H module is specified for the head I/O number of network parameter in the CC-Link IE controller network. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN		OFF	Flicker	Stop	
	LINK PARA. ERROR – Although the CC-Link IE controller network module is mounted, network parameter for the CCLink IE controller network module is not set. – Although the CC-Link IE controller network and MELSECNET/H modules are mounted, network parameter for the MELSECNET/H module is not set. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN		OFF	Flicker	Stop	

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3100	**LINK PARA. ERROR** In a multiple CPU system, the MELSECNET/H under control of another CPU is specified as the head I/O number in the network setting parameter of the MELSECNET/H. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Delete the MELSECNET/H network parameter of the MELSECNET/H under control of another CPU. – Change the setting to the head I/O number of the MELSECNET/H under control of the host CPU.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)
	LINK PARA. ERROR The network parameter of the MELSECNET/H operating as the normal station is overwritten to the control station. Or, the network parameter of the MELSECNET/H operating as the control station is overwritten to the normal station. (The network parameter is updated on the module by resetting.) ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Reset the CPU module.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH QnU
	LINK PARA. ERROR – The number of modules actually mounted is different from that is set in Network parameter for MELSECNET/H. – The head I/O number of actually installed modules is different from that designated in the network parameter of MELSECNET/H. – Some data in the parameters cannot be handled. – The network type of MELSECNET/H is overwritten during power-on. (When changing the network type, switch RESET to RUN.) – The mode switch of MELSECNET/H module (for module with first 5 digits of serial No. is "07032" or higher) is outside the range. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Check the network parameters and actual mounting status, and if they differ, make them match. When network parameters are modified, write them to the CPU module. – Check the setting of extension base unit stage number. – Check the connection status of extension base unit and extension cables. When the GOT is busconnected to the main base unit or extension base unit, also check its connection status. If the error occurs even after the above checks, the possible cause is a hardware fault. Contact your local Mitsubishi representative. – Set the mode switch of MELSECNET/H module (for module with first 5 digits of serial No. is "07032" or higher) within the range.	OFF	Flicker	Stop	QCPU

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3101	**LINK PARA. ERROR** The link refresh range exceeded the file register capacity. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** When an END instruction executed	Change the file register file for the one that enables entire range refresh.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH QnU (except Q00UJCPU)
	LINK PARA. ERROR – When the station number of the MELSECNET/H module is 0, the PLC-to-PLC network parameter has been set. – When the station number of the MELSECNET/H module is other than 0, the remote master parameter setting has been made. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Correct the type or station number of the MELSECNET/H module in the network parameter to meet the used system.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH
	LINK PARA. ERROR The refresh parameter for the CC-Link IE controller network is outside the range. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Check the network parameters and actual mounting status, and if they differ, make them match. When network parameters are modified, write them to the CPU module. – Check the setting of extension base unit stage number. – Check the connection status of extension base unit and extension cables. When the GOT is busconnected to the main base unit or extension base unit, also check its connection status. If the error occurs even after the above checks, the possible cause is a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Qn(H) (first 5 digits of serial No. is 09012 or higher) QnPH (first 5 digits of serial No. is 10042 or higher) QnPRH (first 5 digits of serial No. is 10042 or higher) QnU
	LINK PARA. ERROR – The network No. specified by a network parameter is different from that of the actually mounted network. – The head I/O No. specified by a network parameter is different from that of the actually mounted I/O unit. – The network class specified by a network parameter is different from that of the actually mounted network. – The network refresh parameter of the MELSECNET/H, MELSECNET/10 is out of the specified area. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN		OFF	Flicker	Stop	QCPU
	LINK PARA. ERROR A multi-remote I/O network was configured using a module that does not support a multi-remote I/O network. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Use a module that supports a multi-remote I/O network.	OFF	Flicker	Stop	QnPH
	LINK PARA. ERROR – The system A of the MELSECNET/H remote master station has been set to other than Station No. 0. – The system B of the MELSECNET/H remote master station has been set to Station No. 0. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Set the system A of the MELSECNET/H remote master station to Station No. 0. – Set the system B of the MELSECNET/H remote master station to any of Station No. 1 to 64.	OFF	Flicker	Stop	QnPRH

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3101	**LINK PARA. ERROR** Since the number of points of the B/W device set in [Device] of the PLC parameter is lower than the number of B/W refresh device points when parameters of the MELSECNET/H are not set, the refresh between the CPU module and the MELSECNET/H cannot be performed. Number of B/W refresh device points when parameters of the MELSECNET/H are not set: – 1 network module mounted B: 8192; W: 8192 – 2 network modules mounted B: 8192 (4096x2); W: 8192 (4096x2) – 3 network modules mounted B: 6144 (2048x3); W: 6144 (2048x3) – 4 network modules mounted B: 8192 (2048x4); W: 8192 (2048x4) ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Set the refresh parameter of the MELSECNET/H in accordance with the number of points of B/W devices set in [Device] of the PLC parameter.	OFF	Flicker	Stop	Qn(H) (first 5 digits of serial No. is 09012 or higher) QnPH (first 5 digits of serial No. is 09012 or higher) QnPRH (first 5 digits of serial No. is 09012 or higher) QnU
	LINK PARA. ERROR The setting of the network refresh range crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W). ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Set the network refresh range so that it does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).	OFF	Flicker	Stop	QnU
3102	**LINK PARA. ERROR** A CC-Link IE controller network parameter error was detected. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Correct and write the network parameters. – If the error occurs after correction, it suggests a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Qn(H) (first 5 digits of serial No. is 09012 or higher) QnPH (first 5 digits of serial No. is 10042 or higher) QnPRH (first 5 digits of serial No. is 10042 or higher) QnU
	LINK PARA. ERROR The network module detected a network parameter error. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN		OFF	Flicker	Stop	QCPU
	LINK PARA. ERROR The station No. specified in pairing setting are not correct. – The stations are not numbered consecutively. – Pairing setting has not been made for the CPU module at the normal station. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Refer to the troubleshooting of the network module, and if the error is due to incorrect pairing setting, reexamine the pairing setting of the network parameter.	OFF	Flicker	Stop	QnPRH

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3102	**LINK PARA. ERROR** The CC-Link IE controller network module whose first 5 digits of serial No. is "09041" or lower is mounted. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Mount the CC-Link IE controller network module whose first 5 digits of serial No. is "09042" or higher.	OFF	Flicker	Stop	QnU
	LINK PARA. ERROR – Different network types are set between the control station and the normal station (CC IE Control Ext. Mode/Normal Mode). – The parameter in which "CC IE Control Ext. Mode" is set for "Network Type" was transferred to the CPU module that does not support the send points expansion function. – The parameter in which "CC IE Control Ext. Mode" is set was backed up to a memory card or GOT and then restored to the CPU module that does not support the send points expansion function. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Set the same network type (CC IE Control Ext. Mode/Normal Mode) for the control station and the normal station. – Do not use the parameter in which "CC IE Control Ext. Mode" is set for "Network Type" for the CPU module that does not support the send points expansion function. Or, use the CPU module and the CC-Link IE controller network module that support the send points expansion function in the same network.	OFF	Flicker	Stop	QnU
	LINK PARA. ERROR Group cyclic function in CC-Link IE controller network that does not correspond to group cyclic function is set. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Set group cyclic function in function version D or later of CC-Link IE controller network.	OFF	Flicker	Stop	QnU (first 5 digits of serial No. is 10042 or higher)
	LINK PARA. ERROR Pairing setting in CC-Link IE controller network modules installed in CPUs except for redundant CPUs was performed. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Examine the pairing setting for the network parameter in the control staion.	OFF	Flicker	Stop	Q00J/Q00/Q01 Qn(H) (first 5 digits of serial No. is 10042 or higher) QnPH (first 5 digits of serial No. is 10042 or higher) QnU (first 5 digits of serial No. is 10042 or higher)
	LINK PARA. ERROR – LB/LW own station send range at LB/LW4000 or later was set. – LB/LW setting (2) was performed. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Correct the network range assignments for the network parameter in the control station.	OFF	Flicker	Stop	Q00J/Q00/Q01

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3103	**LINK PARA. ERROR** In a multiple CPU system, Ethernet interface module under control of another station is specified to the start I/O number of the Ethernet network parameter. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Delete the Ethernet network parameter of Ethernet interface module under control of another station. – Change the setting to the start I/O number of Ethernet interface module under control of the host station.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)
	LINK PARA. ERROR – Although the number of modules has been set to one or a greater number in the Ethernet module count parameter setting, the number of actually mounted module is zero. – The start I/O No. of the Ethernet network parameter differs from the I/O No. of the actually mounted module. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Correct and write the network parameters. If the error occurs after correction, it suggests a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU
	LINK PARA. ERROR – Ethernet module whose network type is set to "Ethernet (main base)" is mounted on the extension base unit in the redundant system. – Ethernet module whose network type is set to "Ethernet (extension base)" is mounted on the main base unit in the redundant system. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Correct and write the network parameters. If the error occurs after correction, it suggests a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH (first 5 digits of serial No. is 09012 or higher)
3104	**LINK PARA. ERROR** – The Ethernet, MELSECNET/H and MELSECNET/10 use the same network number. – The network number, station number or group number set in the network parameter is out of range. – The specified I/O number is outside the range of the used CPU module. – The Ethernet-specific parameter setting is not normal. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Correct and write the network parameters. If the error occurs after correction, it suggests a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3105	**LINK PARA. ERROR** In a multiple CPU system, the CC-Link module under control of another station is specified as the head I/O number of the CC-Link network parameter. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Delete the CC-Link network parameter of the CC-Link module under control of another station. – Change the setting to the start I/O number of the CC-Link module under control of the host station.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)
	LINK PARA. ERROR – Though the number of CC-Link modules set in the network parameters is one or more, the number of actually mounted modules is zero. – The start I/O number in the common parameters is different from that of the actually mounted module. – The station type of the CC-Link module count setting parameters is different from that of the actually mounted station. ■ **Collateral information** • Common Information: File name//Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Correct and write the network parameters. If the error occurs after correction, it suggests a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QCPU L02CPU
	LINK PARA. ERROR – Although two or more CC-Link modules were configured in the Network Parameter dialog box, only one CC-Link modules are installed in the system. The start I/O number of the common parameter specified in the Network Parameter dialog box does not correspond to the system. – The station type specified in the Network Parameter dialog box for CC-Link does not correspond to the system. ■ **Collateral information** • Common Information: File name//Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN		OFF	Flicker	Stop	L26CPU-BT
	LINK PARA. ERROR – CC-Link module whose station type is set to "master station (compatible with redundant function)" is mounted on the extension base unit in the redundant system. – CC-Link module whose station type is set to "master station (extension base)" is mounted on the main base unit in the redundant system. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN		OFF	Flicker	Stop	QnPRH (first 5 digits of serial No. is 09012 or higher)

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3106	**LINK PARA. ERROR** The CC-Link link refresh range exceeded the file register capacity. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** When an END instruction executed	Change the file register file for the one refreshenabled in the whole range.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH QnU LCPU
	LINK PARA. ERROR The network refresh parameter for CC-Link is out of range. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Check the parameter setting.				QCPU LCPU
	LINK PARA. ERROR The setting of the network refresh range crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W). ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Set the network refresh range so that it does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).				QnU LCPU
3107	**LINK PARA. ERROR** – The CC-Link parameter setting is incorrect. – The set mode is not allowed for the version of the mounted CC-Link module. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Check the parameter setting.	OFF	Flicker	Stop	QCPU LCPU
3150	**LINK PARA. ERROR** – When the CC-Link IE field network is used, the network number set in "Network Parameter" and "Switch Setting" is duplicated. – No "Network Parameter" and "Switch Setting" are configured, or the CC-Link IE field network module with an incorrect switch setting is mounted. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset	– Check the parameter setting. – Configure "Network Parameter" and "Switch Setting", and then write network parameters and the switch setting to the module.	OFF	Flicker	Stop	QnU (first five digits of the serial number is "12012" or higher)
3200	**SFC PARA. ERROR** The parameter setting is illegal. Though Block 0 was set to "Automatic start" in the SFC setting of the PLC parameter dialog box, Block 0 does not exist. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. ■ **Diagnostic Timing** STOP → RUN	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnPH QnPRH QnU LCPU
3201	**SFC PARA. ERROR** The block parameter setting is illegal. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. ■ **Diagnostic Timing** STOP → RUN		OFF	Flicker	Stop	Qn(H) QnPH QnPRH

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3202	**SFC PARA. ERROR** The number of step relays specified in the device setting of the PLC parameter dialog box is less than that used in the program. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. ■ **Diagnostic Timing** STOP → RUN	Read the individual information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH
3203	**SFC PARA. ERROR** The execution type of the SFC program specified in the program setting of the PLC parameter dialog box is other than scan execution. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN (The diagnostic timing of CPU modules except for Universal QCPU can be performed only when switching the CPU modules to run.)	Read the individual information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH QnU LCPU
3300	**SP. PARA. ERROR** The start I/O number in the intelligent function module parameter set on GX Configurator differs from the actual I/O number. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. (gained by dividing the head I/O number of parameter in the intelligent function module set by GX Configurator by 10H) ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Check the parameter setting.	OFF	Flicker	Stop	QCPU LCPU

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3301	**SP. PARA. ERROR** – The refresh setting of the intelligent function module exceeded the file register capacity. – The intelligent function module set in GX Configurator differs from the actually mounted module. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. (gained by dividing the head I/O number of parameter in the intelligent function module set by GX Configurator by 10H) ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Change the file register file for the one which allows refresh in the whole range. – Check the parameter setting.	OFF	Flicker	Stop	Q00J/Q00/Q01 Qn(H) (Function version is B or later) QnPH QnPRH QnU LCPU
	SP. PARA. ERROR The intelligent function module's refresh parameter setting is outside the available range. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. (gained by dividing the head I/O number of parameter in the intelligent function module set by GX Configurator by 10H) ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Check the parameter setting. – Check the auto refresh setting.	OFF	Flicker	Stop	QCPU LCPU
	SP. PARA. ERROR The setting of the refresh parameter range crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W). ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. (gained by dividing the head I/O number of parameter in the intelligent function module set by GX Configurator by 10H) ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	Set the refresh parameter range so that it does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).	OFF	Flicker	Stop	QnU LCPU
3302	**SP. PARA. ERROR** The intelligent function module's refresh parameter are abnormal. ■ **Collateral information** • Common Information: File name • Individual Information: Parameter No. (gained by dividing the head I/O number of parameter in the intelligent function module set by GX Configurator by 10H) ■ **Diagnostic Timing** At writing to programmable controller	Check the parameter setting.	OFF	Flicker	Stop	QCPU LCPU
3303	**SP. PARA. ERROR** In a multiple CPU system, the automatic refresh setting or other parameter setting was made to the intelligent function module under control of another station. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/At writing to programmable controller	– Delete the automatic refresh setting or other parameter setting of the intelligent function module under control of another CPU. – Change the setting to the automatic refresh setting or other parameter setting of the intelligent function module under control of the host CPU.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3400	**REMOTE PASS. ERROR** The head I/O number of the target module of the remote password is set to other than 0H to 0FF0H. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Change the head I/O number of the target module to be within the 0H to 0FF0H range.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH QnU (first 5 digits of serial No. is 09012 or higher) LCPU
	REMOTE PASS. ERROR The head I/O number of the target module of the remote password is set to other than 0H to 07E0H. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Change the head I/O number of the target module to be within the 0H to 07E0H range.				Q02U
	REMOTE PASS. ERROR The head I/O number of the target module of the remote password is outside the following range: • Q00JCPU: 0H to 1E0H • Q00CPU/Q01CPU: 0H to 3E0H ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Change the head I/O number of the target module of the remote password for the number within the following range: – Q00JCPU: 0H to 1E0H – Q00CPU/Q01CPU: 0H to 3E0H				Q00J/Q00/Q01 (Function version is B or later)

Tab. 13-3: *Error code list (3000 to 3999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
3401	**REMOTE PASS. ERROR** Position specified as the head I/O number of the remote password file is incorrect due to one of the following reasons: – Module is not loaded. – Other than a the intelligent function module (I/O module) – Intelligent function module other than serial communication module, modem interface module or Ethernet module – Serial communication module or Ethernet module of function version A – The intelligent function module where remote password is available is not mounted. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Mount serial communication module, modem interface module or Ethernet module of function version B or later in the position specified in the head I/O No. of the remote password file.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH QnU
	REMOTE PASS. ERROR Position specified as the head I/O number of the remote password file is incorrect due to one of the following reasons: – No module – The intelligent function module installed is other than a serial communication module. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	In a position specified with a start I/O number of the remote password, install the intelligent function module where the remote password is available.	OFF	Flicker	Stop	LCPU
	REMOTE PASS. ERROR Any of the following modules is not mounted on the slot specified for the head I/O number of the remote password: – Serial communication module of function version B or later – Ethernet module of function version B or later – Modem interface module of function version B or later ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Mount serial communication module, modem interface module or Ethernet module of function version B or later in the position specified in the head I/O No. of the remote password file.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later)
	REMOTE PASS. ERROR Serial communication module, modem interface module or Ethernet module of function version B or later controlled by another CPU was specified in a multiple CPU system. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	– Change it for the Ethernet module of function version B or later connected by the host CPU. – Delete the remote password setting.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)

Tab. 13-3: *Error code list (3000 to 3999)*

13.5 Error code list (4000 to 4999)

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4000	**INSTRCT CODE. ERR.** – The program contains an instruction code that cannot be decoded. – An unusable instruction is included in the program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/ When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4001	**INSTRCT CODE. ERR.** The program contains a dedicated instruction for SFC although it is not an SFC program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/ When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPH QnPRH QnU LCPU
4002	**INSTRCT CODE. ERR.** – The name of dedicated instruction specified by the program is incorrect. – The dedicated instruction specified by the program cannot be executed by the specified module. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/ When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4003	**INSTRCT CODE. ERR.** The number of devices for the dedicated instruction specified by the program is incorrect. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/ When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4004	**INSTRCT CODE. ERR.** The device which cannot be used by the dedicated instruction specified by the program is specified. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN/ When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4010	**MISSING END INS.** There is no END (FEND) instruction in the program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4020	**CAN'T SET (P)** – The total points of the pointers used in the program exceeded 4096 points. – The total points of the local pointers used in the program exceeded the start number of the common pointer. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH QnU LCPU
	CAN'T SET (P) – The total points of the pointers used in the program exceeded 512 points. – The total points of the local pointers used in the program exceeded the start number of the common pointer. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN					Q00UJ/Q00U/ Q01U
4021	**CAN'T SET (P)** – The common pointer Nos. assigned to files overlap. – The local pointer Nos. assigned to files overlap. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4030	**CAN'T SET (I)** The allocation pointer Nos. assigned by files overlap. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4100	**OPERATION ERROR** The instruction cannot process the contained data. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QCPU LCPU
	OPERATION ERROR Access error of ATA card or SD memory card occurs by SP.FREAD/SP.FWRITE instructions. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Take noise reduction measures. – Reset and restart the CPU module. When the same error is displayed again, the ATA card or SD memory card has hardware failure. Contact your local Mitsubishi representative.				Qn(H) QnPH QnPRH QnU (except Q00UJ-, Q00U- and Q01UCPU)
	OPERATION ERROR The file being accessed by other functions was accessed with SP.FWRITE instruction. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Stop the file accessed with other functions to execute SP.FWRITE instruction. – Stop the access with other functions and the SP.FWRITE instruction execution at the same time.				QnU (except Q00UJ-, Q00U- and Q01UCPU) LCPU

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4101	**OPERATION ERROR** – The number of setting data dealt with the instruction exceeds the applicable range. – The storage data and constant of the device specified by the instruction exceeds the applicable range. – When writing to the host CPU shared memory, the write prohibited area is specified for the write destination address. – The range of storage data of the device specified by the instruction is duplicated. – The device specified by the instruction exceeds the range of the number of device points. – The interrupt pointer No. specified by the instruction exceeds the applicable range. – A link direct device, intelligent function module device, or cyclic transmission area device are specified for both (s) and (d) with the BMOV instruction. – There are no link direct devices (J□\□). ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QCPU LCPU
	OPERATION ERROR – The storage data of file register specified by the instruction exceeds the applicable range. – Or, file register is not set. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed					QnU (except Q00UJCPU) LCPU
	OPERATION ERROR Block data that crosses over the boundary between the internal user device and the extended data register (D) or extended link register is specified (including 32-bit binary, real number (single precision, double precision), indirect address, and control data). ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed					QnU LCPU

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4102	**OPERATION ERROR** In a multiple CPU system, the link direct device (J□\□), was specified for the network module under control of another station. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Delete from the program the link direct device which specifies the network module under control of another CPU. – Using the link direct device (J□\□), specify the network module under control of the host CPU.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)
	OPERATION ERROR – The network No. or station No. specified for the dedicated instruction is wrong. – The link direct device (J□\□) setting is incorrect. – The module No./ network No./number of character strings exceeds the range that can be specified. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.				QCPU
	OPERATION ERROR – The module number specified with a dedicated instruction is incorrect. – The module number, network number, or the number of character strings specified with a dedicated instruction exceeded the allowable range. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed					LCPU
	OPERATION ERROR The specification of character string (" ") specified by dedicated instruction cannot be used for the character string. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.				QnU LCPU
4103	**OPERATION ERROR** The configuration of the PID dedicated instruction is incorrect. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPRH QnU LCPU
4105	**OPERATION ERROR** PLOADP/PUNLOADP/PSWAPP instructions were executed while setting program memory check. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Delete the program memory check setting. – When using the program memory check, delete PLOADP/PUNLOADP/PSWAPP instructions.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QnPH (first 5 digits of serial No. is 07032 or higher)

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4107	**OPERATION ERROR** 33 or more multiple CPU dedicated instructions were executed from one CPU module. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Using the multiple CPU dedicated instruction completion bit, provide interlocks to prevent one CPU module from executing 33 or more multiple CPU dedicated instructions.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH Q00U/Q01U/ Q02U
4109	**OPERATION ERROR** With high speed interrupt setting PR, PRC, UDCNT1, UDCNT2, PLSY or PWM instruction is executed. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Delete the high-speed interrupt setting. When using high-speed interrupt, delete the PR, PRC, UDCNT1, UDCNT2, PLSY and PWM instructions.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) (first 5 digits of serial No. is 04012 or higher)
4111	**OPERATION ERROR** An attempt was made to perform write/read to/ from the CPU shared memory write/read disabled area of the host station CPU module with the instruction. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Q00/Q01 (Function version is B or later) QnU
4112	**OPERATION ERROR** A CPU module that cannot be specified with the multiple CPU dedicated instruction was specified. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Q00/Q01 (Function version is B or later) QnU (except Q00UJCPU)
4113	**OPERATION ERROR** – When the SP.DEVST instruction is executed, the number of writing to the standard ROM of the day exceeds the value specified by SD695. – A value outside the specified range is set to SD695. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Check that the number of execution of the SP.DEVST instruction is proper. – Execute the SP.DEVST instruction again the following day or later day. Or, arrange the value of SD695. – Correct the value of SD695 so that it does not exceed the range.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QnU LCPU
4116	**OPERATION ERROR** A built-in I/O instruction that is disabled with a parameter was executed. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Enable the built-in I/O function with parameters. – Prohibit executions of a built-in I/O instruction that is disabled with a parameter.	OFF/ON	Flicker/ ON	Stop/ Continue	LCPU
4120	**OPERATION ERROR** Since the manual system switching enable flag (special register SM1592) is OFF, manual system switching cannot be executed by the control system switching instruction (SP. CONTSW). ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	To execute control system switching by the SP. CONTSW instruction, turn ON the manual system switching enable flag (special register SM1592).	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QnPRH

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4121	**OPERATION ERROR** – In the separate mode, the control system switching instruction (SP. CONTSW) was executed in the standby system CPU module. – In the debug mode, the control system switching instruction (SP. CONTSW) was executed. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Reexamine the interlock signal for the SP.CONTSW instruction, and make sure that the SP.CONTSW instruction is executed in the control system only. (Since the SP. CONTSW instruction cannot be executed in the standby system, it is recommended to provide an interlock using the operation mode signal or like. Refer to the manual of the redundant system). – As the SP. CONTSW instruction cannot be executed in the debug mode, reexamine the interlock signal related to the operation mode.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QnPRH
4122	**OPERATION ERROR** – The dedicated instruction was executed to the module mounted on the extension base unit in the redundant system. – The instruction for accessing the intelligent function module mounted on the extension base unit from the standby system at separate mode was executed. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Delete the dedicated instruction for the module mounted on the extension base unit. – Delete the instruction for accessing the intelligent function module mounted on the extension base unit from the standby system.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QnPRH (first 5 digits of serial No. is 09012 or higher)
4130	**OPERATION ERROR** Instructions to read SFC step comment (S(P).SFCSCOMR) and SFC transition condition comment (S(P).SFCTCOMR) are executed for the comment file in ATA card. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Target comment file has to be other than the comment file in ATA card.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) (first 5 digits of serial No. is 07012 or higher) QnPH (first 5 digits of serial No. is 07032 or higher) QnPRH
4131	**OPERATION ERROR** The SFC program is started up by an instruction while another SFC program has not yet been completed. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Check the SFC program specified by the instruction. – Or, check the executing status of the SFC program.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QnU LCPU
4140	**OPERATION ERROR** Operation with non-allowed input data ("-0", unnormalized number, nonnumeric, ±∞) is performed. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QnU LCPU
4141	**OPERATION ERROR** Overflow occurs at operation. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	QnU LCPU
4200	**FOR NEXT ERROR** No NEXT instruction was executed following the execution of a FOR instruction. Alternatively, there are fewer NEXT instructions than FOR instructions. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4201	**FOR NEXT ERROR** A NEXT instruction was executed although no FOR instruction has been executed. Alternatively, there are more NEXT instructions than FOR instructions. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4202	**FOR NEXT ERROR** More than 16 nesting levels are programmed. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Keep nesting levels at 16 or under.	OFF	Flicker	Stop	QCPU LCPU
4203	**FOR NEXT ERROR** A BREAK instruction was executed although no FOR instruction has been executed prior to that. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4210	**CAN'T EXECUTE (P)** The CALL instruction is executed, but there is no subroutine at the specified pointer. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4211	**CAN'T EXECUTE (P)** There was no RET instruction in the executed subroutine program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4212	**CAN'T EXECUTE (P)** The RET instruction exists before the FEND instruction of the main routine program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4213	**CAN'T EXECUTE (P)** More than 16 nesting levels are programmed. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Keep nesting levels at 16 or under.	OFF	Flicker	Stop	QCPU LCPU
4220	**CAN'T EXECUTE (I)** Though an interrupt input occurred, the corresponding interrupt pointer does not exist. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4221	**CAN'T EXECUTE (I)** An IRET instruction does not exist in the executed interrupt program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4223	**CAN'T EXECUTE (I)** The IRET instruction exists before the FEND instruction of the main routine program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed **CAN'T EXECUTE (I)** – The IRET instruction was executed in the fixed scan execution type program. – The STOP instruction was executed in the fixed scan execution type program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU LCPU
4225	**CAN'T EXECUTE (I)** The interrupt pointer for the module mounted on the extension base unit is set in the redundant system. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Delete the setting of interrupt pointer for the module mounted on the extension base unit, since it cannot be used.	OFF	Flicker	Stop	QnPRH (first 5 digits of serial No. is 09012 or higher)
4230	**INST. FORMAT ERR** The number of CHK and CHKEND instructions is not equal. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Qn(H) QnPH
4231	**INST. FORMAT ERR** The number of IX and IXEND instructions is not equal. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QCPU
4235	**INST. FORMAT ERR** The configuration of the check conditions for the CHK instruction is incorrect. Alternatively, a CHK instruction has been used in a low speed execution type program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Qn(H) QnPH

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4350	**MULTI-COM.ERROR** – The multiple CPU high-speed transmission dedicated instruction used in the program specifies the wrong CPU module. Or, the setting in the CPU module is incompatible with the multiple CPU high-speed transmission dedicated instruction. – The reserved CPU is specified. – The uninstalled CPU is specified. – The head I/O number of the target CPU/16 (n1) is outside the range of 3EH to 3E3H. – The CPU module where the instruction cannot be executed is specified. – The instruction is executed in a single CPU system. – The host CPU is specified. – The instruction is executed without setting the "Use multiple CPU high speed communication". ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
4351	**MULTI-COM.ERROR** – The multiple CPU high-speed transmission dedicated instruction specified by the program cannot be executed to the specified target CPU module. – The instruction name is wrong. – The instruction unsupported by the target CPU module is specified. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
4352	**MULTI-COM.ERROR** The number of devices for the multiple CPU highspeed transmission dedicated instruction specified by the program is wrong. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
4353	**MULTI-COM.ERROR** The device which cannot be used for the multiple CPU high-speed transmission dedicated instruction specified by the program is specified. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
4354	**MULTI-COM.ERROR** The character string which cannot be handled by the multiple CPU high-speed transmission dedicated instruction is specified. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4355	**MULTI-COM.ERROR** The number of read/write data (number of request/receive data) for the multiple CPU high-speed transmission dedicated instruction specified by the program is not valid. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
4400	**SFCP. CODE ERROR** No SFCP or SFCPEND instruction in SFC program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH
4410	**CAN'T SET (BL)** The block number designated by the SFC program exceeds the range. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPRH QnU
4411	**CAN'T SET (BL)** Block number designations overlap in SFC program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	
4420	**CAN'T SET (S)** A step number designated in an SFC program exceeds the range. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	
4421	**CAN'T SET (S)** Total number of steps in all SFC programs exceed the maximum. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPRH QnU LCPU
		Increase the total number of step relays in the Device tab of the PLC Parameter dialog box.				QnU (first five digits of the serial number is 12052 or higher)
4422	**CAN'T SET (S)** Step number designations overlap in SFC program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPRH QnU LCPU

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4423	**CAN'T SET (S)** The total number of (maximum step No.+1) of each block exceeds the total number of step relays. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Correct the total number of step relays so that it does not exceed the total number of (maximum step No.+1) of each block. Increase the total number of step relays in the Device tab of the PLC Parameter dialog box.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnU LCPU QnU (first five digits of the serial number is 12052 or higher)
4430	**SFC EXE. ERROR** The SFC program cannot be executed. – The data of the block data setting is illegal. – The SFC data device of the block data setting is beyond the device setting range set in the PLC parameter. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** STOP → RUN	– Write the program to the CPU module again using the programming tool. – After correcting the setting of the SFC data device, write it to the CPU module. – After correcting the device setting range set in the PLC parameter, write it to the CPU module.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnU LCPU
4431	**SFC EXE. ERROR** The SFC program cannot be executed. – The block parameter setting is abnormal. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	
4432	**SFC EXE. ERROR** The SFC program cannot be executed. The structure of the SFC program is illegal. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	
4500	**SFCP. FORMAT ERR.** The numbers of BLOCK and BEND instructions in an SFC program are not equal. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH
4501	**SFCP. FORMAT ERR.** The configuration of the STEP* to TRAN* to TSET to SEND instructions in the SFC program is incorrect. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH LCPU
4502	**SFCP. FORMAT ERR.** The structure of the SFC program is illegal. STEPI* instruction does not exist in the block of the SFC program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** STOP → RUN	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPRH QnU LCPU

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4503	**SFCP. FORMAT ERR.** The structure of the SFC program is illegal: – The step specified in the TSET instruction does not exist. – In jump transition, the host step number was specified as the destination step number. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** STOP → RUN	– Write the program to the CPU module again using the programming tool. – Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPRH
	SFCP. FORMAT ERR. The structure of the SFC program is illegal: – The step specified in the TSET instruction does not exist. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When SFC program is executed					QnU LCPU
4504	**SFCP. FORMAT ERR.** The structure of the SFC program is illegal. The step specified in the TAND instruction does not exist. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When SFC program is executed	Write the program to the CPU module again using the programming tool.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPRH QnU LCPU
4505	**SFCP. FORMAT ERR.** The structure of the SFC program is illegal. In the operation output of a step, the SET Sn/BLmSn or RST Sn/BLmSn instruction was specified for the host step. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction is executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnU LCPU
4506	**SFCP. FORMAT ERR.** The structure of the SFC program is illegal. In a reset step, the host step number was specified as the destination step. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction is executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4600	**SFCP. OPE. ERROR** The SFC program contains data that cannot be processed. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	Qn(H) QnPH QnPRH
4601	**SFCP. OPE. ERROR** Exceeds device range that can be designated by the SFC program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	
4602	**SFCP. OPE. ERROR** The START instruction in an SFC program is preceded by an END instruction. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF/ON	Flicker/ ON	Stop/ Continue (can be set in the parameters at error occurrence)	
4610	**SFCP. EXE. ERROR** The active step information at presumptive start of the SFC program is incorrect. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** STOP → RUN	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem. The program is automatically subjected to an initial start.	ON	ON	Continue	Qn(H) QnPH QnPRH
4611	**SFCP. EXE. ERROR** Key-switch was reset during RUN when presumptive start was designated for SFC program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** STOP → RUN	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem. The program is automatically subjected to an initial start.	ON	ON	Continue	Qn(H) QnPH QnPRH
4620	**BLOCK EXE. ERROR** Startup was executed at a block in the SFC program that was already started up. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH QnU (first five digits of the serial number is 12052 or higher)
4621	**BLOCK EXE. ERROR** Startup was attempted at a block that does not exist in the SFC program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem. – Turn ON if the special relay SM321 is OFF.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPH QnPRH QnU LCPU
4630	**STEP EXE. ERROR** Startup was executed at a block in the SFC program that was already started up. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH

Tab. 13-4: *Error code list (4000 to 4999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
4631	**STEP EXE. ERROR** – Startup was attempted at the step that does not exist in the SFC program. Or, the step that does not exist in the SFC program was specified for end. – Forced transition was executed based on the transition condition that does not exist in the SFC program. Or, the transition condition for forced transition that does not exist in the SFC program was cancelled. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	– Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem. – Turn ON special relay SM321 if it is OFF.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) Qn(H) QnPH QnPRH QnU LCPU
4632	**STEP EXE. ERROR** There were too many simultaneous active steps in blocks that can be designated by the SFC program. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH QnU LCPU
4633	**STEP EXE. ERROR** There were too many simultaneous active steps in all blocks that can be designated. ■ **Collateral information** • Common Information: Program error location • Individual Information: — ■ **Diagnostic Timing** When instruction executed	Read the common information of the error using the programming tool, check error step corresponding to its numerical value (program error location), and correct the problem.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH QnU LCPU

Tab. 13-4: *Error code list (4000 to 4999)*

13.6 Error code list (5000 to 5999)

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
5000	**WDT ERROR** The scan time of the initial execution type program exceeded the initial execution monitoring time specified in the PLC RAS setting of the PLC parameter. ■ **Collateral information** • Common Information: Time (value set) • Individual Information: Time (value actually measured) ■ **Diagnostic Timing** Always	– Read the individual information of the error from the programming tool, check its value (time), and shorten the scan time. – Change the initial execution monitoring time or the WDT value in the PLC RAS setting of the PLC parameter. – Resolve the endless loop caused by jump transition.	OFF	Flicker	Stop	Qn(H) QnPH QnPRH QnU LCPU
	WDT ERROR – The power supply of the standby system is turned OFF. – The tracking cable is disconnected or connected without turning off or resetting the standby system. – The tracking cable is not secured by the connector fixing screws. ■ **Collateral information** • Common Information: Time (value set) • Individual Information: Time (value actually measured) ■ **Diagnostic Timing** Always	– Since power-off of the standby system increases the control system scan time, reset the WDT value, taking the increase of the control system scan time into consideration. – When the tracking cable is disconnected during operation, securely connect it and restart the CPU module. If the same error is displayed again, the tracking cable or CPU module has a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH
5001	**WDT ERROR** The scan time of the program exceeded the WDT value specified in the PLC RAS setting of the PLC parameter. ■ **Collateral information** • Common Information: Time (value set) • Individual Information: Time (value actually measured) ■ **Diagnostic Timing** Always	– Read the individual information of the error from the programming tool, check its value (time), and shorten the scan time. – Change the initial execution monitoring time or the WDT value in the PLC RAS setting of the PLC parameter. – Resolve the endless loop caused by jump transition.	OFF	Flicker	Stop	QCPU LCPU
	WDT ERROR – The power supply of the standby system is turned OFF. – The tracking cable is disconnected or connected without turning off or resetting the standby system. – The tracking cable is not secured by the connector fixing screws. ■ **Collateral information** • Common Information: Time (value set) • Individual Information: Time (value actually measured) ■ **Diagnostic Timing** Always	– Since power-off of the standby system increases the control system scan time, reset the WDT value, taking the increase of the control system scan time into consideration. – When the tracking cable is disconnected during operation, securely connect it and restart the CPU module. If the same error is displayed again, the tracking cable or CPU module has a hardware fault. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH

Tab. 13-5: *Error code list (5000 to 5999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
5010	**PRG. TIME OVER** The program scan time exceeded the constant scan setting time specified in the PLC RAS setting of the PLC parameter. ■ **Collateral information** • Common Information: Time (value set) • Individual Information: Time (value actually measured) ■ **Diagnostic Timing** Always	– Review the constant scan setting time. – Review the constant scan setting time and low speed program execution time in the PLC parameter so that the excess time of constant scan can be fully secured.	ON	ON	Continue	Qn(H) QnPH QnPRH QnU LCPU
	PRG. TIME OVER The low speed program execution time specified in the PLC RAS setting of the PLC parameter exceeded the excess time of the constant scan. ■ **Collateral information** • Common Information: Time (value set) • Individual Information: Time (value actually measured) ■ **Diagnostic Timing** Always					Qn(H) QnPH QnPRH
	PRG. TIME OVER The program scan time exceeded the constant scan setting time specified in the PLC RAS setting of the PLC parameter. ■ **Collateral information** • Common Information: Time (value set) • Individual Information: Time (value actually measured) ■ **Diagnostic Timing** Always	– Review the constant scan setting time in the PLC parameter so that the excess time of constant scan can be fully secured.				Q00J/Q00/Q01
5011	**PRG. TIME OVER** The scan time of the low speed execution type program exceeded the low speed execution watch time specified in the PLC RAS setting of the PLC parameter dialog box. ■ **Collateral information** • Common Information: Time (value set) • Individual Information: Time (value actually measured) ■ **Diagnostic Timing** Always	– Read the individual information of the error using the programming tool, check the numerical value (time) there, and shorten scan time if necessary. – Change the low speed execution watch time in the PLC RAS setting of the PLC parameter dialog box.	ON	ON	Continue	Qn(H) QnPH

Tab. 13-5: *Error code list (5000 to 5999)*

13.7 Error code list (6000 to 6999)

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
6000	**FILE DIFF.** In a redundant system, the control system and standby system do not have the same programs and parameters. The file type detected as different between the two systems can be checked by the file name of the error common information. – The program is different. (File name = ********.QPG) – The PLC parameters/network parameters/ redundant parameters are different. (File name = PARAM.QPA) – The remote password is different. (File name = PARAM.QPA) – The intelligent function module parameters are different. (File name = IPARAM.QPA) – The device initial values are different. (File name = ********.QDI) – The capacity of each write destination within the CPU for online pchange of multiple program blocks is different. (File name = MBOC.QMB) (This can be detected from the standby system of the redundant system.) ■ **Collateral information** • Common Information: File name • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/At tracking cable connection/At changing to backup mode/At completion of write during RUN/At Switching execution/At switching both systems into RUN	– Match the programs and parameters of the control system and standby system. – Verify the systems by either of the following procedures 1), 2) to clarify the differences between the files of the two systems, then correct a wrong file, and execute "Write to PLC" again. 1) After reading the programs/parameters of System A using a programming tool, verify them with those of System B. 2) Verify the programs/parameters saved in the offline environment with those written to the CPU modules of both systems. – When the capacity of each write destination within the CPU for online change of multiple program blocks is different between the two systems, take corrective action 1) or 2). 1) Using the memory copy from control system to standby system, copy the program memory from the control system to the standby system. 2) Format the CPU module program memories of both systems. (For the capacity of each write destination within the CPU for online change of multiple program blocks, set the same value to both systems.)	OFF	Flicker	Stop	QnPRH
6001	**FILE DIFF.** In a redundant system, the valid parameter drive settings (SW2, SW3) made by the DIP switches are not the same. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/At tracking cable connection/At operation mode change	Match the valid parameter drive settings (SW2, SW3) by the DIP switches of the control system and standby system.	OFF	Flicker	Stop	QnPRH
6010	**OPE. MODE DIFF.** The operational status of the control system and standby system in the redundant system is not the same. (This can be detected from the standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	Synchronise the operation statuses of the control system and standby system.	ON	ON	Continue	QnPRH
6020	**OPE. MODE DIFF.** At power ON/reset, the RUN/STOP switch settings of the control system and standby system are not the same in a redundant system. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	Set the RUN/STOP switches of the control system and standby system to the same setting.	OFF	Flicker	Stop	QnPRH

Tab. 13-6: *Error code list (6000 to 6999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
6030	**UNIT LAY. DIFF.** – In a redundant system, the module configuration differs between the control system and standby system. – The network module mode setting differs between the two systems. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: Module No. • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/At tracking cable connection/At operation mode change	– Match the module configurations of the control system and standby system. – In the redundant setting of the network parameter dialog box, match the mode setting of System B to that of System A.	OFF	Flicker	Stop	QnPRH
6035	**UNIT LAY. DIFF.** In a redundant system, the CPU module model name differs between the control system and standby system. (This can be detected from the standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/At tracking cable connection/At operation mode change	Match the model names of the control system and standby system.	OFF	Flicker	Stop	QnPRH
6036	**UNIT LAY. DIFF.** A difference in the remote I/O configuration of the MELSECNET/H multiplexed remote I/O network between the control system and standby system of a redundant system was detected. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: Module No. • Individual Information: — ■ **Diagnostic Timing** Always	Check the network cables of the MELSECNET/H multiplexed remote I/O network for disconnection.	OFF	Flicker	Stop	QnPRH
6040	**CARD TYPE DIFF.** In a redundant system, the memory card installation status (installed/not installed) differs between the control system and standby system. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	Match the memory card installation statuses (installed/not installed) of the control system and standby system.	OFF	Flicker	Stop	QnPRH
6041	**CARD TYPE DIFF.** In a redundant system, the memory card type differs between the control system and standby system. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	Match the memory card types of the control system and standby system.	OFF	Flicker	Stop	QnPRH
6050	**CAN'T EXE. MODE.** The function inexecutable in the debug mode or operation mode (backup/separate mode) was executed. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	Execute the function in the debug mode or operation mode (backup/separate mode).	ON	ON	Continue	QnPRH

Tab. 13-6: *Error code list (6000 to 6999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
6060	**CPU MODE DIFF.** In a redundant system, the operation mode (backup/separate) differs between the control system and standby system. (This can be detected from the standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/At tracking cable connection/	Match the operation modes of the control system and standby system.	OFF	Flicker	Stop	QnPRH
6061	**CPU MODE DIFF.** In a redundant system, the operation mode (backup/separate) differs between the control system and standby system. (This can be detected from the standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	Match the operation modes of the control system and standby system.	OFF	Flicker	Stop	QnPRH
6062	**CPU MODE DIFF.** Both System A and B are in the same system status (control system). (This can be detected from the system B of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset/At tracking cable connection/	Power the CPU module (System B) which resulted in a stop error, OFF and then ON.	OFF	Flicker	Stop	QnPRH
6100	**TRK. TRANS. ERR.** – An error (e.g. retry limit exceeded) occurred in tracking data transmission. (This error may be caused by tracking cable removal or other system power-off (including reset).) – The error occurred at a startup since the redundant system startup procedure was not followed. ■ **Collateral information** • Common Information: Tracking transmission data classification • Individual Information: — ■ **Diagnostic Timing** Always	– Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	ON	ON	Continue	QnPRH
6101	**TRK. TRANS. ERR.** – A timeout error occurred in tracking (data transmission). (This error may be caused by tracking cable removal or other system power-off (including reset).) – The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: Tracking transmission data classification • Individual Information: — ■ **Diagnostic Timing** Always	– Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	ON	ON	Continue	QnPRH

Tab. 13-6: *Error code list (6000 to 6999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
6102	**TRK. TRANS. ERR.** A data sum value error occurred in tracking (data reception). (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	ON	ON	Continue	QnPRH
6103	**TRK. TRANS. ERR.** – A data error (other than sum value error) occurred in tracking (data reception). (This error may be caused by tracking cable removal or other system power-off (including reset).) – The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	ON	ON	Continue	QnPRH
6105	**TRK. TRANS. ERR.** – An error (e.g. retry limit exceeded) occurred in tracking (data transmission). (This error may be caused by tracking cable removal or other system power-off (including reset).) – The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: Tracking transmission data classification • Individual Information: — ■ **Diagnostic Timing** Always	– Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	ON	ON	Continue	QnPRH
6106	**TRK. TRANS. ERR.** – A timeout error occurred in tracking (data transmission). (This error may be caused by tracking cable removal or other system power-off (including reset).) – The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: Tracking transmission data classification • Individual Information: — ■ **Diagnostic Timing** Always	– Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	ON	ON	Continue	QnPRH
6107	**TRK. TRANS. ERR.** A data sum value error occurred in tracking (data reception). (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	ON	ON	Continue	QnPRH

Tab. 13-6: *Error code list (6000 to 6999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
6108	**TRK. TRANS. ERR.** – A data error (other than sum value error) occurred in tracking (data reception). (This error may be caused by tracking cable removal or other system power-off (including reset).) – The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	ON	ON	Continue	QnPRH
6110	**TRK. SIZE ERROR** The tracking capacity exceeded the allowed range. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: Tracking capacity excess error factor • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	Reexamine the tracking capacity.	ON	ON	Continue	QnPRH
6111	**TRK. SIZE ERROR** The control system does not have enough file register capacity for the file registers specified in the tracking settings. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	Switch to the file registers of which capacity is greater than the file registers specified in the tracking settings.	ON	ON	Continue	QnPRH
6112	**TRK. SIZE ERROR** File registers greater than those of the standby system were tracked and transmitted from the control system. (This can be detected from the standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** When an END instruction executed	Switch to the file registers of which capacity is greater than the file registers specified in the tracking settings.	ON	ON	Continue	QnPRH
6120	**TRK. CABLE ERR.** – A start was made without the tracking cable being connected. – A start was made with the tracking cable faulty. – As the tracking hardware on the CPU module side was faulty, communication with the other system could not be made via the tracking cable. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	Make a start after connecting the tracking cable. If the same error still occurs, this indicates the tracking cable or CPU module side tracking transmission hardware is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnPRH

Tab. 13-6: *Error code list (6000 to 6999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
6130	**TRK. DISCONNECT** – The tracking cable was removed. – The tracking cable became faulty while the CPU module is running. – The CPU module side tracking hardware became faulty. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– If the tracking cable was removed, connect the tracking cable to the connectors of the CPU modules of the two systems. – When the error is not resolved after connecting the tracking cable to the connectors of the CPU modules of the two systems and resetting the error, the tracking cable or CPU module side tracking hardware is faulty. Contact your local Mitsubishi representative.	ON	ON	Continue	QnPRH
6140	**TRK.INIT. ERROR** – The other system did not respond during initial communication at power ON/reset. – The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	– Power the corresponding CPU module OFF and then ON again, or reset it and then unreset. If the same error still occurs, this indicates the CPU module is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	OFF	Flicker	Stop	QnPRH
6200	**CONTROL EXE.** The standby system has been switched to the control system in a redundant system. (Detected by the CPU that was switched from the standby system to the control system). Since this error code does not indicate the error information of the CPU module but indicates its status, the error code and error information are not stored into SD0 to SD26, but are stored into the error log every system switching. (Check the error information by reading the error log using the programming tool.) ■ **Collateral information** • Common Information: Reason(s) for system switching • Individual Information: — ■ **Diagnostic Timing** Always	—	ON	OFF	No error	QnPRH
6210	**STANDBY** The control system has been switched to the standby system in a redundant system. (Detected by the CPU that was switched from the standby system to the control system). Since this error code does not indicate the error information of the CPU module but indicates its status, the error code and error information are not stored into SD0 to SD26, but are stored into the error log every system switching. (Check the error information by reading the error log using the programming tool.) ■ **Collateral information** • Common Information: Reason(s) for system switching • Individual Information: — ■ **Diagnostic Timing** Always	—	ON	OFF	No error	QnPRH

Tab. 13-6: *Error code list (6000 to 6999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
6220	**CAN'T SWITCH** System switching cannot be executed due to standby system error/ tracking cable error/ online module change in execution at separate mode. Causes for switching system at control system are as follows: – System switching by SP. CONTSW instruction – System switching request from network module ■ **Collateral information** • Common Information: Reason(s) for system switching • Individual Information: Reason(s) for system switching failure ■ **Diagnostic Timing** At switching execution	– Check the status of the standby system and resolve the error. – Complete the online module change.	ON	ON	No error	QnPRH
6300	**STANDBY SYS. DOWN** Any of the following errors was detected in the backup mode. – The standby system has not started up in the redundant system. – The standby system has developed a stop error in the redundant system. – The CPU module in the debug mode was connected to the operating control system. (This can be detected from the control system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– Check whether the standby system is on or not, and if it is not on, power it on. – Check whether the standby system has been reset or not, and if it has been reset, unreset it. – Check whether the standby system has developed a stop error or not, and if it has developed the error, remove the error factor and restart it. – When the CPU module in the debug mode was connected to the control system operating in the backup mode, make connection so that the control system and standby system are combined correctly.	ON	ON	Continue	QnPRH
6310	**CONTROL SYS. DOWN** Any of the following errors was detected in the backup mode. – The control system has not started up in the redundant system. – The control system has developed a stop error in the redundant system. – The CPU module in the debug mode was connected to the operating standby system. – The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	– The standby system exists but the control system does not exist. – Check whether the system other than the standby system is on or not, and if it is not on, power it on. – Check whether the system other than the standby system has been reset or not, and if it is has been reset, unreset it. – Check whether the system other than the standby system has developed a stop error or not, and if has developed the error, remove the error factor, set the control system and standby system to the same operating status, and restart. – When the CPU module in the debug mode was connected to the control system operating in the backup mode, make connection so that the control system and control system are combined correctly. – Confirm the redundant system startup procedure, and execute a startup again.	OFF	Flicker	Stop	QnPRH
6311 6312	**CONTROL SYS. DOWN** – As consistency check data have not been transmitted from the control system in a redundant system, the other system cannot start as a standby system. – The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the standby system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	– Replace the tracking cable. If the same error still occurs, this indicates the CPU module is faulty. Contact your local Mitsubishi representative. – Confirm the redundant system startup procedure, and execute a startup again.	OFF	Flicker	Stop	QnPRH

Tab. 13-6: *Error code list (6000 to 6999)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
6313	**CONTROL SYS. DOWN** The control system detected the error of the system configuration and informed the standby system (host system) in the redundant system. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	Restart the system after checking that the connection between base unit and the system configuration (type/number/parameter of module) are correct.	OFF	Flicker	Stop	QnPRH (first five digits of the serial number of the CPU module is 09102 or higher)
6400	**PRG. MEM. CLEAR** The memory copy from control system to standby system was executed, and the program memory was cleared. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At execution of the memory copy from control system to standby system	After the memory copy from control system to standby system is completed, switch power OFF and then ON, or make a reset.	OFF	Flicker	Stop	QnPRH
6410	**MEM.COPY EXE]** The memory copy from control system to standby system was executed. (This can be detected from the control system of the redundant system.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At execution of the memory copy from control system to standby system	After the memory copy from control system to standby system is completed, switch power OFF and then ON, or make a reset.	ON	ON	Continue	QnPRH
6500	**TRK. PARA. ERROR** The file register file specified in the tracking setting of the PLC parameter dialog box does not exist. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset	Read the individual information of the error using the programming tool, and check and correct the drive name and file name. Create the specified file.	OFF	Flicker	Stop	QnPRH
6501	**TRK. PARA. ERROR** The file register range specified in the device detail setting of the tracking setting of the PLC parameter dialog box exceeded the specified file register file capacity. ■ **Collateral information** • Common Information: File name/Drive name • Individual Information: Parameter No. ■ **Diagnostic Timing** At power ON/At reset	Read the individual information of the error using the programming tool, and increase the file register capacity.	OFF	Flicker	Stop	QnPRH

Tab. 13-6: *Error code list (6000 to 6999)*

13.8 Error code list (7000 to 10000)

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
7000	**MULTI CPU DOWN** – In the operating mode of a multiple CPU system, a CPU error occurred at the CPU where "All station stop by stop error of CPU " was selected. – In a multiple CPU system, a CPU module incompatible with the multiple CPU system was mounted. – CPU modules other than CPU No.1 were removed from the base unit in operation, or reset. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** Always	– Read the individual information of the error using the programming tool, identify the error of the CPU module, and remove the error. – Remove the CPU module incompatible with the multiple CPU system from the main base unit. – Check the mounting status of CPU modules other than CPU No.1 and whether the CPU modules were reset.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)
	MULTI CPU DOWN In a multiple CPU system, CPU other than CPU No.1 cannot be started up due to stop error of the CPU No.1 at power-on, which occurs to CPU No.2 to No.4. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	Read the individual information of the error using the programming tool, identify the error of the CPU module, and remove the error.				
7002	**MULTI CPU DOWN** – There is no response from the target CPU module in a multiple CPU system during initial communication. – In a multiple CPU system, a CPU module incompatible with the multiple CPU system was mounted. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Reset the CPU module and RUN it again. If the same error is displayed again, this suggests the hardware fault of any of the CPU modules. Contact your local Mitsubishi representative.) – Remove the CPU module incompatible with the multiple CPU system from the main base unit. Or, replace the CPU module incompatible with the multiple CPU system with the compatible one.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH
	MULTI CPU DOWN There is no response from the target CPU module in a multiple CPU system during initial communication. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the CPU module and RUN it again. If the same error is displayed again, this suggests the hardware fault of any of the CPU modules. Contact your local Mitsubishi representative.				QnU (except Q00UJCPU)
7003	**MULTI CPU DOWN** There is no response from the target CPU module in a multiple CPU system at initial communication stage. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Reset the CPU module and RUN it again. If the same error is displayed again, this suggests the hardware fault of any of the CPU modules. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH
7004	**MULTI CPU DOWN** In a multiple CPU system, a data error occurred in communication between the CPU modules. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** Always	– Check the system configuration to see if modules are mounted in excess of the number of I/O points. – When there are no problems in the system configuration, this indicates the CPU module hardware is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later)

Tab. 13-7: *Error code list (7000 to 10000)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
7010	**MULTI EXE. ERROR** – In a multiple CPU system, a faulty CPU module was mounted. – In a multiple CPU system, a CPU module incompatible with the multiple CPU system was mounted. (The CPU module compatible with the multiple CPU system was used to detect an error.) – In a multiple CPU system, any of the CPU No. 2 to 4 was reset with power ON. (The CPU whose reset state was cancelled was used to detect an error.) ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Read the individual information of the error using the programming tool, and replace the faulty CPU module. – Replace the CPU module with the one compatible with the multiple CPU system. – Do not reset any of the No. 2 to 4 CPU modules. – Reset CPU No. 1 and restart the multiple CPU system.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)
	MULTI EXE. ERROR The PC CPU module-compatible software package (PPC-DRV-01) whose version is 1.06 or earlier is used in a multiple CPU system. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Change the version of the PC CPU module-compatible software package (PPC-DRV-01) to 1.07 or later.				Q00/Q01 (Function version is B or later)
	MULTI EXE. ERROR The Q172(H)CPU(N) or Q173(H)CPU(N) is mounted on the multiple CPU high-speed main base unit (Q3□DB). (This may result in a module failure.) ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Replace the Q172(H)CPU(N) and Q173(H)CPU(N) with the Motion CPU compatible with the multiple CPU high-speed main base unit.				Qn(H) (first 5 digits of serial No. is 09082 or higher) QnPH (first 5 digits of serial No. is 09082 or higher)
	MULTI EXE. ERROR The Universal model QCPU (except Q02UCPU) and Q172(H)CPU(N) are mounted on the same base unit. (This may result in a module failure.) ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Check the QCPU and Motion CPU that can be used in a multiple CPU system, and change the system configuration.				Qn(H) (first 5 digits of serial No. is 09082 or higher) QnPH (first 5 digits of serial No. is 09082 or higher)
7011	**MULTI EXE. ERROR** Either of the following settings was made in a multiple CPU system. – Multiple CPU automatic refresh setting was made for the inapplicable CPU module. – "I/O sharing when using multiple CPUs" setting was made for the inapplicable CPU module. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Correct the settings.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) QnU (except Q00UJCPU)

Tab. 13-7: *Error code list (7000 to 10000)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
7011	**MULTI EXE. ERROR** The system configuration for using the Multiple CPU high speed transmission function is not met. – The QnUCPU is not used for the CPU No.1. – The Multiple CPU high speed main base unit Q3□BD is not used. – Points other than 0 is set to the send range for the CPU module incompatible with the multiple CPU high speed transmission function. – Points other than 0 is set to the send range for the CPU module incompatible with the multiple CPU. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Change the system configuration to meet the conditions for using the Multiple CPU high speed transmission function. – Set the send range of CPU, that does not correspond to multiple CPU compatible area, at 0 point, when performing automatic refreshing in multiple CPU compatible area.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
7013	**MULTI EXE. ERROR** The Q172(H)CPU(N) or Q173(H)CPU(N) is mounted to the CPU slot or slots 0 to 2. (The module may break down.) ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Check the QCPU and Motion CPU that can be used in a multiple CPU system, and change the system configuration. – Remove the Motion CPU incompatible with the multiple CPU system.	OFF	Flicker	Stop	QnU
7020	**MULTI CPU ERROR** In the operating mode of a multiple CPU system, an error occurred in the CPU where "system stop" was not selected. (The CPU module where no error occurred was used to detect an error.) ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** Always	Read the individual information of the error using the programming tool, check the error of the CPU module resulting in CPU module fault, and remove the error.	ON	ON	Continue	Q00/Q01 (Function version is B or later) Qn(H) (Function version is B or later) QnPH QnU (except Q00UJCPU)
7030	**CPU LAY. ERROR** An assignment error occurred in the CPU-mountable slot (CPU slot, I/O slot 0, 1) in excess of the number of CPU modules specified in the multiple CPU setting of the PLC parameter dialog box. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Set the same value to the number of CPU modules specified in the multiple CPU setting of the PLC parameter dialog box and the number of mounted CPU modules (including CPU (empty)). – Make the type specified in the I/O assignment setting of the PLC parameter dialog box consistent with the CPU module configuration.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnU
7031	**CPU LAY. ERROR** An assignment error occurred within the range of the number of CPUs specified in the multiple CPU setting of the PLC parameter dialog box. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Set the same value to the number of CPU modules specified in the multiple CPU setting of the PLC parameter dialog box and the number of mounted CPU modules (including CPU (empty)). – Make the type specified in the I/O assignment setting of the PLC parameter dialog box consistent with the CPU module configuration.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnU
7032	**CPU LAY. ERROR** The number of CPU modules mounted in a multiple CPU system is wrong. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Configure a system so that the number of mountable modules of each CPU module does not exceed the maximum number of mountable modules specified in the specification.	OFF	Flicker	Stop	Q00/Q01 (Function version is B or later) QnU (except Q00UJCPU)

Tab. 13-7: *Error code list (7000 to 10000)*

Error code	Error contents and cause	Corrective action	LED status		CPU status	Corresponding CPU
			RUN	ERR.		
7035	**CPU LAY. ERROR** The CPU module has been mounted on the inapplicable slot. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	Mount the CPU module on the applicable slot.	OFF	Flicker	Stop	Q00J/Q00/Q01 (Function version is B or later) QnPRH QnU
7036	**CPU LAY. ERROR** The host CPU No. set by the multiple CPU setting and the host CPU No. determined by the mounting position of the CPU module are not the same. ■ **Collateral information** • Common Information: Module No. (CPU No.) • Individual Information: — ■ **Diagnostic Timing** At power ON/ At reset	– Mount the mounting slot of the CPU module correctly. – Correct the host CPU No. set by the multiple CPU setting to the CPU No. determined by the mounting position of the CPU module.	OFF	Flicker	Stop	QnU (except Q00UJ-, Q00U-, Q01U- and Q02UCPU)
8031	**INCORRECT FILE** The error of stored file (enabled parameter file) is detected. ■ **Collateral information** • Common Information: — • Individual Information: File diagnostic information ■ **Diagnostic Timing** At power ON/ At reset/STOP → RUN/ At writing to programmable controller	Write the file shown as SD17 to SD22 of individual information to the drive shown as SD16(L) of individual information, and turn ON from OFF the power supply of the CPU module or cancel the reset. If the same error is displayed again, this indicates the CPU module hardware is faulty. Contact your local Mitsubishi representative.	OFF	Flicker	Stop	QnU LCPU
9000	**F****** Annunciator (F) was set ON. (The "*****" portion of the error message indicates an annunciator number.) ■ **Collateral information** • Common Information: Program error location • Individual Information: Annunciator number ■ **Diagnostic Timing** When instruction executed	Read the individual information of the error using the programming tool, and check the program corresponding to the numerical value (annunciator number).	ON	ON/OFF USER LED: ON	Continue	QCPU LCPU
9010	**<CHK> ERR ***_***** Error detected by the CHK instruction. ■ **Collateral information** • Common Information: Program error location • Individual Information: Error number ■ **Diagnostic Timing** When instruction executed	Read the individual information of the error using the programming tool, and check the program corresponding to the numerical value (error number).	ON	OFF USER LED: ON	Continue	Qn(H) QnPH QnPRH
9020	**BOOT OK** Storage of data onto ROM was completed normally in automatic write to standard ROM. (BOOT LED also flickers.) ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** At power ON/At reset	Use the DIP switches to set the valid parameter drive to the standard ROM. Then, switch power on again, and perform boot operation from the standard ROM.	OFF	Flicker	Stop	Qn(H) (Function version is B or later) QnPH QnPRH
10000	**CONT.UNIT ERROR** In the multiple CPU system, an error occurred in the CPU module other than the Process CPU/ High performance model QCPU. ■ **Collateral information** • Common Information: — • Individual Information: — ■ **Diagnostic Timing** Always	Check the details of the generated error by connecting to the corresponding CPU module.	OFF	Flicker	Continue	Qn(H) (Function version is B or later) QnPH

Tab. 13-7: *Error code list (7000 to 10000)*

13.9 Error codes returned to request source

If an error occurs at communication request from a programming tool, intelligent function module, or network system, the CPU module returns an error code to the request source.

This error code is not stored in SD0 because the error is not the one detected by the selfdiagnostic function of the CPU module.

When the request source is a programming tool, a message and an error code are displayed on the programming tool. When the request source is an intelligent function module or network system, the CPU module returns an error code corresponding to the requested processing.

Error code	Error Item	Error Details	Corrective Action
4000H	Common error	Serial communication sum check error	• Connect the serial communication cable correctly. • Take noise reduction measures.
4001H		Unsupported request was executed.	• Check the command data of the MC protocol, etc. • Check the CPU module model name selected in the programming tool.
4002H			
4003H		Command for which a global request cannot be performed was executed.	Check the command data of the MC protocol, etc.
4004H		Any operation for the CPU module is prohibited by the system protect function provided against the following events. • The system protect switch is ON. • The CPU module is starting.	• Set the system protect switch of the CPU module to OFF. • Perform operation again after the CPU module has completed starting.
4005H		The volume of data handled according to the specified request is too large.	Check the command data of the MC protocol, etc.
4006H		Serial communication could not be initialized.	• Check with the external device maker for the support condition. • Check the CPU module model name selected in the programming tool.
4008H		The CPU module is BUSY. (The buffer is not vacant).	After the free time has passed, reexecute the request.
4010H	CPU mode error	Since the CPU module is running, the request contents cannot be executed.	Execute after setting the CPU module to STOP status.
4013H		Since the CPU module is not in a STOP status, the request contents cannot be executed.	
4021H	CPU file related error	The specified drive memory does not exist or there is an error.	• Check the specified drive memory status. • After backing up the data in the CPU module, execute programmable controller memory format.
4022H		The file with the specified file name or file No. does not exist.	Check the specified file name and file No.
4023H		The file name and file No. of the specified file do not match.	Delete the file and then recreate the file.
4024H		The specified file cannot be handled by a user.	Do not access the specified file.
4025H		The specified file is processing the request from another programming tool.	Complete the current processing and then send the request again.
4026H		Any of the file password, drive keyword, or file password 32 set in advance to the target drive (memory) must be specified.	Specify any of the file password, drive keyword, or file password 32 set in advance to the target drive (memory) and then access.
4027H		The specified range is larger than the file size range.	Check the specified range and access within that range.
4028H		The same file already exists.	Reexecute after changing the file name.
4029H		The specified file capacity cannot be obtained.	Revise the specified file contents. Or reexecute after cleaning up and reorganizing the specified drive memory.
402AH		The specified file is abnormal.	After backing up the data in the CPU module, execute programmable controller memory format.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
402BH	CPU file related error	The request contents cannot be executed in the specified drive memory.	Execute again after setting the CPU module to the STOP status. Execute programmable controller memory arrangement to increase the continuous free space of the drive (memory).
402CH		The requested operation cannot be executed currently.	Execute again after a while.
4030H	CPU device specified error	The specified device name cannot be handled.	Check the specified device name.
4031H		The specified device No. is outside the range.	• Check the specified device No. • Check the device assignment parameters of the CPU module.
4032H		There is a mistake in the specified device qualification. The unusable device name (TS, TC, SS, SC, CS, CC) must be specified in MC protocol random reading, random writing (word), monitor registration and monitor command.	• Check the specified device qualification method. • Check the specified device name.
4033H		Writing cannot be done because the specified device is for system use.	Do not write the data in the specified device, and do not turn on or off.
4034H		Cannot be executed since the completion device for the dedicated instruction cannot be turned ON.	Since the completion device for the target station CPU module cannot be turned ON by the SREAD instruction/ SWRITE instruction, execute again after setting the operating status of the target station CPU module to the RUN status.
4040H	Intelligent function module specification error	The request contents cannot be executed in the specified intelligent function module.	Check whether the specified module is the intelligent function module having the buffer memory.
4041H		The access range exceeds the buffer memory range of the specified intelligent function module.	Check the header address and access number of points and access using a range that exists in the intelligent function module.
4042H		The specified intelligent function module cannot be accessed.	• Check that the specified intelligent function module is operating normally. • Check the specified module for a hardware fault.
4043H		The intelligent function module does not exist in the specified position.	Check the I/O No. of the specified intelligent function module.
4044H		A control bus error occurred during access to the intelligent function module.	Check the specified intelligent function module and other modules and base units for a hardware fault.
4050H	Protect error	The request contents cannot be executed because the memory card write protect switch is on.	Turn off the memory card write protect switch.
4051H		The specified device memory cannot be accessed.	Check the following and take countermeasures. • Is the memory one that can be used? • Is the specified drive memory correctly installed?
4052H		The specified file attribute is read only so the data cannot be written.	Do not write data in the specified file. Or change the file attribute.
4053H		An error occurred when writing data to the specified drive memory.	Check the specified drive memory. Or reexecute write after changing the corresponding drive memory.
4054H		An error occurred when deleting the data in the specified drive memory.	Check the specified drive memory. Or re-erase after replacing the corresponding drive memory.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4060H	Online registration error	The online debug function (such as online change, sampling trace, and monitoring condition setting) and the data logging function are being executed with another programming tool.	• Finish the operation of another programming tool and then execute the function. • If the operation of another programming tool is on hold, resume with that programming tool to complete the operation, and then execute the function again.
4061H		Communication of the online debug function was unsuccessful.	• Register an online debug function (such as online change, sampling trace, and monitoring condition setting) and then establish a communication. • Execute again after checking the communication route such as the communication cable.
4063H		The registered number of locked files exceeded the maximum value.	Finish the file access from another programming tool, and then execute again.
4064H		Settings for the online debug function (such as online change, sampling trace, and monitoring condition setting) and for the data logging function are incorrect.	• Check the settings for the online debug function (such as online change, sampling trace, and monitoring condition setting) and data logging function. • Execute again after checking the communication route such as the communication cable.
4065H		The device allocation information differs from the parameter.	Check the device assignment parameters of the CPU module or the device assignment of the request data.
4066H		The specified drive keyword, file password, or file password 32 is incorrect.	• Correct the drive keyword of the specified drive. • Correct the file password of the specified file. • Correct the file password 32 of the specified file.
4067H		Monitor communication was unsuccessful.	• Check the system area capacity of the user setting specified for programmable controller memory format. • Execute again after checking the communication route such as the communication cable.
4068H		Operation is disabled because it is being performed with another programming tool.	Finish the operation of another programming tool and then execute again.
406AH		The drive (memory) number that cannot be handled (other than 0 to 4) was specified.	Check the specified drive and specify the correct drive.
4070H	Circuit inquiry error	The program not yet corrected and the one corrected by online program change are different.	Read the program from the CPU module to match it with that of the programming tool, and then execute online change again.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4080H	Other errors	Request data error	Check the request data that has been specified.
4081H		The sort subject cannot be detected.	Check the data to be searched.
4082H		The specified command is executing and therefore cannot be executed.	Complete the processing for a request from another programming tool and then execute the command again.
4083H		An attempt was made to perform operation for the program not registered to the parameters.	Register the program to the parameters.
4084H		The specified pointer P, I did not exist.	Check the pointer P, I in the specified data.
4085H		Pointer P, I cannot be specified because the program is not specified in the parameter.	Specify pointer P, I after registering the program to be executed in the parameter.
4086H		Pointer P, I has already been added.	Check the pointer No. to be added and make correction.
4087H		Trying to specify too many pointer P, I.	Check the specified pointer P, I and make a correction.
4088H		• The specified step number is not at the head of the instruction. • Contents of the program is different from those stored in the CPU module.	• Check and correct the specified step No. • Read the program from the CPU module to match it with that of the programming tool, and then execute online change again.
4089H		An attempt was made to insert/delete the END instruction by online program change.	• Check the specified program file contents. • Write the program after setting the CPU module to the STOP status.
408AH		The file capacity was exceeded by the write during Run.	• Check the capacity of the specified program file. • Write the program after setting the CPU module to the STOP status.
408BH		The remote request cannot be executed.	• Reexecute after the CPU module is in a status where the mode request can be executed. • For remote operation, set the parameter to "Enable remote reset".
408CH		An attempt was made to remote-start the program, which uses the CHK instruction, as a low speed program.	The program including the CHK instruction cannot be executed at low speed. Execute again after checking the program.
408DH		The instruction code that cannot be handled exists.	• Check whether the model of the used CPU module is correct or not. • The program where online change was attempted includes the instruction that cannot be handled by the CPU module specified for the project. Check the program and delete the instruction.
408EH		The write step is illegal.	• Write the program after setting the CPU module to the STOP status. • The starting position of online program change is not specified with the correct program step No. Check whether the programming tool supports the model and version of the CPU module that is specified for the project.
40A0H	SFC device specification error	A block No. outside the range was specified.	Check the setting contents and make a correction.
40A1H		A number of blocks that exceeds the range was specified.	Check the number of settings and make a correction.
40A2H		A step No. that is outside the range was specified.	Check the setting contents and make a correction.
40A3H		Step range limit exceeded	Check the number of settings and make a correction.
40A4H		The specified sequence step No. is outside the range.	Check the setting contents and make a correction.
40A5H		The specified device is outside the range.	Check the number of settings and make a correction.
40A6H		The block specification pattern and step specification pattern were wrong.	Check the setting contents and make a correction.
40B0H	SFC file related error	The drive (memory) specified in SFC file operation is wrong.	Check the setting contents and make a correction.
40B1H		The SFC program specified in SFC file operation does not exist.	Check the specified file name and make a correction.
40B2H		The program specified in SFC file operation is not an SFC program.	Check the specified file name and make a correction.
40B3H		Using online program change of SFC, an attempt was made to execute rewrite operation of the "SFC dedicated instruction", such as the "STEP start instruction or transition start instruction", that shows an SFC chart. (SFC dedicated instruction cannot be written during RUN.)	Write the program after setting the CPU module to the STOP status.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4100H	Other errors	CPU module hardware fault	Change the CPU module.
4101H		Serial communication connection was executed for a different CPU module series.	Check the CPU module series.
4102H		An attempt was made to erase the Flash ROM during use of the file register.	Execute again after setting the CPU module to the STOP status.
4103H		The instruction written during RUN is wrong or illegal.	Execute online program change again, or write the program after setting the CPU module to the STOP status.
4105H		CPU module internal memory hardware fault	Change the CPU module.
4106H		The command cannot be executed since the CPU module is performing system initialization processing.	Execute the operation again after the CPU module has started.
4107H		An attempt was made to perform the operation of a function unavailable for the target CPU module model name.	Do not execute the function unsupported by the target CPU module.
4108H		Operation cannot be made normally by device monitor/test.	Execute device monitor/test again. Before execution, check that access is not made to the access prohibited area.
4109H		The specified operation cannot be executed since the monitoring, set the condition for other application in same computer, is in execution.	Execute the request again after deregistering the monitoring condition on the same screen.
410AH		The specified command cannot be executed because of online program change.	Execute the request again after the online program change.
410BH		The registration of monitoring condition was canceled because of online program change.	Execute the registration of monitoring condition again after the online program change.
4110H	CPU module error	Since the CPU module is in a stop error status, it cannot execute the request.	Execute the request again after resetting the CPU module.
4111H		The requested operation cannot be performed since the other CPU modules have not yet started in the multiple CPU system.	Execute the request again after the other CPU modules have started.
4121H	File-related errors	The specified drive (memory) or file does not exist.	Execute again after checking the specified drive (memory) or file.
4122H		The specified drive (memory) or file does not exist.	Execute again after checking the specified drive (memory) or file.
4123H		The specified drive (memory) is abnormal.	Execute programmable controller memory format to make the drive (memory) normal. In the case of the Flash ROM, check the data to be written to the Flash ROM, and write them to the Flash ROM.
4124H		The specified drive (memory) is abnormal.	Execute programmable controller memory format to make the drive (memory) normal. In the case of the Flash ROM, check the data to be written to the Flash ROM, and write them to the Flash ROM.
4125H		The specified drive (memory) or file is performing processing.	Execute again after a while.
4126H		The specified drive (memory) or file is performing processing.	Execute again after a while.
4127H		File password mismatch	Execute again after checking the file password.
4128H		File password mismatch with copy destination	Execute again after checking the file password.
4129H		Cannot be executed since the specified drive (memory) is ROM.	Execute again after changing the target drive (memory).
412AH		Cannot be executed since the specified drive (memory) is ROM.	Execute again after changing the target drive (memory).
412BH		The specified drive (memory) is write-inhibited.	Execute again after changing the write inhibit condition or drive (memory).
412CH		The specified drive (memory) is write-inhibited.	Execute again after changing the write inhibit condition or drive (memory).
412DH		The specified drive (memory) does not have enough free space.	Execute again after increasing the free space of the drive (memory).
412EH		The specified drive (memory) does not have enough free space.	Execute again after increasing the free space of the drive (memory).
412FH		The drive (memory) capacity differs between the drive (memory) copy destination and copy source.	Execute again after checking the drive (memory) copy destination and copy source.
4130H		The drive (memory) type differs between the drive (memory) copy destination and copy source.	Execute again after checking the drive (memory) copy destination and copy source.
4131H		The file name of the file copy destination is the same as that of the copy source.	Execute again after checking the file names.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4132H	File-related errors	The specified number of files does not exist.	Execute again after checking the specified data.
4133H		The specified drive (memory) has no free space.	Execute again after increasing the free space of the drive (memory).
4134H		The attribute specification data of the file is wrong.	Execute again after checking the specified data.
4135H		The date/time data of the peripheral device (personal computer) is beyond the range.	Execute again after checking the clock setting of the peripheral device (personal computer).
4136H		The specified file already exists.	Execute again after checking the specified file name.
4137H		The specified file is read-only.	Execute again after changing the condition of the specified file.
4138H		Simultaneously accessible files exceeded the maximum.	Execute again after decreasing file operations.
4139H		The size of the specified file has exceeded that of the existing file.	Execute again after checking the size of the specified file.
413AH		The specified file has exceeded the already existing file size.	Execute again after checking the size of the specified file.
413BH		The same file was simultaneously accessed from different programming tools.	Execute again after a while.
413CH		The specified file is write-inhibited.	Execute again after changing the file condition.
413DH		The specified file capacity cannot be secured.	Execute again after increasing the capacity of the specified drive (memory).
413EH		Operation is disabled for the specified drive (memory).	Execute again after changing the target drive (memory).
413FH		The file is inhibited from write to the standard RAM.	Execute again after changing the specified drive (memory).
414AH	Intelligent function module specification error	Operation was executed for the intelligent function module of the non-control group in the multiple CPU system.	Execute the operation again from the control CPU of the target module.
414CH		The I/O address of the specified CPU module is wrong.	Execute again after checking the I/O address of the specified CPU module.
4150H	File-related errors	An attempt was made to format the drive protected by the system.	Do not format the target drive (memory) as it cannot be formatted.
4151H		An attempt was made to delete the file protected by the system.	Do not delete the target file as it cannot be deleted.
4160H	Online registration error	The registered number of forced inputs/outputs exceeded the maximum value.	Deregister the unused forced inputs/outputs.
4165H		The multiple-block online change system file does not exist.	Execute again after securing the area that enables multiple-block online change at the time of programmable controller memory format.
4166H		Online change (files) is disabled because it is being executed from the same source.	Due to unsuccessful online change (files) previously occurred for some reason (example: communication failure), the processing is kept incomplete. Forcibly perform another online change (files).
4167H		Online change (files) is disabled because it is being performed from another source.	Due to unsuccessful online change (files) from another source previously occurred for some reason (example: communication failure), the processing is kept incomplete. If online change (files) is not being performed by any other programming tool, forcibly perform another online change (files).
4168H		The registered number of device test with executing condition exceeds 32.	Deregister the device test with executing condition in CPU module, or decrease the number of registering device test with executing condition at one time.
4169H		The device test with executing condition has never been registered.	Deregister the device test with executing condition after checking the registered number of device test with executing condition in CPU module.
416AH		The specified executing condition does not exist. (Device test with executing condition)	Check whether the specified executing conditions (program, step No. operation timing, device name) in deregistering are registered.
416BH		The specified program is SFC program. (Device test with executing condition)	Check the specifying program name in de/registering the device test with executing condition.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4170H	Ethernet I/F Error	The password is wrong.	Check the specified remote password, then execute the lock/unlock processing of the remote password again.
4171H		The port for communication use is at remote password locking status.	Execute communication after unlocking the remote password processing.
4174H		Requested for a wrong module to unlock remote password.	• Stop transmitting from several modules simultaneously when setting a remote password and using User Datagram Protocol (UDP) in MELSOFT connection. • The MELSOFT connection can be used with Transmission Control protocol (TCP) when setting a remote password.
4176H		Communication error occurred in direct connection.	• Do not specify the direct connection when using other connection setting. • Do not turn off the CPU power during communication, reset the power, and plug out the cable in direct connection.
4178H		• File operation is disabled because the File Transfer Protocol (FTP) function is in operation. • Online operation requiring a file access is performed with a programming tool while the File Transfer Protocol (FTP) function is in operation.	Retry after the operation for FTP function is completed.
4180H		System error.(The setting data in OS is abnormal.)	• Ensure that the power supply module and the CPU module are properly installed to the base unit. • Ensure that the operating environment for the system is met the general specifications of the CPU module. • Check whether the power capacity is sufficient. • Reset the CPU module. If the same error code is displayed again, the cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
4181H		Transmission to the receiving modules is unsuccessful.	• Check the receiving module operation. • Check the status of the lines, such as cables, hubs and routes, connected to receiving modules. • Some line packets may be engaged. Retry to communicate a little while later. • The receiving module may have no free space in receive area (TCP window size is small). Check whether the receiving module processes receive data, or whether the CPU module does not send unnecessary data. • Check whether the settings of the subnet mask pattern and the default router IP address of the CPU module and the receiving modules are correct, or whether the class of the IP address is correct.
4182H		Communication with receiving modules caused a time-out error.	• Check the receiving module operation. • Check the status of the lines such as a cable, a hub and a route to receiving modules. • Some line packets may be engaged. Retry to communicate a little while later.
4183H		Communication with receiving modules was interrupted.	• Check the receiving module operation. • Check the status of the lines such as cables, hubs and routes connected to receiving modules.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4184H	Ethernet I/F Error	• Communication processing buffer has run out of space due to consecutive reception of request messages using the MC protocol. • Communication processing buffer has ran out of space because received data read is not performed or cannot keep up with the volume. • Communication processing is disabled due to insufficient space in the communication processing buffer.	• For MC protocol, send a request after receiving a response to the previous request. • For socket communication, enable received data read. • For socket communication, decrease the number of data sent from the target device.
4185H		• Connection to the target device is disconnected before sending a response using the MC protocol. • Connection to the target device is disconnected during communication.	• For MC protocol, keep the connection until a response is sent. • Keep the connection until a sequence of communication is completed. • Other error such as 4184H may be the cause. If any other error has occurred, take corrective action of that error.
4186H		System error (The argument data in OS is abnormal.)	• Check whether the power supply module and the CPU module are properly installed to the base unit. • Check whether the operating environment for the system satisfies the general specifications of the CPU module. • Check whether the power capacity is sufficient. • Reset the CPU module. If the same error code is displayed again, the cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
4187H		System error (The wait processing in OS is abnormal.)	
4188H		System error (The data length in OS is abnormal.)	
4189H		System error (The protocol information in OS is abnormal.)	
418AH		System error (The address data of communicating module in OS is abnormal.)	
418BH		System error (The protocol information in OS is abnormal.)	
418CH		System error (The protocol specification processing in OS is abnormal.)	
418DH		System error (The typed data in OS is abnormal.)	
418EH		System error (The expedited data processing in OS is abnormal.)	
418FH 4190H		System error (The protocol information in OS is abnormal.)	
4191H		System error (The address data of communicating module in OS is abnormal.)	
4192H		System error (The host module address processing in OS is abnormal.)	
4193H to 4196H		System error (The transmission processing in OS is abnormal.)	
4197H 4198H		System error (The connection processing in OS is abnormal.)	
4199H		System error (The connection termination processing is abnormal.)	
419AH		System error (The connection processing in OS is abnormal.)	
419BH		System error (The connection termination processing is abnormal.)	
419CH 419DH		System error (The processing order in OS is abnormal.)	

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
419EH	Ethernet I/F Error	Connection to the module was unsuccessful or interrupted.	• Check the receiving module operation. • Check the status of the lines such as a cable, a hub and a route to receiving modules • Retry to connect a little while later, if the error occurred in communication.
419FH		System error (The I/O control processing is abnormal.)	• Check whether the power supply module and the CPU module are properly installed to the base unit. • Check whether the operating environment for the system is met the general specifications of the CPU module. • Check whether the power capacity is sufficient. • Reset the CPU module. • If the same error code is displayed again, the cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative, explaining a detailed description of the problem.
41A0H		Data cannot be sent since the target device has no free space in receive area (TCP window size is zero).	• Retry after a while • Check the behavior of the target device. • Check whether the target device is processing data receiving. • Check whether any unnecessary data are being sent from the CPU module side.
41A1H	Ethernet I/F socket communication error	The port number setting for the CPU module is incorrect.	Correct the port number.
41A2H		The port number setting for the target device is invalid.	
41A3H		• For TCP/IP, the same Host station port No. is specified as MC protocol. • For TCP/IP, a connection with the same host station No. and the same destination port No. are already specified to one communication target.	• Specify a port number that is not duplicated with that of MC protocol. • Correct both or either of the port numbers of the CPU module and target device to avoid duplication.
41A4H		• For UDP/IP, the same Host station port No. is specified as MC protocol. • For UDP/IP, the specified host station No. is duplicated.	• Specify a port number that is not duplicated with that of MC protocol. • Correct the port number of the CPU module to avoid duplication.
41A5H		The IP address setting of the target device for OPEN processing is invalid.	Correct the IP address. Specify A, B, or C for the class.
41A6H		Connection was not established in OPEN processing for TCP connection.	• Check the behavior of the target device. • Check OPEN processing of the target device. • Correct the port number of the CPU module and the IP address, port number, and open processing method of the target device. • Check whether the cables are securely connected.
41A8H		Data length is out of permissible range (2046 bytes for the Built-in Ethernet port QCPU whose serial number (first five digits) is "12051" or lower and 10238 bytes for "12052" or higher)	• Correct the data length. • If the data is longer than the range, split the data and send them. • When the data length is 2047 to 10238 bytes, use the Built-in Ethernet port QCPU whose serial number (first five digits) is "12052" or higher.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
41ABH	Ethernet I/F socket communication error	Failed to send data due to resend timeout in TCP.	• Review the IP address and Ethernet address of the target device. • Check whether the target device has the ARP function. If not, communicate with the one that has the ARP function. • Check the behavior of the target device. • The line may be congested with packets. Resend data after a while. • Check whether the cables are securely connected.
41ACH		• The target device cannot be found. • The TCP connection is disconnected by the target device. • The Fullpassive device rejects the communication and the TCP connection is disconnected.	• Check the behavior of the target device. • Check whether the cables are securely connected. • Check whether the target IP address setting of the Fullpassive device and the IP address of the Active device are matched.
41ADH		Data cannot be send due to no connection or disconnection of the cable.	• Check whether the cables are securely connected. • Check the line status by PING test from the target device. • Check the CPU module for error by conducting a self-diagnostic test (resetting the CPU module). • Send data again.
41B4H		The connection number setting is invalid.	• Specify the connection No. within 1 to 16. • Check whether "Socket communication" is selected for "Open system" parameter.
41B6H		The specified connection has already completed OPEN processing.	Perform CLOSE processing and then OPEN processing.
41B7H		The specified connection has not completed OPEN processing.	Reexecute after OPEN processing is completed.
41B9H		• Contents of control data is not correct. • Open instruction was executed through open settings parameter even though parameters are not set for "Open settings".	• Correct the contents of the control data. • Configure the open settings parameters or execute the OPEN instruction through control data.
41C1H	File-related errors	The format information data of the specified drive (memory) is abnormal.	The file information data may be corrupted. After backing up the data in the CPU module, execute programmable controller memory format.
41C2H		File open specification data for file access is wrong.	Execute again after checking the specification data.
41C3H		Simultaneously accessible files exceeded the maximum.	Execute again after decreasing file operations.
41C4H		Simultaneously accessible files exceeded the maximum.	Execute again after decreasing file operations.
41C5H		The specified file does not exist.	Execute again after checking the file.
41C7H		The specified file or drive (memory) does not exist.	Execute again after checking the file or drive (memory).
41C8H		The size of the specified file has exceeded that of the existing file.	Execute again after checking the size of the specified file. If the error recurs after re-execution, the file information data may be corrupted. After backing up the data in the CPU module, execute programmable controller memory format.
41C9H		Access to the file sector was unsuccessful. The format information data of the target drive (memory) is abnormal.	After backing up the data in the CPU module, execute programmable controller memory format.
41CAH		Access to the file sector was unsuccessful. The format information data of the target drive (memory) is abnormal.	After backing up the data in the CPU module, execute programmable controller memory format.
41CBH		The file name is specified in a wrong method.	Execute again after checking the file name.
41CCH		The specified file or subdirectory does not exist.	Execute again after checking the name of the file and subdirectory.
41CDH		Access to the file is inhibited by the system.	Do not access the specified file.
41CEH		The file cannot be written because the specified file is read-only.	The specified file is write-protected. Execute again after checking the attribute.
41CFH		The specified drive (memory) has been used exceeding the capacity.	Execute again after checking the drive (memory) capacity.
41D0H		The specified drive (memory) has no free space. Or the number of files in the directory of the specified drive (memory) has exceeded the maximum.	• Execute again after increasing the free space of the drive (memory). • Execute again after deleting file(s) in the drive (memory).

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
41D1H	File-related errors	• The file name is specified in a wrong method. • The SD memory card is being disabled by SM606 (SD memory card forced disable instruction).	• Execute again after checking the file name. If the error recurs after re-execution, the file information data may be corrupted. After backing up the data in the CPU module, format the CPU module memory. • Cancel the SD memory card forced disable instruction.
41D4H		The size of the specified file has exceeded that of the existing file.	Execute again after checking the size of the specified file. If the error recurs after re-execution, the file information data may be corrupted. After backing up the data in the CPU module, execute programmable controller memory format.
41D5H		The file of the same name exists.	Forcibly execute the request, or execute after changing the file name.
41D6H		The format information data of the specified drive (memory) is abnormal.	The file information data may be corrupted. After backing up the data in the CPU module, execute programmable controller memory format.
41D7H		The format information data of the specified drive (memory) is abnormal.	The file information data may be corrupted. After backing up the data in the CPU module, execute programmable controller memory format.
41D8H		The specified file is being accessed.	Execute again after a while.
41DFH		The specified drive (memory) is write-protected.	Execute again after canceling the write protect of the specified drive (memory).
41E0H		The specified drive (memory) is abnormal or does not exist.	• Execute again after checking whether the memory card has been installed. • After backing up the data, execute programmable controller memory format.
41E1H		Access to the flash ROM was unsuccessful.	• After backing up the data, execute write to PLC (Flash ROM). • Execute again after checking whether the specified drive is the Flash ROM card and whether the memory card size is correct.
41E4H		Access to the memory card was unsuccessful.	• Execute again after checking whether the memory card has been installed. • Execute again after replacing the memory card. • After backing up the data, execute programmable controller memory format.
41E7H		The format information data of the specified drive (memory) is abnormal.	The file information data may be corrupted. After backing up the data in the CPU module, execute programmable controller memory format.
41E8H		The format information data of the specified drive (memory) is abnormal.	The file information data may be corrupted. After backing up the data in the CPU module, execute programmable controller memory format.
41E9H		The specified file is being accessed.	Execute again after some time.
41EBH		The file name is specified in a wrong method.	Execute again after checking the file name.
41ECH		The file system of the specified drive (memory) is logically corrupted.	The file information data may have been corrupted. After backing up the data in the CPU module, execute programmable controller memory format.
41EDH		The specified drive (memory) does not have continuous free space. (The free space for file is sufficient but the continuous free space is insufficient.)	Execute again after deleting unnecessary files or executing programmable controller memory arrangement.
41EFH		Creation of power failure backup for the specified drive (memory) was unsuccessful.	Execute again after checking whether the memory card has been installed.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
41F0H	File-related errors	The power failure backup data of the specified drive (memory) are corrupted.	Execute again after checking whether the memory card has been installed.
41F1H		The power failure backup for the specified drive (memory) has a repair command.	Execute again after checking whether the memory card has been installed.
41F2H		Operation cannot be performed since the specified drive (memory) is Flash ROM.	Execute again after checking the specified drive (memory). When performing operation for the Flash ROM, use write to PLC (Flash ROM).
41F3H		File size is larger than 4Gbyte-2byte.	Specify a smaller value for the file size when creating a file or changing the file size. Alternatively, divide the file so that each file size is smaller.
41F4H		Since the operation prohibited by the system is performed, the requested processing cannot be performed.	Since the operation is prohibited by the system, the file operation is not performed.
41F8H		The data is being accessed from another programming tool.	PLC write to the program memory or transfer to the backup memory is in execution. Access the data again after checking that the above-mentioned function has been completed.
41F9H		The data is being accessed from another programming tool.	Another device data saving was executed during execution. Access the data again after the current one is completed.
41FAH		Program was written beyond the area where the program can be executed.	Execute again after reducing either the already written program or newly written program.
41FBH		The specified file is already being manipulated from the programming tool.	Execute again after the currently performed operation is completed.
41FCH		An attempt was made to erase the drive (memory) being used.	The specified drive (memory) is being used and cannot be erased.
41FDH		There are no data written to the Flash ROM.	Write a file by executing write to PLC (Flash ROM).
41FEH		• The memory card has not been inserted. • Or the SD memory card lock switch is not slid down. • The SD memory card is being disabled by SM606 (SD memory card forced disable instruction).	• Insert or re-insert the memory card. • Slide the SD memory card lock switch down. • Cancel the SD memory card forced disable instruction.
41FFH		The memory card type differs.	Check the memory card type.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4200H	Online module change-related error	The requested processing cannot be performed because online module change is disabled by parameter setting.	Do not send the request where this error occurred. Alternatively, enable online module change by parameter setting and send the request again.
4201H		The requested processing cannot be performed because online module change is enabled by parameter setting.	Do not send the request where this error occurred. Alternatively, disable online module change by parameter setting and send the request again.
4202H		The requested processing cannot be performed since an online module change is being performed.	Make a request again after completion of the online module change.
4203H		The module mounted on the main base unit cannot be replaced online since the extension base unit is mounted.	Take following procedures to replace the module mounted on the main base unit. • Switch the system where the target module to be replaced is mounted to the standby system. • Turn OFF power supply of the standby system. • Replace the target module.
4204H		The specified module of the extension base unit cannot be replaced online since it is connected to the standby system.	Change the connection destination to the control system and perform the online module change again. (This corrective action can be made to the module mounted on the extension base unit only.)
4210H		The specified head I/O number is outside the range.	When making a request, specify the head I/O number of the module that will be changed online.
4211H		An online module change request is abnormal.	Check the command used to make a request.
4212H		An online module change is already being made for other equipment.	Make a request again after completion of the online module change, or continue after changing the connection path.
4213H		The specified head I/O number differs from the one registered for online module change.	When making a request, specify the head I/O number of the module being changed online.
4214H		The specified module differs from the one changed online.	Make a request again after mounting the module that is the same as the one changed online.
4215H		The specified module does not exist.	When making a request, specify the head I/O number of the module that will be changed online, or make a request again after mounting the module.
4216H		The specified module is faulty.	Make a request again after changing the module.
4217H		There is no response from the specified module.	Continue the online module changing operation.
4218H		The specified module is incompatible with online module change.	Do not make a request where an error occurred, or make a request again to the module compatible with online module change.
4219H		The specified module is mounted on the extension base unit of the type that requires no power supply module.	Do not make a request to any modules mounted on the extension base unit of the type that requires no power supply module and the main base unit.
421AH		The specified module is not in a control group.	Make a request to the CPU module that controls the specified module.
421BH		An error occurred in the setting of the initial setting parameter of the intelligent function module.	Resume processing after checking the contents of the intelligent function module buffer memory.
421CH		Cannot be executed as the parameter file has been rewritten.	Operation cannot be performed. Operation is interrupted.
421DH		System switching occurs during the online module change.	Connect the programming tool to the new control system to check the status of the online module change. According to the status of online module change, take procedures for it.
421EH	Online module change-related error	The information of the online module change cannot be sent to the standby system. When the system switching occurs during the online module change, the online module change may not be continued.	The tracking cable may be faulty or the standby system may have an error. • Check the mounting status of the tracking cable or replace the tracking cable. • Check the status of the standby system. When a stop error was detected by the standby system, perform troubleshooting.
421FH		The module mounted on the extension base unit cannot be replaced online when the connection destination is set to the standby system in the separate mode.	• Set the connection destination of a programming tool to the present control system. • Perform the online module change to the module mounted on the extension base unit again.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4240H	Redundant system-related error	Any of the following unsupported operations was executed for the standby system. • Operation mode change • System switching • Memory copy from control system to standby system	Execute the operation again after changing the transfer setup to the control system.
4241H		Communication cannot be made since the standby system has been powered off or reset or is in a user watchdog timer error or CPU module hardware fault status.	Request communication after powering on the standby system or setting its Reset switch to the neutral position.
4242H		Communication with the standby system cannot be made since the tracking cable is faulty or disconnected.	Execute again after checking the tracking cable for disconnection or changing it for a normal one.
4243H		The command cannot be executed since the standby system is in stop error.	Execute again after removing the stop error of the standby system.
4244H		The command cannot be executed since the operation status differs from that of the standby system.	Execute again after placing the standby system in the same operation status (RUN/STOP) as the control system.
4245H		Other system CPU module status error	Check that the other system CPU module has normally started up and that the tracking cable is connected.
4246H		The command cannot be executed since operation mode (separate/backup) change or system (control/standby system) switching is being executed.	Execute again after the operation mode change or system switching being executed is completed.
4247H		Memory copy from control system to standby system is already being executed.	Execute again after memory copy from control system to standby system is completed. Check the following and take corrective action. • Is SM1596 of the control system or standby system ON? (ON: Memory copy being executed) Execute again after SM1596 has turned OFF since it is turned OFF by the system on completion of memory copy. • Is SM1597 of the control system ON? (ON: Memory copy completed) Execute again after turning OFF SM1597 of the control system
4248H		• Communication was made during system switching. • The system specified in the transfer setup (request destination module I/O number) does not exist.	• Execute again after system switching. • After checking whether the specified system exists or not, restart communication.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4249H	Redundant system-related error	The redundant system is not established. (Control system/standby system or System A/ System B not yet definite)	• Normally start the system as the redundant system. (Make communication again after establishing the system.) • Execute again after changing the transfer setup (request destination module I/O number) to "No settings have been made" (03FFH).
424AH		The command that could not be processed was executed when the transfer setup (request destination module I/O number) is Control system/ Standby system/System A/System B.	Execute again after changing the transfer setup (request destination module I/O number) to "No settings have been made" (03FFH).
424BH		The command cannot be executed since system switching is inhibited by the manual system switching enable flag (SM1592).	Manual system switching is inhibited by the manual system switching enable flag (SM1592). Execute again after turning ON SM1592.
424CH		The specified command cannot be executed during online program change operation.	Execute again after the online program change operation is finished.
424DH		The transfer setup or function unavailable for the debug mode was used.	• Execute again after changing to the backup mode. • Execute again after changing the transfer setup (request destination module I/O number) to System A or control system.
424EH		The control system/standby system specifying method is not supported.	This function cannot be executed since it is not supported.
424FH		System switching was executed by the other condition during execution of system switching by the programming tool.	Although system switching was executed from the programming tool, system switching was executed first by the other condition. Check the system for any problem and execute the operation again as necessary.
4250H		Sum check error occurred in tracking communication.	Execute communication again after changing the tracking cable. If the same error recurs after the tracking cable is changed, the cause is the hardware fault of the CPU module. (Please consult your local Mitsubishi representative, explaining a detailed description of the problem.)
4251H		The command cannot be executed in the separate mode.	Execute again after changing to the backup mode.
4252H		System switching was not executed since an error occurred in the redundant system compatible network module of the standby system.	By monitoring SD1690 (other system network module No. that issued system switching request), identify the faulty redundant-compatible intelligent module of the standby system, then remove the module fault, and execute again.
4253H		Since a communication error or system switching occurred during online program change to the control system CPU module, online program change to the standby CPU module cannot be executed.	Since a communication error or system switching occurred during execution of online program change to the control system CPU module, online program change redundant tracking was suspended. Execute online program change again after confirming that communication with the control system CPU module and standby system CPU module can be normally made. If it takes time for the communication between the programming tool and either the control system CPU module or standby system CPU module, change the value in SD1710 (standby system online start waiting time) so that errors may be avoided.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4254H	Redundant system-related error	The command cannot be executed since an error was detected in the tracking communication hardware.	The tracking cable may not be connected correctly, or the tracking communication hardware of the CPU module may be faulty. Check the connection status of the tracking cable. If the condition is not restored to normal after the cable connection status is corrected, the possible cause is the hardware fault of the CPU module.
4255H		The command cannot be executed since tracking communication is being prepared.	Tracking communication is being prepared when it is connected. Execute the operation again after a while (about 1 second).
4256H		The command cannot be executed since a time-out error occurred in tracking communication.	The tracking cable may not be connected correctly, or the tracking communication hardware of the CPU module may be faulty. Check the connection status of the tracking cable. If the condition is not restored to normal after the cable connection status is corrected, the possible cause is the hardware fault of the CPU module.
4257H		The command cannot be executed since the host system CPU module is in a watchdog timer error or CPU module hardware fault status.	Execute again after checking the host system status.
4258H		Operation mode being changed (from backup mode to separate mode)	Execute again after completing the operation mode change by changing the status from STOP to RUN using the RUN/ STOP switch of the CPU module whose RUN LED is flickering or by remote operation.
4259H		Operation mode is being changed with another programming tool in the communication route different from the one currently used.	Execute again in the same communication route as the one where the operation mode change was executed.
425BH		Although the communication was made via the intelligent function module mounted on the extension base unit, the combination of the connection destination specification (Redundant CPU specification) and the command is unsupported.	Change the combination of the connection destination specification and the command to the supported combination.
425CH		System switching cannot be made since the module mounted on the extension base unit is being replaced online.	Switch systems after the online module change has been completed.
425DH		Operation mode cannot be changed since the module mounted on the extension base unit is being replaced online.	Change the operation mode after the online module change has been completed.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4270H	Data logging (To check logging status, use LCPU Logging Configuration Tool. For operation, refer to the MELSEC LCPU Module User's Manual (Data Logging Function)	Data logging is being performed (logging, saving the logging data, completing, on hold, or in error) with a different memory.	Register the data logging to the memory where a data logging is being performed. Alternatively, stop the data logging being performed and register again.
4271H		The specified data logging is already being performed (logging, saving the logging data, completing, on hold, or in error).	Stop the data logging. Alternatively, write, delete, or register to the setting number where no data logging is being performed.
4272H		The trigger logging with "Device" specified as a trigger condition is being performed (logging, saving the logging data, completing, on hold, or in error).	Change the trigger condition. Alternatively, stop the trigger logging being performed (logging, saving the logging data, completing, on hold, or in error) with "Device" specified as the trigger condition, and then register.
4273H		The data logging function cannot be executed because the sampling trace function is being performed.	Hold the sampling trace to register the data logging.
4274H		Trigger loggings have registered exceeding the number of trigger loggings collectable in the data logging buffer.	• Increase the capacity of the data logging buffer. • Reduce the number of records set for the trigger logging.
4275H		Auto logging is being performed.	After the auto logging, replace the SD memory card and execute again.
4276H		The specified command cannot be executed because the data logging function is being performed (i.e. logging, saving the logging data, completing, on hold, or in error).	Stop the data logging and then execute the command.
4277H		The number of stored files has exceeded the value set in advance.	Delete files or change the storage destination and then register.
4278H		The number of stored files has reached to the maximum.	The number of stored files has reached to the maximum of FFFFFFFFH. Delete files or change the storage destination and then register.
427AH		Common setting file does not exist.	• Write the common settings to the target memory. • Register the data logging to the memory where the common settings are stored.
427BH	Maintenance	A data logging is being performed (logging in execution, logging data are being saved, completing, on hold, or in error) in the same file storage destination.	Stop the data logging destined for the same storage and then register. Alternatively, change the storage destination of the file and then register.
4330H		The processing is being executed from the same source.	Ensure that the CPU module change function (backup/ restoration) with SD memory card is not being performed from the same source, and then execute again.
4332H		The specified command cannot be executed because the CPU module change function (backup/ restoration) with SD memory card is being performed.	Finish the CPU module change function (backup/ restoration) with SD memory card, and then execute again.
4333H		Not ready for backup.	Complete the preparation for backup and then execute again.
4334H		Backup file does not exist.	Insert a memory card with a backup file stored and then execute again.
4335H		The specified function cannot be executed because latch data are being backed up.	Complete the latch data backup function and then execute again.
4336H		The specified function cannot be executed because a FTP client is being FTP-connected to the CPU module via the built-in Ethernet port.	Disconnect all FTP connections to the CPU module and then execute again.
4337H		Module error collection file does not exist.	Power off and then on or reset the CPU module and then execute again.
4338H		Readout of module error collection data has been failed when opening the screen to display the data or when updating the data.	Retry the operation. Increase the number of module error collections to be stored.
4339H		Readout of module error collection data was failed because the function is disabled by parameter settings.	Enable the module error collection function by parameter settings and then execute again.

Tab. 13-8: *Error codes returned to request source*

Error code	Error Item	Error Details	Corrective Action
4400H	Security	The file protected by a password 32 has been opened without using the password.	Configure a correct password, authorize it, and then access.
4401H		• Password authorization for the file password 32 has failed in accessing when it is required for reading. • The password format for the password 32 is incorrect.	• Configure a correct password for reading, authorize it, and then access. • Access the file with the method that is applicable to the file password 32.
4402H		• Password authorization for the file password 32 has failed in accessing when it is required for writing. • The password format for the password 32 is incorrect.	• Configure a correct password for writing, authorize it, and then access. • Access the file with the method that is applicable to the file password 32.
4403H		Both passwords for reading and for writing that are set upon Create, Change, Delete, or Disable do not match the previous ones.	Configure a correct password for both reading and for writing, authorize them, and then access.
4404H		A file error was detected before or after performing Create, Change, or Delete.	• Format the drive including the target file by formatting the CPU module memory. • Write the target file to the CPU module again, and then register or cancel the file password 32 again.
4A00H	Link-related error	• Access to the specified station cannot be made since the routing parameters are not set to the start source CPU module and/or relay CPU module. • For routing via a multiple CPU system, the control CPU of the network module for data routing has not started. • When System A/System B is not yet identified in a redundant system configuration, communication was made with the other station via the network module.	• Set to the related stations the routing parameters for access to the specified station. • Retry after a while, or start communication after confirming that the system for data routing has started. • In a redundant system configuration, connect the tracking cable, start System A/System B normally, and then restart communication.
4A01H		The network of the No. set to the routing parameters does not exist.	Check and correct the routing parameters set to the related stations.
4A02H		Access to the specified station cannot be made.	• Check the network module/link module for an error or offline status. • Check to see if the network number/PC number setting has no mistake.
4A03H		A request for network test was issued.	Check the request data of the MC protocol, etc.
4B00H	Target-related error	An error occurred in the access destination or relay station, or the specified transfer setup (request destination module I/O number) is illegal.	• Take corrective action after checking the error that occurred at the specified access destination or the relay station to the accessed station. • Check the transfer setup (request destination module I/O number or programmable controller number) in the request data of the MC protocol, etc.
4B01H		The target is not the No. 1 CPU of the multiple CPU system.	Execute the request for the No. 1 CPU of the multiple CPU system.
4B02H		The request is not addressed to the CPU module.	Perform operation for the module that can execute the specified function.
4B03H		• The specified route is not supported by the specified CPU module version. • The communication target CPU module is not mounted.	Check whether the specified route is supported or not.
4B04H		The specified transfer setup (request destination module I/O number) is not supported.	In the target setup, an illegal value is set as the head I/O number of the target module.
4C00H	Multiple CPU-related error	The specified device is unavailable for the motion CPU or outside the device range.	Check the request data contents.
4C08H		There are a total of 33 or more DDWR and DDRD requests.	Execute again after reducing the number of DDWR and DDRD requests to be executed simultaneously.
4C09H		The specification of the requested CPU module No. is illegal.	Check the request data contents.

Tab. 13-8: *Error codes returned to request source*

A Appendix A

A.1 Definition of the processing times

The operation processing time is the total of the following:

- Total of each instruction processing time.
- The END processing time. This time consists of the time to execute the END instruction, the MELSECNET related refresh time, the processing time for the communication with peripheral devices, and the time for serial communication.
- Processing time for the function that increases the scan time

Refer to the following manual(s) for the END processing time, I/O refresh time, and processing time for the function that increases the scan time.

- QnUCPU User's Manual (Functions Explanation, Program Fundamentals)
- Qn(H)/QnPH/QnPRHCPU User's Manual (Functions Explanation, Program Fundamentals)
- MELSEC-L CPU Module User's Manual (Functions Explanation, Program Fundamentals)

A.2 Processing times for MELSEC System Q CPUs

The tables on the following pages contain the processing times of all instructions.

The according processing times depend on the values of source and destination data. The values contained in the following tables should therefore be taken as a set of general guidelines to processing time rather than as being strictly accurate.

When the instruction is not executed the processing time is calculated as follows:

Type of CPU	Processing time when the instruction is not executed (μs)
Q00JCPU	0.20 x (Number of steps for each instruction + 1)
Q00CPU	0.16 x (Number of steps for each instruction + 1)
Q01CPU	0.10 x (Number of steps for each instruction + 1)
Q02CPU)	0.079 x (Number of steps for each instruction + 1)
Q02HCPU Q06HCPU Q12HCPU Q25HCPU Q02PHCPU Q06PHCPU Q12PHCPU Q25PHCPU Q12PRHCPU Q25PRHCPU	0.034 x (Number of steps for each instruction + 1)

A.2.1 Table of Processing Times

Following tables show the processing time for the instructions of Basic Model QCPU, High Performance Model QCPU/Process CPU/Redundant CPU.

NOTE *When using a file register (ZR), module access device (Un\G□, U3En\G0 to G511 (for Basic model QCPU) resp. U3En\G0 to G4095 (for High Performance Model QCPU/Process CPU/ Redundant CPU), and link direct device (Jn\□), add the processing time shown in tables A-12 (for Basic model QCPU) and A-14 (other CPU modules) to that of the instruction.*

Instruction	Processing (Device)				Processing time (μs)						
					Basic model			High Performance		Process	Redund.
					Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
LD, LDI, AND, ANI, OR, ORI			e.g. X0		0.20	0.16	0.10	0.079	0.034	0.034	0.034
			e.g. D0.0		0.30	0.24	0.15				
LDP, LDF, ANDP, ANDF, ORP, ORF					0.30	0.24	0.15	0.158	0.068	0.068	0.068
ANB, ORB, MPS, MRD, MPP					0.20	0.16	0.10	0.079	0.034	0.034	0.034
INV	not executed				0.20	0.16	0.10	0.079	0.034	0.034	0.034
	executed										
MEP MEF	not executed				0.30	0.24	0.15	0.173	0.073	0.073	0.073
	executed										
EGP	not changed				0.20	0.16	0.10	0.158	0.068	0.068	0.068
	changed (OFF/ON or ON/OFF)										
EGF	not changed				17	9.5	9.4				
	changed (OFF/ON or ON/OFF)				18	14	14				
OUT	excl. F, T and C	not changed			0.20	0.16	0.10	0.158	0.068	0.068	0.068
		changed (OFF/ON or ON/OFF)									
	D0.0	not changed			0.40	0.32	0.20				
		changed (OFF/ON or ON/OFF)									
	F	when OFF			24	20	19	2.8	1.2	1.2	1.2
		when ON	displayed		260	210	200	162	69.7	69.7	69.7
			display completed		205	165	155	126	54	54	54
	T	not executed			1.1	0.88	0.55	0.63	0.27	0.27	0.27
		executed	after time out								
			added	K							
				D	1.2	0.96	0.60				
	C	not executed			1.1	0.88	0.55	0.63	0.27	0.27	0.27
		executed	after time out								
			added	K							
				D	1.2	0.96	0.6				
	T	not executed			1.1	0.88	0.55	0.63	0.27	0.27	0.27
		executed	after time out								
			added	K							
				D	1.2	0.96	0.6				

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)			Processing time (µs)						
				Basic model			High Performance		Process	Redund.
				Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
SET	all devices exept F and D0.0	not executed		0.20	0.16	0.10	0.158	0.068	0.068	0.068
		executed	not changed							
			changed							
	D0.0	not executed		0.40	0.32	0.20				
		executed	not changed							
			changed							
	F	not executed		0.50	0.44	0.25	0.47	0.20	0.20	0.20
		executed	displayed	255	205	195	161	69	69	69
			display completed	195	160	150	0.47	0.20	0.20	0.20
RST	All devices exept the ones listed below	not executed		0.20	0.16	0.10	0.158	0.068	0.068	0.068
		executed	not changed							
			changed							
	D0.0	not executed		0.40	0.32	0.20	0.158	0.068	0.068	0.068
		executed	not changed							
			changed							
	SM	not executed		0.20	0.16	0.10	0.158	0.068	0.068	0.068
		executed								
	F	not executed		0.48	0.44	0.25	0.47	0.20	0.20	0.20
		executed	displayed	75	69	65	90	38	38	38
			display completed	43	35	33	0.47	0.20	0.20	0.20
	T, C	not executed		0.80	0.64	0.40	0.63	0.27	0.27	0.27
		executed		1.0	0.80	0.50				
	D	not executed		0.40	0.32	0.20	0.24	0.10	0.10	0.10
		executed		0.60	0.48	0.30				
	Z	not executed		0.50	0.40	0.25	0.47	0.20	0.20	0.20
		executed		9.4	7.9	7.4	4.3	1.9	1.9	1.9
	R	not executed		-	0.32	0.20	0.40	0.17	0.17	0.17
		executed		-	0.48	0.30				
PLS				12	9.5	9.2	1.0	0.44	0.44	0.44
PLF				11	9.5	8.9	1.0	0.44	0.44	0.44
FF	Y	not executed		0.68	0.40	0.25	0.47	0.20	0.20	0.20
		executed		7.5	6.2	5.7				
DELTA	DY0	not executed		0.50	0.40	0.25	0.47	0.20	0.20	0.20
		executed		26	21	21	5.9	2.6	2.6	2.6
DELTAP	DY0	not executed		0.48	0.40	0.25	0.47	0.20	0.20	0.20
		executed		58	45	43	5.9	2.6	2.6	2.6
SFT SFTP	not executed			0.50	0.34	0.25	0.47	0.20	0.20	0.20
	executed			12	8.7	8.3	1.66	0.71	0.71	0.71
MC	M0.0			0.40	0.32	0.20	0.24	0.10	0.10	0.10
	D0.0			3.3	2.9	2.8				
MCR				0.20	0.16	0.10	0.079	0.034	0.034	0.034
FEND END	error check executed			660	530	480	348	150	150	500
	without error check: - Battery check - Blown fuse check - Verification of I/O module			660	530	480	359	150	150	500
NOP				0.20	0.16	0.10	0.079	0.034	0.034	0.034
NOPLF PAGE				0.20	0.16	0.10	0.79	0.034	0.034	0.034
LD=	continuity			0.80	0.64	0.40	0.24	0.10	0.10	0.10
	no continuity									
AND=	not executed			0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity		0.80	0.64	0.40				
		no continuity								
OR=	not executed			0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity		0.80	0.64	0.40				
		no continuity								

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)		Processing time (µs)						
			Basic model			High Performance		Process	Redund.
			Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
LD<>	continuity		0.80	0.64	0.40	0.24	0.10	0.10	0.10
	no continuity								
AND<>	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
OR<>	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
LD>	continuity		0.80	0.64	0.40	0.24	0.10	0.10	0.10
	no continuity								
AND>	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
OR>	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
LD<=	continuity		0.80	0.64	0.40	0.24	0.10	0.10	0.10
	no continuity								
AND<=	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
OR<=	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
LD<	continuity		0.80	0.64	0.40	0.24	0.10	0.10	0.10
	no continuity								
AND<	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
OR<	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
LD>=	continuity		0.80	0.64	0.40	0.24	0.10	0.10	0.10
	no continuity								
AND>=	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
OR>=	not executed		0.70	0.56	0.35	0.24	0.10	0.10	0.10
	executed	continuity	0.80	0.64	0.40				
		no continuity							
LDD=	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
	no continuity					0.39	0.17	0.17	0.17
ANDD=	not executed		0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity	1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity				0.39	0.17	0.17	0.17
ORD=	not executed		0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity	1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity							
LDD<>	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
	no continuity								
ANDD<>	not executed		0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity	1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity							
ORD<>	not executed		0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity	1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity							
LDD>	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
	no continuity								

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)			Processing time (μs)						
				Basic model			High Performance		Process	Redund.
				Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
ANDD>	not executed			0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity								
ORD>	not executed			0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity								
LDD<=	continuity			1.0	0.80	0.50	0.55	0.24	0.24	0.24
	no continuity									
ANDD<=	not executed			0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity								
ORD<=	not executed			0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity								
LDD<	continuity			1.0	0.80	0.50	0.55	0.24	0.24	0.24
	no continuity									
ANDD<	not executed			0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity								
ORD<	not executed			0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity								
LDD>=	continuity			1.0	0.80	0.50	0.55	0.24	0.24	0.24
	no continuity									
ANDD>=	not executed			0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity								
ORD>=	not executed			0.80	0.64	0.40	0.39	0.17	0.17	0.17
	executed	continuity		1.0	0.80	0.50	0.55	0.24	0.24	0.24
		no continuity								
LDE= [1)]	single precision	continuity		—	—	—	93	40	6.4	6.4
							14.9	6.4		
		no continuity		—	—	—	92	40	6.4	6.4
							14.9	6.4		
	double precision	continuity		—	—	—	93	40	—	—
							14.9	6.4		
		no continuity		—	—	—	92	40	—	—
							14.9	6.4		
ANDE= [1)]	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	93	40	6.4	6.4
							14.9	6.4		
			no continuity	—	—	—	92	40	6.4	6.4
							14.9	6.4		
	double precision	not executed		—	—	—	—	—	—	—
		executed	continuity	—	—	—	93	40	—	—
							14.9	6.4		
			no continuity	—	—	—	92	40	—	—
							14.9	6.4		

[1] The Qn/QnH changes in processing time depending on the serial No. of the CPU module.
Top : The first 5 digits of the serial No. are "05031" or lower
Bottom: The first 5 digits of the serial No. are "05032" or higher
For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)			Processing time (μs)						
				Basic model			High Performance		Process	Redund.
				Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
ORE= [1)]	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	93 14.9	40 6.4	6.4	6.4
			no continuity	—	—	—	92 14.9	40 6.4	6.4	6.4
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	93 14.9	40 6.4	—	—
			no continuity	—	—	—	92 14.9	40 6.4	—	—
LDE<> [1)]	single precision	continuity		—	—	—	92 14.9	40 6.4	6.4	6.4
		no continuity		—	—	—	92 14.9	40 6.4	6.4	6.4
	double precision	continuity		—	—	—	92 14.9	40 6.4	—	—
		no continuity		—	—	—	92 14.9	40 6.4	—	—
ANDE<> [1)]	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	92 14.9	40 6.4	6.4	6.4
			no continuity	—	—	—	93 14.9	40 6.4	6.4	6.4
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	92 14.9	40 6.4	—	—
			no continuity	—	—	—	92 14.9	40 6.4	—	—
ORE<> [1)]	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	93 14.9	40 6.4	6.4	6.4
			no continuity	—	—	—	92 14.9	40 6.4	6.4	6.4
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	93 14.9	40 6.4	—	—
			no continuity	—	—	—	92 14.9	40 6.4	—	—
LDE> [1)]	single precision	continuity		—	—	—	92 14.9	40 6.4	6.4	6.4
		no continuity		—	—	—	92 14.9	40 6.4	6.4	6.4
	double precision	continuity		—	—	—	92 14.9	40 6.4	—	—
		no continuity		—	—	—	92	40	—	—
ANDE> [1)]	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	92 14.9	40 6.4	6.4	6.4
			no continuity	—	—	—	93 14.9	40 6.4	6.4	6.4
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	92 14.9	40 6.4	—	—
			no continuity	—	—	—	92 14.9	40 6.4	—	—

[1] The Qn/QnH changes in processing time depending on the serial No. of the CPU module.
Top : The first 5 digits of the serial No. are "05031" or lower
Bottom: The first 5 digits of the serial No. are "05032" or higher
For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)			Processing time (μs)						
				Basic model			High Performance		Process	Redund.
				Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
ORE> 1)	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	93 14.9	40 6.4	6.4	6.4
			no continuity	—	—	—	92 14.9	40 6.4	6.4	6.4
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	93 14.9	40 6.4	—	—
			no continuity	—	—	—	92 14.9	40 6.4	—	—
LDE<= 1)	single precision	continuity		—	—	—	93 14.9	40 6.4	6.4	6.4
		no continuity		—	—	—	92 14.9	40 6.4	6.4	6.4
	double precision	continuity		—	—	—	93 14.9	40 6.4	—	—
		no continuity		—	—	—	92 14.9	40 6.4	—	—
ANDE<= 1)	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	92 14.9	40 6.4	6.4	6.4
			no continuity	—	—	—	92 14.9	40 6.4	6.4	6.4
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	92 14.9	40 6.4	—	—
			no continuity	—	—	—	92 14.9	40 6.4	—	—
ORE<= 1)	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	92 14.9	40 6.4	6.4	6.4
			no continuity	—	—	—	92 14.9	40 6.4	6.4	6.4
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	92 14.9	40 6.4	—	—
			no continuity	—	—	—	92 14.9	40 6.4	—	—
LDE< 1)	single precision	continuity		—	—	—	92 14.9	40 6.4	6.4	6.4
		no continuity		—	—	—	92 14.9	40 6.4	6.4	6.4
	double precision	continuity		—	—	—	92 14.9	40 6.4	—	—
		no continuity		—	—	—	92 14.9	40 6.4	—	—

1 The Qn/QnH changes in processing time depending on the serial No. of the CPU module.
Top : The first 5 digits of the serial No. are "05031" or lower
Bottom: The first 5 digits of the serial No. are "05032" or higher
For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)			Processing time (μs)						
				Basic model			High Performance		Process	Redund.
				Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
ANDE< [1]	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	92	40	6.4	6.4
							14.9	6.4		
			no continuity	—	—	—	92	40	6.4	6.4
							14.9	6.4		
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	92	40	—	—
							14.9	6.4		
			no continuity	—	—	—	92	40	—	—
							14.9	6.4		
ORE< [1]	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	93	40	6.4	6.4
							14.9	6.4		
			no continuity	—	—	—	92	40	6.4	6.4
							14.9	6.4		
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	93	40	—	—
							14.9	6.4		
			no continuity	—	—	—	92	40	—	—
							14.9	6.4		
LDE>= [1]	single precision	continuity		—	—	—	93	40	6.4	6.4
							14.9	6.4		
		no continuity		—	—	—	92	40	6.4	6.4
							14.9	6.4		
	double precision	continuity		—	—	—	93	40	—	—
							14.9	6.4		
		no continuity		—	—	—	92	40	—	—
							14.9	6.4		
ANDE>= [1]	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	92	40	6.4	6.4
							14.9	6.4		
			no continuity	—	—	—	92	40	6.4	6.4
							14.9	6.4		
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	92	40	—	—
							14.9	6.4		
			no continuity	—	—	—	92	40	—	—
							14.9	6.4		
ORE>= [1]	single precision	not executed		—	—	—	0.55	0.24	0.24	0.24
		executed	continuity	—	—	—	92	40	6.4	6.4
							14.9	6.4		
			no continuity	—	—	—	92	40	6.4	6.4
							14.9	6.4		
	double precision	not executed		—	—	—	0.55	0.24	—	—
		executed	continuity	—	—	—	92	40	—	—
							14.9	6.4		
			no continuity	—	—	—	92	40	—	—
							14.9	6.4		
LD$=	continuity			—	—	—	38	16	16	16
	no continuity			—	—	—	34	15	15	15
AND$=	not executed			—	—	—	0.56	0.23	0.23	0.23
	executed	continuity		—	—	—	39	17	17	17
		no continuity		—	—	—	32	14	14	14

[1] The Qn/QnH changes in processing time depending on the serial No. of the CPU module.
Top : The first 5 digits of the serial No. are "05031" or lower
Bottom: The first 5 digits of the serial No. are "05032" or higher
For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)		Processing time (μs)						
			Basic model			High Performance		Process	Redund.
			Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
OR$=	not executed		—	—	—	0.56	0.24	0.24	0.24
	executed	continuity	—	—	—	40	17	17	17
		no continuity	—	—	—	33	14	14	14
LD$<>	continuity		—	—	—	32	14	14	14
	no continuity		—	—	—	40	17	17	17
AND$<>	not executed		—	—	—	0.56	0.23	0.23	0.23
	executed	continuity	—	—	—	33	14	14	14
		no continuity	—	—	—	39	17	17	17
OR$<>	not executed		—	—	—	0.56	0.24	0.24	0.24
	executed	continuity	—	—	—	32	14	14	14
		no continuity	—	—	—	39	17	17	17
LD$>	continuity		—	—	—	32	14	14	14
	no continuity		—	—	—	40	17	17	17
AND$>	not executed		—	—	—	0.56	0.23	0.23	0.23
	executed	continuity	—	—	—	33	14	14	14
		no continuity	—	—	—	39	17	17	17
OR$>	not executed		—	—	—	0.56	0.24	0.24	0.24
	executed	continuity	—	—	—	32	14	14	14
		no continuity	—	—	—	39	17	17	17
LD$<=	continuity		—	—	—	40	17	17	17
	no continuity		—	—	—	32	14	14	14
AND$<=	not executed		—	—	—	0.56	0.23	0.23	0.23
	executed	continuity	—	—	—	39	17	17	17
		no continuity	—	—	—	32	14	14	14
OR$<=	not executed		—	—	—	0.56	0.24	0.24	0.24
	executed	continuity	—	—	—	40	17	17	17
		no continuity	—	—	—	33	14	14	14
LD$<	continuity		—	—	—	32	14	14	14
	no continuity		—	—	—	40	17	17	17
AND$<	not executed		—	—	—	0.56	0.23	0.23	0.23
	executed	continuity	—	—	—	32	14	14	14
		no continuity	—	—	—	39	16	16	16
OR$<	not executed		—	—	—	0.56	0.24	0.24	0.24
	executed	continuity	—	—	—	32	14	14	14
		no continuity	—	—	—	39	16	16	16
LD$>=	continuity		—	—	—	40	17	17	17
	no continuity		—	—	—	32	14	14	14
AND$>=	not executed		—	—	—	0.56	0.23	0.23	0.23
	executed	continuity	—	—	—	39	16	16	16
		no continuity	—	—	—	32	14	14	14
OR$>=	not executed		—	—	—	0.56	0.24	0.24	0.24
	executed	continuity	—	—	—	39	17	17	17
		no continuity	—	—	—	32	14	14	14
BKCMP= BKCMP=P	n = 1		130	105	97	48	21	21	21
	n = 96		205	175	165	142	61	61	61
BKCMP<> BKCMP<>P	n = 1		130	105	98	48	21	21	21
	n = 96		210	180	165	150	65	65	65
BKCMP> BKCMP>P	n = 1		130	105	97	48	21	21	21
	n = 96		210	180	165	142	61	61	61
BKCMP>= BKCMP>=P	n = 1		130	105	98	48	21	21	21
	n = 96		205	175	165	150	65	65	65
BKCMP< BKCMP<P	n = 1		130	105	98	48	21	21	21
	n = 96		210	180	165	158	68	68	68
BKCMP<= BKCMP<=P	n = 1		130	105	97	48	21	21	21
	n = 96		205	175	165	150	65	65	65

[1] The Qn/QnH changes in processing time depending on the serial No. of the CPU module.
Top : The first 5 digits of the serial No. are "05031" or lower
Bottom: The first 5 digits of the serial No. are "05032" or higher
For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)		Processing time (μs)						
			Basic model			High Performance		Process	Redund.
			Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
+ (s, d) +P (s, d)	executed		1.0	0.80	0.50	0.39	0.17	0.17	0.17
+ (s1, s2, d) +P (s1, s2, d)	executed		1.2	0.96	0.60	0.47	0.20	0.20	0.20
- (s, d) -P (s, d)	executed		1.0	0.80	0.50	0.39	0.17	0.17	0.17
- (s1, s2, d) -P (s1, s2, d)	executed		1.2	0.96	0.60	0.47	0.20	0.20	0.20
D+ (s, d) D+P (s, d)	executed		1.3	1.04	0.65	0.71	0.31	0.31	0.31
D+ (s1, s2, d) D+P (s1, s2, d)	executed		1.5	1.2	0.75	0.79	0.34	0.34	0.34
D- (s, d) D-P (s, d)	executed		1.3	1.04	0.65	0.71	0.30	0.30	0.30
D- (s1, s2, d) D-P (s1, s2, d)	executed		1.5	1.2	0.75	0.79	0.34	0.34	0.34
x (s1,s2,d) xP (s1,s2,d)	executed		1.1	0.88	0.55	0.47	0.20	0.20	0.20
/ (s1,s2,d) /P (s1,s2,d)			19	16	15	2.7	1.2	1.2	1.2
Dx (s1,s2,d) DxP (s1,s2,d)			41	34	31	7.9	3.4	3.4	3.4
D/ (s1,s2,d) D/P (s1,s2,d)			28	23	21	14	6.1	6.1	6.1
B+ (s, d) B+P (s, d)			34	28	26	2.2	1.0	1.0	1.0
B+ (s1, s2, d) B+P (s1, s2, d)			47	39	37	5.0	2.2	2.2	2.2
B- (s, d) B-P (s, d)			34	28	26	2.0	0.9	0.9	0.9
B- (s1,s2,d) B-P (s1,s2,d)			48	40	38	4.9	2.1	2.1	2.1
DB+ (s, d) DB+P (s, d)			58	48	44	12	5.0	5.0	5.0
DB+ (s1,s2,d) DB+P (s1,s2,d)			60	49	46	12	5.3	5.3	5.3
DB- (s, d) DB-P (s, d)			59	48	45	11	4.8	4.8	4.8
DB- (s1,s2,d) DB-P (s1,s2,d)			60	51	45	12	5.2	5.2	5.2
Bx (s1, s2, d) BxP (s1, s2, d)			42	35	33	3.7	1.6	1.6	1.6
B/ (s1, s2, d) B/P (s1, s2, d)			48	40	37	3.8	1.6	1.6	1.6
DBx (s1, s2, d) DBxP (s1, s2, d)			140	120	110	24	10	10	10
DB/ (s1, s2, d) DB/P (s1,s2,d)			83	69	65	27	12	12	12
E+ (s, d) E+P (s, d)	single precision	s = 0, d = 0 s = 2^{127}, d = 2^{127}	—	—	—	1.8	0.78	0.78	0.78
	double precision	s = 0, d = 0 s = 2^{127}, d = 2^{127}	—	—	—	203	87	—	—
E+ (s1, s2, d) E+P (s1, s2, d)	single precision	s1 = 0, s2 = 0 s1 = 2^{127}, s2 = 2^{127}	—	—	—	2.4	1.1	1.1	1.1
	double precision	s = 0, d = 0 s = 2^{127}, d = 2^{127}	—	—	—	209	90	—	—

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)		Processing time (μs)						
			Basic model			High Performance		Process	Redund.
			Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
E- (s, d) E-P (s, d)	single precision	s = 0, d = 0 s = 2^{127}, d = 2^{127}	—	—	—	1.8	0.78	0.78	0.78
	double precision	s = 0, d = 0 s = 2^{127}, d = 2^{127}	—	—	—	202	87	—	—
E- (s1, s2, d) E-P (s1, s2, d)	single precision	s1 = 0, s2 = 0 s1 = 2^{127}, s2 = 2^{127}	—	—	—	2.4	1.1	1.1	1.1
	double precision	s = 0, d = 0 s = 2^{127}, d = 2^{127}	—	—	—	210	90	—	—
Ex (s1, s2, d) ExP (s1, s2, d)	single precision	s1 = 0, s2= 0 s1 = 2^{127}, s2 = 2^{127}	—	—	—	2.4	1.1	1.1	1.1
	double precision	s = 0, d = 0 s = 2^{127}, d = 2^{127}	—	—	—	222	96	—	—
E/ (s1, s2, d) E/P (s1, s2, d)	single precision	s1 = 0, s2 = 1 s1 = 2^{127}, s2 = -2^{126}	—	—	—	12	5.2	5.2	5.2
	double precision	s = 0, d = 0 s1 = 2^{127}, s2 = -2^{126}	—	—	—	369	159	—	—
$+ (s, d) $+P (s, d)			—	—	—	68	29	29	29
$+ (s1, s2, d) $+P (s1, s2, d)			—	—	—	81	35	35	35
INC INCP			0.70	0.56	0.35	0.32	0.14	0.14	0.14
DINC DINCP			0.90	0.72	0.45	0.47	0.20	0.20	0.20
DEC DECP			0.70	0.56	0.35	0.32	0.14	0.14	0.14
DDEC DDECP			0.90	0.72	0.45	0.47	0.20	0.20	0.20
BCD BCDP			20	16	15	1.1	0.48	0.48	0.48
DBCD DBCDP			26	21	20	3.2	1.4	1.4	1.4
BIN BINP			19	16	15	1.0	0.44	0.44	0.44
DBIN DBINP			22	18	17	1.9	0.82	0.82	0.82
INT INTP	single precision	s = 0 s = 32766.5	—	—	—	3.2	1.4	1.4	1.4
	double precision	s = 0 s = 32766.5	—	—	—	22	9.3	—	—
DINT DINTP	single precision	s = 0 s = 1234567890.3	—	—	—	2.5	1.1	1.1	1.1
	double precision	s = 0 s = 1234567890.3	—	—	—	24	10	—	—
FLT FLTP	single precision	s = 0 s = 7FFFH	—	—	—	2.1	0.92	0.92	0.92
	double precision	s = 0 s = 7FFFH	—	—	—	22	9.6	—	—
DFLT DFLTP	single precision	s = 0 s = 7FFFFFFFH	—	—	—	2.1	0.88	0.88	0.88
	double precision	s = 0 s = 7FFFFFFFH	—	—	—	26	11	—	—
DBL DBLP			19	16	15	4.5	1.9	1.9	1.9
WORD WORDP			23	19	17	4.7	2.0	2.0	2.0

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)	Processing time (µs)						
		Basic model			High Performance		Process	Redund.
		Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
GRY GRYP		19	16	15	4.7	2.0	2.0	2.0
DGRY DGRYP		23	19	17	5.3	2.3	2.3	2.3
GBIN GBINP		52	42	40	18	7.7	7.7	7.7
DGBIN DGBINP		110	88	84	32	14	14	14
NEG NEGP		16	13	12	3.6	1.6	1.6	1.6
DNEG DNEGP		19	17	15	4.3	1.8	1.8	1.8
ENEG ENEGP		—	—	—	3.9	1.7	1.7	1.7
BKBCD (s, d, n) BKBCDP (s, d, n)	n = 1	78	63	57	38	17	17	17
	n = 96	315	275	250	99	43	43	43
BKBIN (s, d, n) BKBINP (s, d, n)	n = 1	74	61	57	38	17	17	17
	n = 96	285	255	230	99	43	43	43
MOV MOVP	s = D0, d = D1	0.70	0.56	0.35	0.24	0.10	0.10	0.10
	s = D0, d = J1/W1	155	130	120	140 [2)]	60 [2)]	60 [2)]	60 [2)]
DMOV DMOVP	s = D0, d = D1	0.90	0.72	0.45	0.47	0.20	0.20	0.20
	s = D0, d = J1/W1	165	135	120	147 [2)]	64 [2)]	64 [2)]	64 [2)]
EMOV EMOVP		—	—	—	0.63	0.27	0.27	0.27
$MOV $MOVP	0 characters (basic model)	46	38	35	40	17	17	17
	32 characters (basic model)	98	80	73				
CML CMLP		0.70	0.56	0.35	0.40	0.17	0.17	0.17
DCML DCMLP		0.90	0.72	0.45	0.55	0.24	0.24	0.24
BMOV (s, d, n) BMOVP (s, d, n)	n = 1	27	21	20	17	7.1	7.1	7.1
	n = 96	72	62	53	32	14	14	14
FMOV (s, d, n) FMOVP (s, d, n)	n = 1	23	19	17	6.7	2.9	2.9	2.9
	n = 96	48	41	36	14	6.1	6.1	6.1
XCH XCHP		7.6	6.3	5.7	1.3	0.54	0.54	0.54
DXCH DXCHP		9.5	8.0	7.1	1.3	0.54	0.54	0.54
BXCH (d1, d2, n) BXCHP (d1, d2, n)	n = 1	62	51	48	31	13	13	13
	n = 96	165	140	125	84	36	36	36
SWAP SWAPP		17	14	13	3.7	1.6	1.6	1.6
CJ		10	8.5	8.1	3.2	1.4	1.4	1.4
SCJ		10	8.5	8.1	3.2	1.4	1.4	1.4
JMP		11	8.5	8.1	3.2	1.4	1.4	1.4
GOEND		3.3	2.9	2.8	0.39	0.34	0.34	0.34
EI		14	11	11	1.3	0.54	0.54	0.54
DI		13	12	11	0.95	0.41	0.41	0.41
IMASK		41	34	35	11	4.6	4.6	4.6
IRET		205	170	155	1.6	0.68	0.68	0.68
RFS RFSP	s= X, n = 1	55	46	43	6.7	4.7	4.7	4.7
	s = Y, n = 1	54	45	41				
	s = X, n = 96	79	64	59	19	13	13	13
	s = Y, n =96	73	61	56				

[2] Processing time when Q312B is used.

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)	Processing time (μs)						
		Basic model			High Performance		Process	Redund.
		Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
UDCNT1		—	—	—	15	6.5	6.5	—
UDCNT2		—	—	—	16	6.8	6.8	—
TTMR		—	—	—	10	4.4	4.4	—
STMR		—	—	—	20	7.1	7.1	—
ROTC		—	—	—	26	11	11	—
RAMP		—	—	—	18	7.7	7.7	—
SPD		—	—	—	19	8.3	8.3	—
PLSY		—	—	—	10	4.5	4.5	—
PWM		—	—	—	9.1	3.9	3.9	—
MTR		—	—	—	11	4.9	4.9	—
WAND (s, d) WANDP (s, d)	executed	1.0	0.80	0.50	0.39	0.17	0.17	0.17
WAND (s1, s2, d) WANDP (s1, s2, d)	executed	1.2	0.96	0.60	0.47	0.20	0.20	0.20
DAND (s, d) DANDP (s, d)	executed	1.3	1.04	0.65	0.71	0.31	0.31	0.31
DAND (s1, s2, d) DANDP (s1, s2, d)	executed	1.5	1.2	0.75	0.79	0.34	0.34	0.34
BKAND (s1, s2, d, n) BKANDP (s1, s2, d, n)	n = 1	110	87	79	36	16	16	16
	n = 96	185	155	140	74	32	32	32
WOR (s, d) WOR (s, d)	executed	1.0	0.80	0.50	0.40	0.17	0.17	0.17
WOR (s1, s2, d) WORP (s1, s2, d)	executed	1.2	0.96	0.60	0.47	0.20	0.20	0.20
DOR (s, d) DORP (s, d)	executed	1.3	1.04	0.65	0.71	0.31	0.31	0.31
DOR (s1, s2, d) DORP (s1, s2, d)	executed	1.5	1.2	0.75	0.79	0.34	0.34	0.34
BKOR (s1, s2, d, n)	n = 1	110	87	81	36	16	16	16
BKORP (s1, s2, d, n)	n = 96	185	155	140	74	32	32	32
WXOR (s, d) WXORP (s, d)	executed	1.0	0.80	0.50	0.39	0.17	0.17	0.17
WXOR (s1, s2, d) WXORP (s1, s2, d)	executed	1.2	0.96	0.60	0.47	0.20	0.20	0.20
DXOR (s, d) DXORP (s, d)	executed	1.3	1.04	0.65	0.71	0.31	0.31	0.31
DXOR (s1, s2, d) DXORP (s1, s2, d)	executed	1.5	1.2	0.75	0.79	0.34	0.34	0.34
BKXOR (s1, s2, d, n) BKXORP (s1, s2, d, n)	n = 1	110	87	81	36	16	16	16
	n = 96	183	155	140	74	32	32	32
WXNR (s, d) WXNRP (s, d)	executed	1.0	0.80	0.50	0.40	0.17	0.17	0.17
WXNR (s1, s2, d) WXNRP (s1, s2, d)	executed	1.2	0.96	0.60	0.47	0.20	0.20	0.20
DXNR (s, d) DXNRP (s, d)	executed	1.3	1.04	0.65	0.71	0.31	0.31	0.31
DXNR (s1,s2,d) DXNRP (s1,s2,d)	executed	1.5	1.2	0.75	0.79	0.34	0.34	0.34
BKXNR (s1, s2, d, n) BKXNR (s1, s2, d, n)	n = 1	110	87	82	36	16	16	16
	n = 96	185	155	140	74	32	32	32
ROR (d, n) RORP (d, n)	n = 1	13	11	9.7	2.0	0.85	0.85	0.85
	n = 15	13	11	9.7	2.0	0.85	0.85	0.85

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)			Processing time (μs)						
				Basic model			High Performance		Process	Redund.
				Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
RCR (d, n) RCRP (d, n)	n = 1			15	12	12	1.6	0.68	0.68	0.68
	n = 15			15	13	12	1.6	0.68	0.68	0.68
ROL (d, n) ROLP (d, n)	n = 1			13	11	10	2.0	0.85	0.85	0.85
	n = 15			13	11	10	2.0	0.85	0.85	0.85
RCL (d, n) RCLP (d, n)	n = 1			15	13	12	1.6	0.68	0.68	0.68
	n = 15			16	13	12	1.6	0.68	0.68	0.68
DROR (d, n) DRORP (d, n)	n = 1			15	12	12	3.9	1.7	1.7	1.7
	n = 31			15	13	12	4.0	1.7	1.7	1.7
DRCR (d, n) DRCRP (d, n)	n = 1			17	14	14	4.3	1.8	1.8	1.8
	n = 31			18	16	15	4.3	1.9	1.9	1.9
DROL (d, n) DROLP (d, n)	n = 1			14	13	12	3.9	1.7	1.7	1.7
	n = 31			14	13	12	4.0	1.7	1.7	1.7
DRCL (d, n) DRCLP (d, n)	n = 1			18	15	14	4.3	1.8	1.8	1.8
	n = 31			20	17	16	4.3	1.9	1.9	1.9
SFR (d, n) SFRP (d, n)	n = 1			13	10	9.7	1.7	0.75	0.75	0.75
	n = 15			13	11	9.5	2.0	0.85	0.85	0.85
SFL (d, n) SFLP (d, n)	n = 1			12	10	9.5	1.7	0.75	0.75	0.75
	n = 15			12	9.8	9.5	2.0	0.85	0.85	0.85
BSFR (d, n) BSFRP (d, n)	n = 1			42	35	33	20	8.6	8.6	8.6
	n = 96			69	58	54	24	10	10	10
BSFL (d, n) BSFLP (d, n)	n = 1			41	34	32	20	8.5	8.5	8.5
	n = 96			63	53	50	23	10	10	10
DSFR (d, n) DSFRP (d, n)	n = 1			19	16	15	1.3	0.58	0.58	0.58
	n = 96			71	61	53	25	11	11	11
DSFL (d, n) DSFLP (d, n)	n = 1			19	16	15	1.3	0.58	0.58	0.58
	n = 96			70	60	52	26	11	11	11
BSET (d, n) BSETP (d, n)	n = 1			27	22	20	7.6	3.3	3.3	3.3
	n = 15			27	22	20	7.6	3.3	3.3	3.3
BRST (d, n) BRSTP (d, n)	n = 1			27	22	21	7.6	3.3	3.3	3.3
	n = 15			27	22	21	7.6	3.3	3.3	3.3
TEST (s1, s2, d) TESTP (s1, s2, d)				35	30	27	8.2	3.5	3.5	3.5
DTEST (s1, s2, d) DTESTP (s1, s2, d)				37	31	28	9.2	3.9	3.9	3.9
BKRST (s, n) BKRST (s, n)	n = 1			49	41	38	18	7.8	7.8	7.8
	n = 96			64	54	50	19	8.2	8.2	8.2
SER (s1, s2, d, n) SERP (s1, s2, d, n)	n = 1	match		56	54	42	22	9.6	9.6	9.6
		no match		56	54	42	21	8.9	8.9	8.9
	n = 96	match		280	240	220	115	49	49	49
		no match		280	240	220	133	57	57	57
DSER (s1, s2, d, n) DSERP (s1, s2, d, n)	n = 1	match		71	67	53	23	9.9	9.9	9.9
		no match		71	67	54	23	9.7	9.7	9.7
	n = 96	match		495	415	375	142	61	61	61
		no match		500	415	375	132	57	57	57
SUM SUMP	s = 0			32	26	25	3.9	1.7	1.7	1.7
	s = FFFF$_H$			27	22	21				
DSUM DSUMP	s = 0			54	44	42	4.7	2.0	2.0	2.0
	s = FFFFFFFF$_H$			54	44	42	12	5.0	5.0	5.0
DECO (s, d, n) DECOP (s, d, n)	n = 2			60	50	46	20	8.6	8.6	8.6
	n = 8			80	65	61	27	12	12	12
ENCO (s, d, n) ENCOP (s, d, n)	n = 2		M1 = ON	66	55	51	21	9.1	9.1	9.1
			M4 = ON	66	54	51	21	9.1	9.1	9.1
	n = 8		M1 = ON	90	76	71	28	12	12	12
			M256 = ON	76	74	71	26	11	11	11
SEG SEGP				8.0	6.8	6.1	1.3	0.54	0.54	0.54

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)	Processing time (μs)						
		Basic model			High Performance		Process	Redund.
		Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
DIS (s, d, n) DISP (s, d, n)	n = 1	47	39	36	18	7.7	7.7	7.7
	n = 4	53	43	40	19	8.3	8.3	8.3
UNI (s, d, n) UNIP (s, d, n)	n = 1	54	44	41	21	8.9	8.9	8.9
	n = 4	60	49	46	23	9.7	9.7	9.7
NDIS (s1, d, s2) NDISP (s1, d, s2)		92	76	38	41	18	18	18
NUNI (s1, d, s2) NUNIP (s1, d, s2)		47	39	36	42	18	18	18
WTOB (s, d, n) WTOBP (s, d, n)	n = 1	56	46	42	47	20	20	20
	n = 96	190	155	145	99	43	43	43
BTOW (s, d, n) BTOWP (s, d, n)	n = 1	56	46	42	45	19	19	19
	n = 96	190	155	145	89	38	38	38
MAX (s, d, n) MAXP (s, d, n)	n = 1	48	40	36	17	7.1	7.1	7.1
	n = 96	300	240	235	136	59	59	59
MIN (s, d, n) MINP (s, d, n)	n = 1	48	40	36	17	7.1	7.1	7.1
	n = 96	300	240	235	159	69	69	69
DMAX (s, d, n) DMAXP (s, d, n)	n = 1	52	43	39	27	12	12	12
	n = 96	600	490	460	181	78	78	78
DMIN (s, d, n) DMINP (s, d, n)	n = 1	52	43	39	27	12	12	12
	n = 96	585	475	445	112	48	48	48
SORT (s1, n, s2, d1, d2)	n = 1	66	55	50	16	7.1	7.1	7.1
	n = 96	105	86	80	14	6.2	6.2	6.2
DSORT (s1, n, s2, d1, d2)	n = 1	98	57	52	17	7.1	7.1	7.1
	n = 96	115	96	88	16	6.8	6.8	6.8
WSUM (s, d, n) WSUMP (s, d, n)	n = 1	52	43	40	16.4	7.1	7.1	7.1
	n = 96	175	140	135	68.4	29.5	29.5	29.5
DWSUM (s, d, n) DWSUMP (s, d, n)	n = 1	61	51	46	18.9	8.2	8.2	8.2
	n = 96	515	420	395	130.4	56.1	56.1	56.1
FOR	n = 0	11	8.9	8.1	2.3	1.0	1.0	1.0
NEXT		8.8	7.3	6.8	3.3	1.4	1.4	1.4
BREAK BREAKP		37	30	28	11	4.6	4.6	4.6
CALL (pn) CALLP (pn)	internal file pointer	17	14	13	2.1	0.88	0.88	0.88
	common file pointer				33	14	14	14
CALL (pn s1 to s5) CALLP (pn s1 to s5)		245	200	190	135	58	58	58
RET	Return to origin program	16	13	12	2.9	1.3	1.3	1.3
	Return to other program	——	—	—	20	8.5	8.5	8.5
FCALL (pn) FCALLP (pn)	internal file pointer	29	24	22	3.6	1.6	1.6	1.6
	Common file pointer				20	8.7	8.7	8.7
FCALL (pn s1 to s5) FCALLP (pn s1 to s5)		250	205	190	134	57	57	57
ECALL (pn) ECALL P (pn)		—	—	—	77	33	33	33
ECALL (pn s1 to s5) ECALLP (pn s1 to s5)		—	—	—	162	70	70	70
EFCALL (pn) EFCALLP (pn)		—	—	—	78	34	34	34
EFCALL (pn s1 to s5) EFCALLP (pn s to s5)		—	—	—	200	86	86	86
COM		110	77	72	55	16	16	16
IX		65	54	51	12	5.2	5.2	5.2

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)		Processing time (µs)						
			Basic model			High Performance		Process	Redund.
			Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
IXEND			30	26	25	4.7	2.0	2.0	2.0
IXDEV	number of contacts: 1		145	120	110	48	21	21	21
	number of contacts: 14		770	630	585	93	40	40	40
IXSET	number of contacts: 1		145	120	110	48	21	21	21
	number of contacts: 14		770	630	585	93	40	40	40
FIFW FIFWP	number of data points: 1		36	32	28	11	4.5	4.5	4.5
	number of data points: 96								
FIFR FIFRP	number of data points: 1		45	41	36	13	5.6	5.6	5.6
	number of data points: 96		93	82	70	32	14	14	14
FPOP FPOPP	number of data points: 1		40	37	32	16	7.0	7.0	7.0
	number of data points: 96								
FINS FINSP	number of data points: 1		53	44	38	20	8.4	8.4	8.4
	number of data points: 96		100	89	76	36	15	15	15
FDEL FDELP	number of data points: 1		60	50	43	19	7.5	7.5	7.5
	number of data points: 96		110	95	82	39	15	15	15
FROM (n1, n2, d, n3) FROMP (n1, n2, d, n3 [3]))	n3 = 1		125	105	93	47 [4]	22 [4]	22 [4]	22 [4]
	n3 = 1000		740	695	685	47 6 [4]	437 [4]	437 [4]	437 [4]
DFRO (n1, n2, d, n3) DFROP (n1, n2, d, n3 [3])	n3 = 1		130	110	100	51 [4]	24 [4]	24 [4]	24 [4]
	n3 = 500		745	695	675	478 [4]	437 [4]	437 [4]	437 [4]
TO (n1, n2, d, n3) TOP (n1, n2, d, n3 [3])	n3 = 1		120	105	92	48 [4]	20 [4]	20 [4]	20 [4]
	n3 = 1000		735	680	645	479 [4]	412 [4]	412 [4]	412 [4]
DTO (n1, n2, d, n3) DTOP (n1, n2, d, n3 [3])	n3 = 1		130	110	99	50 [4]	23 [4]	23 [4]	23 [4]
	n3 = 500		740	680	640	457 [4]	416 [4]	416 [4]	416 [4]
PR	SM701 ON	1 character	—	—	—	33	11	11	—
		32 character				48	18	18	
	SM701 OFF					21	7.8	7.8	
PRC			—	—	—	181	16	16	—
LED	displayed		—	—	—	—	—	—	—
	display completed								
LEDC	displayed		—	—	—	—	—	—	—
	display completed								
LEDR	no display → no display		—	—	—	0.40	0.17	0.17	0.17
	Execute LED instruction → no display					103	44	44	44
CHKST			—	—	—	5.8	2.5	2.5	2.5
CHK (error check)	1 input contact	no error at contact 1	—	—	—	24	10	10	10
	150 input contacts	no error at contact 150				1676	721	721	721
		no error at contact 1				88	38	38	38
CHKCIR	10 failure check circuits		—	—	—	5.8	2.5	2.5	2.5
SLT	all internal devices		—	—	—	—	—	—	—
	file registers 8 kByte								
	completion of SLT instruction								
SLTR			—	—	—	—	—	—	—
STRA	Start		—	—	—	—	—	—	—
	completion of STRA instruction								
STRAR			—	—	—	—	—	—	—
PTRA			—	—	—	—	—	—	—
PTRAR			—	—	—	—	—	—	—

[3] The FROM/TO instruction differs in processing time according to the number of slots and the loaded modules. (The CPU also differs in processing time according to the extension base type.)

[4] Processing times when the Q312B is used to execute the instruction for the QJ71C24 in slot 0.

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)		Processing time (μs)						
			Basic model			High Performance		Process	Redund.
			Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
PTRAEXE PTRAEXEP	instruction execution		—	—	—	—	—	—	—
	program trace								
BINDA BINDAP	s = 1		—	—	—	15	6.7	6.7	6.7
	s = -32768					24	10	10	10
DBINDA DBINDAP	s = 1		—	—	—	43	18	18	18
	s = -2147483648					86	37	37	37
BINHA BINHAP	s = 1		—	—	—	18	7.7	7.7	7.7
	s = FFFFH					19	8.2	8.2	8.2
DBINHA DBINHAP	s = 1		—	—	—	23	10	10	10
	s = FFFFFFFFH					24	10	10	10
BCDDA BCDDAP	s = 1		—	—	—	23	9.8	9.8	9.8
	s = 9999					21	8.9	8.9	8.9
DBCDDA DBCDDAP	s = 1		—	—	—	22	9.5	9.5	9.5
	s = 99999999					29	13	13	13
DABIN DABINP	s = 1		—	—	—	57	25	25	25
	s = -32768					58	25	25	25
DDABIN DDABINP	s = 1		—	—	—	92	40	40	40
	s = -2147483648					106	46	46	46
HABIN HABINP	s = 1		—	—	—	13	5.8	5.8	5.8
	s = FFFFH					15	6.4	6.4	6.4
DHABIN DHABINP	s = 1		—	—	—	22	9.5	9.5	9.5
	s = FFFFFFFFH					25	11	11	11
DABCD DABCDP	s = 1		—	—	—	16	6.9	6.9	6.9
	s = 9999					17	7.2	7.2	7.2
DDABCD DDABCDP	s = 1		—	—	—	25	11	11	11
	s = 99999999					29	13	13	13
COMRD COMRDP			—	—	—	40	17	17	17
LEN LENP	1 character		—	—	—	18	8.0	8.0	8.0
	96 characters					86	37	37	37
STR STRP			—	—	—	53	23	23	23
DSTR DSTRP			—	—	—	123	53	53	53
VAL VALP			—	—	—	95	41	41	41
DVAL DVALP			—	—	—	166	72	72	72
ESTR ESTRP			—	—	—	564	243	243	243
EVAL EVALP	floating-point format		—	—	—	100	43	43	43
	exponential format					127	55	55	55
ASC (s, d, n) ASCP (s, d, n)	n = 1		—	—	—	64	28	28	28
	n = 96					289	125	125	125
HEX (s, d, n) HEXP (s, d, n)	n = 1		—	—	—	60	26	26	26
	n = 96					343	148	148	148
RIGHT (s, d, n) RIGHTP (s, d, n)	n = 1		—	—	—	49	21	21	21
	n = 96					131	56	56	56
LEFT (s, d, n) LEFTP (s, d, n)	n = 1		—	—	—	50	21	21	21
	n = 96		—	—	—	131	56	56	56
MIDR MIDRP			—	—	—	53	23	23	23
MIDW MIDWP			—	—	—	128	55	55	55
INSTR INSTRP	no match		—	—	—	58	25	25	25
	match	first				55	24	24	24
		finals				58	25	25	25

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)		Processing time (μs)						
			Basic model			High Performance		Process	Redund.
			Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
EMOD EMODP			—	—	—	527	227	227	227
EREXP EREXPP			—	—	—	1656	713	713	713
SIN SINP	single precision		—	—	—	115	50	50	50
	double precision					1945	837	—	—
COS COSP	single precision		—	—	—	122	53	53	53
	double precision					2618	1127	—	—
TAN TANP	single precision		—	—	—	123	53	53	53
	double precision					2618	1127	—	—
ASIN ASINP	single precision		—	—	—	111	48	48	48
	double precision					2491	1072	—	—
ACOS ACOSP	single precision		—	—	—	115	49	49	49
	double precision					2367	1019	—	—
ATAN ATANP	single precision		—	—	—	157	68	68	68
	double precision					3140	1352	—	—
RAD RADP	single precision		—	—	—	17	7.2	7.2	7.2
	double precision					24	10	—	—
DEG DEGP	single precision		—	—	—	17	7.2	7.2	7.2
	double precision					23	9.9	—	—
SQR SQRP	single precision		—	—	—	28	12	12	12
	double precision					1812	780	—	—
EXP EXPP	single precision	s = -10	—	—	—	129	56	56	56
		s = 1							
	double precision	s = -10				2386	1026	—	—
		s = 1							
LOG LOGP	single precision	s = 1	—	—	—	113	49	49	49
		s = 10							
	double precision	s = 1				2146	924	—	—
		s = 10							
RND RNDP			—	—	—	3.9	1.7	1.7	1.7
SRND SRNDP			—	—	—	3.5	1.5	1.5	1.5
BSQR BSQRP	s = 0		—	—	—	6.2	2.7	2.7	2.7
	s = 9999					38	16	16	16
BDSQR BDSQRP	s = 0		—	—	—	6.2	2.7	2.7	2.7
	s = 99999999					38	16	16	16
BSIN BSINP			—	—	—	12	5.1	5.1	5.1
BCOS BCOSP			—	—	—	12	5.2	5.2	5.2
BTAN BTANP			—	—	—	12	5.2	5.2	5.2
BASIN BASINP			—	—	—	20	8.7	8.7	8.7
BACOS BACOSP			—	—	—	21	9.0	9.0	9.0
BATAN BATANP			—	—	—	22	9.6	9.6	9.6
LIMIT LIMITP			34	28	26	10	4.3	4.3	4.3
DLIMIT DLIMITP			41	34	30	11	4.7	4.7	4.7
BAND BANDP			33	28	25	9.8	4.2	4.2	4.2
DBAND DBANDP			40	34	30	11	4.9	4.9	4.9

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)	Processing time (μs)						
		Basic model			High Performance		Process	Redund.
		Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
ZONE ZONEP		31	25	24	9.1	3.9	3.9	3.9
DZONE DZONEP		37	29	28	11	4.6	4.6	4.6
RSET RSETP		—	18	16	6.8	2.9	2.9	2.9
QDRSET QDRSETP		—	—	—	205	88	88	88
QCDSET QCDSETP		—	—	—	147	63	63	63
DATERD DATERDP		30	25	23	13	5.5	5.5	5.5
DATEWR DATEWRP		69	57	54	15	6.4	6.4	6.4
DATE+ DATE+P	no digit increase	47	39	36	13	5.4	5.4	5.4
	digit increase	50	42	38	13	5.4	5.4	5.4
DATE- DATE-P	no digit increase	47	40	36	12	5.2	5.2	5.2
	digit increase	50	42	38	12	5.2	5.2	5.2
SECOND SECONDP		28	24	22	10	4.5	4.5	4.5
HOUR HOURP		38	32	29	12	5.2	5.2	5.2
MSG	1 character	—	—	—	3.0	1.3	1.3	1.3
	32 characters							
PKEY	initial time	—	—	—	20	8.6	8.6	8.6
	no acceptance				19	8.2	8.2	8.2
PSTOP PSTOPP		—	—	—	79	34	34	34
POFF POFFP		—	—	—	79	34	34	34
PSCAN PSCANP		—	—	—	75	32	32	32
PLOW PLOWP		—	—	—	80	34	34	34
WDT WDTP		18	15	14	5.9	2.6	2.6	2.6
DUTY		41	36	32	9.3	4.0	4.0	4.0
ZRRDB ZRRDBP		—	24	22	7.9	3.4	3.4	3.4
ZRWRB ZRWRBP		—	27	24	9.4	4.0	4.0	4.0
ADRSET ADRSETP		23	19	18	4.9	2.1	2.1	2.1
KEY		—	—	—	17	7.3	7.3	—
ZPUSH ZPUSHP		38	33	30	11	4.7	4.7	4.7
ZPOP ZPOPP		37	31	29	5.1	2.2	2.2	2.2
EPROMWR EPROMWRP		—	—	—	—	—	—	—
ZCOM		105	82	80	691	289	289	289
READ		—	—	—	554	—	—	—
SREAD		—	—	—	588	—	—	—
WRITE		—	—	—	582	—	—	—
SWRITE		—	—	—	625	—	—	—
SEND		—	—	—	—	—	—	—
RECV		—	—	—	—	—	—	—
REQ		—	—	—	—	—	—	—
ZNFR		—	—	—	—	—	—	—

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

Instruction	Processing (Device)		Processing time (μs)						
			Basic model			High Performance		Process	Redund.
			Q00J	Q00	Q01	Qn	QnH	QnPH	QnPRH
ZNTO			—	—	—	—	—	—	—
ZNRD			—	—	—	—	—	—	—
			—	—	—	—	—	—	—
ZNWR			—	—	—	—	—	—	—
			—	—	—	—	—	—	—
RFRP			—	—	—	—	—	—	—
RTOP			—	—	—	—	—	—	—
UNIRD UNIRDP	n = 1		96	80	74	79	34	34	34
	n = 16		440	370	340				
TRACE	Start		—	—	—	176	76	76	76
	completion of the TRACE instruction					6.3	2.7	2.7	2.7
TRACER			—	—	—	19	8.2	8.2	8.2
SP.FWRITE			—	—	—	84	36	36	36
SP.FREAD			—	—	—	82	35	35	35
PLOADP			—	—	—	58	25	25	—
PUNLOADP			—	—	—	272	117	117	—
PSWAPP			—	—	—	308	133	133	—
RBMOV	When standard RAM is used	1 point	—	—	—	45.5	20	20	20
		1000 points				215	91	91	91
	When SRAM card is used	1 point	—	—	—	49.5	22	22	22
		1000 points				540	305	305	305

Tab. A-9: *Processing times for QCPU (except Universal model CPU)*

A.2.2 Instructions executable by the product with the first 5 digits of the serial No. "04122" or higher (Basic model QCPU)

Instruction	Condition (Device)			Processing time (μs)		
				Q00J	Q00	Q01
LDE =	single precision	continuity		43.0	35.5	33
		no continuity		46.0	38.0	35.5
ANDE =	single precision	not executed		1.5	1.2	1.0
		executed	continuity	35.5	29.5	26.5
			no continuity	42.0	35.0	32.5
ORE =	single precision	not executed		1.5	1.2	1.0
		executed	continuity	42.0	35.0	32.5
			no continuity	37.0	31.0	28.5
LDE < >	single precision	continuity		46.0	38.0	35.5
		no continuity		43.5	36.0	33.0
ANDE < >	single precision	not executed		1.5	1.2	1.0
		executed	continuity	38.5	31.5	29.0
			no continuity	39.5	33.0	30.5
ORE < >	single precision	not executed		1.5	1.2	1.0
		executed	continuity	45.0	37.5	35.0
			no continuity	34.5	29.0	26.5
LDE >	single precision	continuity		46.0	37.5	35.5
		no continuity		46.0	38.5	35.0
ANDE >	single precision	not executed		1.5	1.2	1.0
		executed	continuity	38.5	32.0	29.0
			no continuity	42.0	35.0	32.5
ORE >	single precision	not executed		1.5	1.2	1.0
		executed	continuity	45.0	37.5	34.5
			no continuity	37.0	31.0	29.0
LDE < =	single precision	continuity		45.5	37.5	35.0
		no continuity		46.5	38.5	35.5
ANDE < =	single precision	not executed		1.5	1.2	1.0
		executed	continuity	38.5	31.5	29.0
			no continuity	42.5	35.5	32.5
ORE < =	single precision	not executed		1.5	1.2	1.0
		executed	continuity	45.0	37.5	34.5
			no continuity	37.5	31.5	28.5
LDE <	single precision	continuity		45.5	37.5	35
		no continuity		46.5	38.5	35.5
ANDE <	single precision	not executed		1.5	1.2	1.0
		executed	continuity	38.0	31.5	29.0
			no continuity	42.5	35.5	32.5
ORE <	single precision	not executed		1.5	1.2	1.0
		executed	continuity	45.0	37.5	34.5
			no continuity	37.5	31.5	29
LDE > =	single precision	continuity		45.5	38.0	35.5
		no continuity		46.5	38.0	35.0
ANDE > =	single precision	not executed		1.5	1.2	1.0
		executed	continuity	38.5	32.0	29.0
			no continuity	42.5	35.5	32.5
ORE > =	single precision	not executed		1.5	1.2	1.0
		executed	continuity	45.0	38.5	34.5
			no continuity	37.5	31.0	28.5

Tab. A-10: *Processing times for Basic model QCPU with serial no. 04122... or higher*

Instruction	Condition (Device)		Processing time (μs)		
			Q00J	Q00	Q01
E+ (s, d) E+P (s, d)	single precision	s = 0, d = 0	29.5	25.0	23.0
		$s = 2^{127}$, $d = 2^{127}$	65.5	60.5	49.5
E+ (s1, s2, d) E+P (s1, s2, d)	single precision	s1 = 0, s2 = 0	31.0	27.0	24.0
		$s1 = 2^{127}$, $s2 = 2^{127}$	66.5	56.0	51.0
E - (s, d) E -P (s, d)	single precision	s = 0, d = 0	29.5	25.0	23.0
		$s = 2^{127}$, $d = 2^{127}$	48.5	41.0	37.5
E - (s1, s2, d) E -P (s1, s2, d)	single precision	s1 = 0, s2 = 0	31.0	27.0	24.0
		$s1 = 2^{127}$, $s2 = 2^{127}$	50.5	42.5	38.5
E* (s1, s2, d) E*P (s1, s2, d)	single precision	s1 = 0, s2 = 0	30.0	25.5	23.0
		$s1 = 2^{127}$, $s2 = 2^{127}$	65.5	55.0	49.5
E/ (s1, s2, d) E/P (s1, s2, d)	single precision	s1 = 0, s2 = 0	30.0	26.0	23.0
		$s1 = 2^{127}$, $s2 = -2^{126}$	69.5	57.5	53.0
INT INTP	single precision	s = 0	21.5	18.5	16.0
		s = 32766.5	38.0	32.0	29.5
DINT DINTP	single precision	s = 0	23.0	19.5	17.5
		s = 1234567890.3	42.0	35.5	32.0
FLT FLTP	single precision	s = 0	22.5	19.5	17.0
		$s = 7FFF_H$	26.5	23.0	20.0
DFLT DFLTP	single precision	s = 0	23.0	20.0	17.5
		$s = 7FFFFFFF_H$	26.0	23.5	19.5
ENEG ENEGP	s = 0		20.5	17.0	15.5
	s = E - 1.0		31.5	26.0	24.0
EMOV EMOVP	—		1.5	1.2	1.0
ESTR ESTRP	—		604.0	686.0	831.0
EVAL EVALP	decimal point format all 2-digit specification		138.0	148.0	196.0
	exponent format all 6-digit specification		164.0	177.0	214.0
SIN SINP	single precision		204.0	173.0	157.0
COS COSP	single precision		187.0	158.0	144.0
TAN TANP	single precision		224.0	190.0	173.0
RAD RADP	single precision		51.0	43.0	39.0
DEG DEGP	single precision		51.0	43.0	39.0
SQR SQRP	single precision		60.0	51.0	46.5
EXP EXPP	single precision	s = - 10	306.0	259.0	235.0
		s = 1	306.0	259.0	235.0
LOG LOGP	single precision	s = 1	73.0	61.5	56.0
		s = 10	301.0	255.0	232.0
RND RNDP	—		12.5	11.0	10.0
SRND SRNDP	—		13.5	12.0	11.0

Tab. A-10: *Processing times for Basic model QCPU with serial no. 04122... or higher*

Instruction	Condition/Number of Points Processed		Processing time (µs)		
			Q00J	Q00	Q01
COM [1)]	With auto refresh of CPU shared memory	Refresh range: 2k words (0.5k words assigned equally to all CPUs)	—	920	880
	Without auto refresh of CPU shared memory	—	—	150	135
FROM	Read from CPU shared memory of host CPU	n3 = 1	—	100	90
		n3 = 320	—	440	420
	Read from CPU shared memory of another CPU	n3 = 1	—	110	105
		n3 = 320	—	305	290
TO	Write to CPU shared memory of host CPU	n3 = 1	—	100	95
		n3 = 320	—	440	425
S.TO	Write to CPU shared memory of host CPU	n4 = 1	—	205	195
		n4 = 320	—	545	525

[1] If the processing overlaps those of the other CPUs in a multiple CPU system, the processing time increases by a maximum of the following time.
For a system which consists of a base unit only:
Instruction processing time increase [µs] = 4 x 0,54 x (number of points processed) x (number of CPU modules)

For a system which consists of a base unit and extension base units:
Instruction processing time increase [µs] = 4 x 1,30 x (number od points processed) x (number of CPU modules)

Tab. A-11: *Processing times for Basic model QCPU with serial no. 04122... or higher*

A.2.3 Table of the time to be added (Basic model QCPU)

When using a file register (ZR), module access device (Un\G□ and U3En\G0 to G511), and link direct device (Jn\□), add the processing time shown in the following table to that of the instruction.

Device Name	Data	Device Specification Location	Processing time (μs)		
			Q00J	Q00	Q01
File register (ZR)	Bit	Source	—	34	32
		Destination	—	23	22
	Word	Source	—	13	12
		Destination	—	9	8
	Double word	Source	—	14	13
		Destination	—	10	9
Module access device (Un\G□ , U3En\G0 to G511)	Bit	Source	99	82	77
		Destination	167	137	129
	Word	Source	74	61	58
		Destination	72	60	56
	Double word	Source	76	63	59
		Destination	92	75	71
Link direct device (Jn\ □)	Bit	Source	178	147	137
		Destination	303	248	233
	Word	Source	154	126	118
		Destination	153	125	117
	Double word	Source	155	127	119
		Destination	163	133	125

Tab. A-12: *Processing time to be added (Basic model QCPU)*

A.2.4 Instructions availabe from function version B (High Performance model QCPU/Process CPU/Redundant CPU)

Instruction	Processing (Device)			Processing time (μs)			
				High Performance		Process	Redund.
				Qn	QnH	QnPH	QnPRH
COM [1]	With automatic refresh of CPU shared memory	Refresh range = 2 k words (0.5 k words for each CPU)		720	660	660	—
		Refresh range = 4 k words (1 k words for each CPU		860	730	730	—
	Without automatic refresh of CPU shared memory			43	20	20	20
FROM [1]	Read from shared memory of another CPU	n3 = 1		59	29	29	—
		n3 = 1000		530	500	500	—
	Read from buffer memory of a special function module [2]	n3 = 1	main base unit	51	24	24	—
			extension base unit	54	27	27	—
		n3 = 1000	main base unit	540	480	480	—
			extension base unit	1100	1050	1050	—
S.TO	Writing to CPU shared memory of host CPU	n3 = 1 ("TO" instruction) n4 = 1 ("S.TO" instruction)		74	33	33	—
		n2 = 256		126	54	54	—
S(P).DATERD [3]	Reading data of the expansion clock			25	11	11	11
S(P).DATE+ [3]	Expansion clock data addition operation			38	17	17	17
S(P).DATE– [3]	Expansion clock data subtraction operation			38	17	17	17

[1] If the processing overlaps those of the other CPUs in a multiple CPU system, the processing time increases by a maximum of the following time.
For a system which consists of a base unit only:
Instruction processing time increase [μs] = 0,54 x (number of points processed) x (number of CPU modules)

For a system which consists of a base unit and extension base units:
Instruction processing time increase [μs] = 1,30 x (number of points processed) x (number of CPU modules)

[2] The instruction processing time for special function modules under control of the CPU which is executing the instruction is identical to the instruction processing time for special function modules under control of another CPU of the multi-CPU system.

[3] Products with the first 5 digits of the serial No. "07032" or higher are applicable.

Tab. A-13: *Instructions for CPU modules availabe from function version B*

A.2.5 Table of the time to be added (High Performance model QCPU/Process CPU/Redundant CPU)

When using a file register (ZR), module access device (Un\G□ and U3En\G0 to G4096), and link direct device (Jn\□), add the processing time shown in the following table to that of the instruction.

Device Name		Data	Device Specification Location	Processing time (μs)			
				Qn	QnH	QnPH	QnPRH
File register (ZR)	When standard RAM is used	Bit	Source	5.56	2.40	2.40	2.40
			Destination	4.44	1.91	1.91	1.91
		Word	Source	2.60	1.12	1.12	1.12
			Destination	3.76	1.62	1.62	1.62
		Double word	Source	2.83	1.22	1.22	1.22
			Destination	4.00	1.72	1.72	1.72
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	5.22	2.25	2.25	2.25
			Destination	4.09	1.76	1.76	1.76
		Word	Source	2.25	0.97	0.97	0.97
			Destination	3.42	1.47	1.47	1.47
		Double word	Source	2.49	1.07	1.07	1.07
			Destination	3.65	1.57	1.57	1.57
Module access device (Un\G□, U3En\G0 to G4095)		Bit	Source	35.56	15.31	15.31	15.31
			Destination	65.08	28.01	28.01	28.01
		Word	Source	32.76	14.10	14.10	14.10
			Destination	28.84	12.41	12.41	12.41
		Double word	Source	32.99	14.20	14.20	14.20
			Destination	29.07	12.51	12.51	12.51
Link direct device (Jn\ □)		Bit	Source	75.67	32.57	32.57	32.57
			Destination	138.65	59.67	59.67	59.67
		Word	Source	72.73	31.30	31.30	31.30
			Destination	137.32	59.10	59.10	59.10
		Double word	Source	72.96	31.40	31.40	31.40
			Destination	137.55	59.20	59.20	59.20

Tab. A-14: *Processing time to be added (High Performance model QCPU/Process CPU/ Redundant CPU)*

A.2.6 Redundant system instruction

Instruction	Condition (Device)	Processing time (μs)			
		Qn	QnH	QnPH	QnPRH
SP.CONTSW		—	—	—	9.6

Tab. A-15: *Processing time (instruction SP.CONTSW)*

A.3 Operation Processing Time of Universal Model QCPU

NOTES

- *The processing time shown in section A.3.1 applies when the device used in an instruction meets the device condition for subset processing (for device condition triggering subset processing, refer to section 3.8.1).*
- *When using a file register (R, ZR), extended data register (D), extended link register (W), and module access device (U3En\G10000 and the subsequent devices), add the processing time shown in tables A-18 and A-19 to that of the instruction.*
- *When using an F, T(ST), C device with an OUT/SET/RST instruction, add the processing time for each instruction, with reference to the adding time in tables A-20 and A-21.*
- *Since the processing time of an instruction varies depending on that of the cache function, both the minimum and maximum values are described in the table.*

A.3.1 Subset instruction processing time

● Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
LD LDI AND ANI OR ORI LDP LDF ANDP ANDF ORP ORF	executed			0.120		0.080		0.060		0.040
LDPI LDFI	executed			0.360		0.240		0.180		0.120
ANDPI ANDFI ORPI ORFI	executed			0.480		0.320		0.240		0.160
OUT	not changed			0.120		0.080		0.060		0.040
	changed									
SET RST	not executed			0.120		0.080		0.060		0.040
	executed	not changed								
		changed								
LD=	continuity			0.360		0.240		0.180		0.120
	no continuity									
AND=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
OR=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LD<>	continuity			0.360		0.240		0.180		0.120
	no continuity									

Tab. A-16: *Processing times for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
AND<>	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
OR<>	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LD>	continuity			0.360		0.240		0.180		0.120
	no continuity									
AND>	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
OR>	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LD<=	continuity			0.360		0.240		0.180		0.120
	no continuity									
AND<=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
OR<=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LD<	continuity			0.360		0.240		0.180		0.120
	no continuity									
AND<	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
OR<	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LD>=	continuity			0.360		0.240		0.180		0.120
	no continuity									
AND>=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
OR>=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LDD=	continuity			0.360		0.240		0.180		0.120
	no continuity									
ANDD=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
ORD=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LDD<>	continuity			0.360		0.240		0.180		0.120
	no continuity									

Tab. A-16: *Processing times for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ANDD<>	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
ORD<>	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LDD>	continuity			0.360		0.240		0.180		0.120
	no continuity									
ANDD>	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
ORD>	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LDD<=	continuity			0.360		0.240		0.180		0.120
	no continuity									
ANDD<=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
ORD<=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LDD<	continuity			0.360		0.240		0.180		0.120
	no continuity									
ANDD<	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
ORD<	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
LDD>=	continuity			0.360		0.240		0.180		0.120
	no continuity									
ANDD>=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
ORD>=	not executed			0.360		0.240		0.180		0.120
	executed	continuity								
		no continuity								
+ (s, d)	executed			0.360		0.240		0.180		0.120
+ (s1, s2, d)	executed			0.480		0.320		0.240		0.160
- (s, d)	executed			0.360		0.240		0.180		0.120
- (s1, s2, d)	executed			0.480		0.320		0.240		0.160
d + (s, d)	executed			0.360		0.240		0.180		0.120
d + (s1, s2, d)	executed			0.480		0.320		0.240		0.160
d - (s, d)	executed			0.360		0.240		0.180		0.120
d - (s1, s2, d)	executed			0.480		0.320		0.240		0.160
* (s1, s2, d)	executed			0.420		0.300		0.240		0.180
/ (s1, s2, d)	executed			0.520		0.400		0.340		0.280
d * (s1, s2, d)	executed			0.500		0.380		0.320		0.260
d/ (s1, s2, d)	executed			0.640		0.520		0.460		0.400

Tab. A-16: *Processing times for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
B + (s, d)	executed		3.100	12.300	3.100	12.300	3.100	12.300	3.300	8.300
B + (s1, s2, d)	executed		5.900	13.500	5.900	13.500	5.900	13.500	4.600	6.200
B - (s, d)	executed		3.150	12.300	3.150	12.300	3.150	12.300	3.300	9.000
B - (s1, s2, d)	executed		5.950	13.600	5.950	13.600	5.950	13.600	4.600	8.200
B * (s1, s2, d)	executed		3.700	12.100	3.700	12.100	3.700	12.100	4.000	8.200
B/ (s1, s2, d)	executed		4.000	14.000	4.000	14.000	4.000	14.000	4.200	12.400
E + (s, d)	single precision	s = 0, d = 0		0.420		0.300		0.240		0.180
		$s = 2^{127}$, $d = 2^{127}$		0.420		0.300		0.240		0.180
E + (s1, s2, d)	single precision	s1 = 0, s2 = 0		0.540		0.380		0.300		0.220
		$s1 = 2^{127}$, $s2 = 2^{127}$		0.540		0.380		0.300		0.220
E - (s, d)	single precision	s = 0, d = 0		0.420		0.300		0.240		0.180
		$s = 2^{127}$, $d = 2^{127}$		0.420		0.300		0.240		0.180
E - (s1, s2, d)	single precision	s1 = 0, s2 = 0		0.540		0.380		0.300		0.220
		$s1 = 2^{127}$, $s2 = 2^{127}$		0.540		0.380		0.300		0.220
E * (s1, s2, d)	single precision	s1 = 0, s2 = 0		0.420		0.300		0.240		0.180
		$s1 = 2^{127}$, $s2 = 2^{127}$		0.420		0.300		0.240		0.180
E/ (s1, s2, d)	single precision	$s1 = 2^{127}$, $s2 = 2^{127}$	4.900	18.900	4.900	18.900	4.900	18.900	5.100	14.100
INC	executed			0.240		0.160		0.120		0.080
DINC	executed			0.240		0.160		0.120		0.080
DEC	executed			0.240		0.160		0.120		0.080
DDEC	executed			0.240		0.160		0.120		0.080
BCD	executed			0.320		0.240		0.200		0.160
DBCD	executed			0.400		0.320		0.280		0.240
BIN	executed			0.260		0.180		0.140		0.100
DBIN	executed			0.260		0.180		0.140		0.100
FLT	single precision	s = 0		0.300		0.220		0.180		0.140
		$s = 7FFF_H$		0.300		0.220		0.180		0.140
DFLT	single precision	s = 0		0.300		0.220		0.180		0.140
		$s = 7FFFFFFF_H$		0.300		0.220		0.180		0.140
INT	single precision	s = 0		0.300		0.220		0.180		0.140
		s = 32766.5		0.300		0.220		0.180		0.140
DINT	single precision	s= 0		0.300		0.220		0.180		0.140
		s= 1234567890.3		0.300		0.220		0.180		0.140
MOV	—			0.240		0.160		0.120		0.080
DMOV	—			0.240		0.160		0.120		0.080
EMOV	—			0.240		0.160		0.120		0.080
CML	—			0.240		0.160		0.120		0.080
DCML	—			0.240		0.160		0.120		0.080
BMOV	SM237= ON	n=1	4.200	4.600	4.200	4.600	4.200	4.600	4.100	4.500
		n=96	4.850	5.150	4.850	5.150	4.850	5.150	4.700	5.100
	SM237= OFF	n=1	6.800	11.300	6.800	11.300	6.800	11.300	6.300	8.900
		n=96	7.450	11.900	7.450	11.900	7.450	11.900	5.900	9.500
FMOV	SM=237 =ON	n=1	4.100	4.600	4.100	4.600	4.100	4.600	4.100	4.600
		n=96	4.800	5.200	4.800	5.200	4.800	5.200	4.800	5.200
	SM237= OFF	n=1	4.600	8.250	4.600	8.250	4.600	8.250	4.600	7.900
		n=96	6.150	10.600	6.150	10.600	6.150	10.600	5.300	8.500
XCH	—		2.250	8.100	2.250	8.100	2.250	8.100	2.500	6.000
DXCH	—		2.400	8.200	2.400	8.200	2.400	8.200	2.800	7.900

Tab. A-16: *Processing times for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
DFMOV	SM237= ON	n=1	2.700	2.800	2.700	2.800	2.700	2.800	2.350	2.450
		n=96	6.500	6.800	6.500	6.800	6.500	6.800	5.950	6.000
	SM237= OFF	n=1	4.000	8.150	4.000	8.150	4.000	8.150	3.000	6.950
		n=96	8.000	12.200	8.000	12.200	8.000	12.200	6.600	10.600
CJ	—		3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100
SCJ	—		3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100
JMP	—		3,500	10,100	3,500	10,100	3,500	10,100	1,900	10,100
WAND (s, d)	executed			0.360		0.240		0.180		0.120
WAND (s1, s2, d)	executed			0.480		0.320		0.240		0.160
DAND (s, d)	executed			0.360		0.240		0.180		0.120
DAND (s1, s2, d)	executed			0.480		0.320		0.240		0.160
WOR (s, d)	executed			0.360		0.240		0.180		0.120
WOR (s1, s2, d)	executed			0.480		0.320		0.240		0.160
DOR (s, d)	executed			0.360		0.240		0.180		0.120
DOR (s1, s2, d)	executed			0.480		0.320		0.240		0.160
WXOR (s, d)	executed			0.360		0.240		0.180		0.120
WXOR (s1, s2, d)	executed			0.480		0.320		0.240		0.160
DXOR (s, d)	executed			0.360		0.240		0.180		0.120
DXOR (s1, s2, d)	executed			0.480		0.320		0.240		0.160
WXNR (s, d)	executed			0.360		0.240		0.180		0.120
WXNR (s1, s2, d)	executed			0.480		0.320		0.240		0.160
DXNR (s, d)	executed			0.360		0.240		0.180		0.120
DXNR (s1, s2, d)	executed			0.480		0.320		0.240		0.160
ROR (d, n)	n = 1		2.250	10.800	2.250	10.800	2.250	10.800	2.300	7.800
	n = 15		2.250	10.800	2.350	10.800	2.350	10.800	2.400	7.800
RCR (d, n)	n = 1		2.250	10.800	2.250	10.800	2.250	10.800	2.300	3.900
	n = 15		2.250	10.800	2.250	10.800	2.250	10.800	2.400	4.100
ROL (d, n)	n = 1		2.250	10.800	2.350	10.800	2.350	10.800	2.500	4.600
	n = 15		2.250	10.800	2.350	10.800	2.350	10.800	2.400	4.600
RCL (d, n)	n = 1		2.250	11.500	2.300	11.500	2.300	11.500	2.400	7.500
	n = 15		2.250	11.500	2.300	11.500	2.300	11.500	2.500	7.500
DROR (d, n)	n = 1		2.350	11.500	2.350	11.500	2.350	11.500	2.400	10.300
	n = 31		2.350	11.500	2.350	11.500	2.350	11.500	2.500	10.300
DRCR (d, n)	n = 1		2.350	13.300	2.350	13.300	2.350	13.300	2.500	12.700
	n = 31		2.350	14.900	2.350	14.900	2.350	14.900	2.500	12.700
DROL (d, n)	n = 1		2.350	10.800	2.350	10.800	2.350	10.800	2.500	11.800
	n = 31		2.350	10.800	2.350	10.800	2.350	10.800	2.500	11.800
DRCL (d, n)	n = 1		2.350	13.300	2.350	13.300	2.350	13.300	2.500	5.100
	n = 31		2.350	13.300	2.350	13.300	2.350	13.300	2.500	5.100
SFR (d, n)	n = 1		2.350	9.900	2.350	9.900	2.350	9.900	2.400	6.100
	n = 15		2.350	9.900	2.350	9.900	2.350	9.900	2.300	5.700
SFL (d, n)	n = 1		2.350	9.850	2.350	9.850	2.350	9.850	2.400	4.300
	n = 15		2.350	9.850	2.350	9.850	2.350	9.850	2.400	4.300
DSFR (d, n)	n = 1		3.250	15.500	3.250	15.500	3.250	15.500	3.300	12.000
	n = 96		32.600	45.000	32.600	45.000	32.600	45.000	32.600	42.200
DSFL (d, n)	n = 1		3.200	15.500	3.200	15.500	3.200	15.500	3.300	8.200
	n = 96		32.600	45.100	32.600	45.100	32.600	45.100	32.600	37.700
SUM	s = 0		3.100	8.950	3.100	8.950	3.100	8.950	3.400	6.700
	s = FFFFH		3.000	8.850	3.000	8.850	3.000	8.850	3.500	6.700
SEG	executed		2.100	7.700	2.100	7.700	2.100	7.700	2.100	5.900

Tab. A-16: *Processing times for Universal model QCPU (1)*

Instruction	Processing (Device)	Processing time (μs)							
		Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
FOR	—	1.500	7.500	1.500	7.500	1.500	7.500	1.200	6.300
CALL pn	Internal file pointer	4.800	5.400	4.800	5.400	4.800	5.400	2.700	4.800
	Common pointer	7.100	30.500	7.100	30.500	7.100	30.500	4.400	5.700
CALL pn s1 to s5	—	50.200	62.000	50.200	62.000	50.200	62.000	28.700	42.600

Tab. A-16: *Processing times for Universal model QCPU (1)*

NOTE *For the instructions for which a leading edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.*

Example: MOVP instruction, WANDP instruction etc.

- Q03UD(E)HCPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, and Q100UDEHCPU

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
LD LDI AND ANI OR ORI LDP LDF ANDP ANDF ORP ORF	executed			0.020		0.0095		0.0095		0.0095
LDPI LDFI	executed			0.060		0.0285		0.0285		0.0285
ANDPI ANDFI ORPI ORFI	executed			0.080		0.038		0.038		0.038
OUT	changed			0.020		0.0095		0.0095		0.0095
	not changed									
SET RST	not executed			0.020		0.0095		0.0095		0.0095
LD=	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
AND=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
OR=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LD<>	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									

Tab. A-17: *Processing times for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
AND<>	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
OR<>	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LD>	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
AND>	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
OR>	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LD<=	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
AND<=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
OR<=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LD<	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
AND<	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
OR<	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LD>=	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
AND>=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
OR>=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LDD=	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
ANDD=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
ORD=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LDD<>	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									

Tab. A-17: *Processing times for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ANDD<>	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
ORD<>	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LDD>	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
ANDD>	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
ORD>	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LDD<=	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
ANDD<=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
ORD<=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LDD<	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
ANDD<	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
ORD<	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
LDD>=	continuity			0.060		0.0285		0.0285		0.0285
	no continuity									
ANDD>=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
ORD>=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity								
		no continuity								
+ (s, d)	executed			0.060		0.0285		0.0285		0.0285
+ (s1, s2, d)	executed			0.080		0.038		0.038		0.038
- (s, d)	executed			0.060		0.0285		0.0285		0.0285
– (s1, s2, d)	executed			0.080		0.038		0.038		0.038
d + (s, d)	executed			0.060		0.0285		0.0285		0.0285
d + (s1, s2, d)	executed			0.080		0.038		0.038		0.038
d - (s, d)	executed			0.060		0.0285		0.0285		0.0285
d - (s1, s2, d)	executed			0.080		0.038		0.038		0.038
* (s1, s2, d)	executed			0.120		0.057		0.057		0.057
/ (s1, s2, d)	executed			0.220		0.110		0.110		0.110
d * (s1, s2, d)	executed			0.200		0.095		0.095		0.095
d/ (s1, s2, d)	executed			0.340		0.170		0.170		0.170

Tab. A-17: *Processing times for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
B + (s, d)	executed		3.300	5.500	3.000	4.100	3.000	4.100	3.000	4.100
B + (s1, s2, d)	executed		4.600	6.200	4.200	5.900	4.200	5.900	4.200	5.900
B -(s, d)	executed		3.300	4.400	2.900	3.800	2.900	3.800	2.900	3.800
B - (s1, s2, d)	executed		4.600	6.300	4.200	4.600	4.200	4.600	4.200	4.600
B * (s1, s2, d)	executed		4.000	4.800	3.400	4.800	3.400	4.800	3.400	4.800
B/ (s1, s2, d)	executed		4.200	5.700	3.700	5.200	3.700	5.200	3.700	5.200
E + (s, d)	single precision	s = 0, d = 0		0.120		0.057		0.057		0.057
		$s = 2^{127}$, $d = 2^{127}$		0.120		0.057		0.057		0.057
E + (s1, s2, d)	single precision	s1 = 0, s2 = 0		0.140		0.0665		0.0665		0.0665
		$s1 = 2^{127}$, $s2 = 2^{127}$		0.140		0.0665		0.0665		0.0665
E - (s, d)	single precision	s = 0, d = 0		0.120		0.057		0.057		0.057
		$s = 2^{127}$, $d = 2^{127}$		0.120		0.057		0.057		0.057
E - (s1, s2, d)	single precision	s1 = 0, s2 = 0		0.140		0.0665		0.0665		0.0665
		$s1 = 2^{127}$, $s2 = 2^{127}$		0.140		0.0665		0.0665		0.0665
E * (s1, s2, d)	single precision	s1 = 0, s2 = 0		0.120		0.057		0.057		0.057
		$s1 = 2^{127}$, $s2 = 2^{127}$		0.120		0.057		0.057		0.057
E/ (s1, s2, d)	single precision	$s1 = 2^{127}$, $s2 = 2^{127}$	4.500	5.600	3.900	4.900		0.285		0.285
INC	executed			0.040		0.019		0.019		0.019
DINC	executed			0.040		0.019		0.019		0.019
DEC	executed			0.040		0.019		0.019		0.019
DDEC	executed			0.040		0.019		0.019		0.019
BCD	executed			0.120		0.057		0.057		0.057
DBCD	executed			0.200		0.095		0.095		0.095
BIN	executed			0.060		0.0285		0.0285		0.0285
DBIN	executed			0.060		0.0285		0.0285		0.0285
FLT	single precision	s = 0		0.100		0.0475		0.0475		0.0475
		$s = 7FFF_H$		0.100		0.0475		0.0475		0.0475
DFLT	single precision	s = 0		0.100		0.0475		0.0475		0.0475
		$s = 7FFFFFFF_H$		0.100		0.0475		0.0475		0.0475
INT	single precision	s = 0		0.100		0.0475		0.0475		0.0475
		s = 32766.5		0.100		0.0475		0.0475		0.0475
DINT	single precision	s = 0		0.100		0.0475		0.0475		0.0475
		s = 1234567890.3		0.100		0.0475		0.0475		0.0475
MOV	—			0.040		0.019		0.019		0.019
DMOV	—			0.040		0.019		0.019		0.019
EMOV	—			0.040		0.019		0.019		0.019
CML	—			0.040		0.019		0.019		0.019
DCML	—			0.040		0.019		0.019		0.019

Tab. A-17: *Processing times for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
BMOV			6.300	8.200	5.400	7.000	5.400	7.000	5.400	7.000
	n = 1	SM237=OFF [1]	8.200	10.600	3.900	5.100	3.900	5.100	3.900	5.100
		SM237=ON [1]	6.000	7.800	2.900	3.700	2.900	3.700	2.900	3.700
	n = 96	SM237=OFF [1]	9.300	11.900	4.400	5.700	4.400	5.700	4.400	5.700
		SM237=ON [1]	7.100	9.100	3.400	4.300	3.400	4.300	3.400	4.300
FMOV			5.300	5.900	4.200	4.800	4.200	4.800	4.200	4.800
	n = 1	SM237=OFF [1]	7.000	8.000	3.400	3.800	3.400	3.800	3.400	3.800
		SM237=ON [1]	5.900	6.800	2.800	3.200	2.800	3.200	2.800	3.200
	n = 96	SM237=OFF [1]	7.400	12.200	3.600	5.800	3.600	5.800	3.600	5.800
		SM237=ON [1]	6.300	11.000	3.000	5.200	3.000	5.200	3.000	5.200
XCH	—		2.500	2.900	1.800	2.300	1.800	2.300	1.800	2.300
DXCH	—		2.800	3.700	2.100	2.900	2.100	2.900	2.100	2.900
DFMOV [2]	n = 1	SM237=OFF	2.600	3.750	2.250	3.150	2.250	3.150	2.250	3.150
		SM237=ON	2.050	2.250	1.750	1.750	1.750	1.750	1.750	1.750
	n = 96	SM237=OFF	5.850	7.350	4.200	5.500	4.200	5.500	5.380	7.440
		SM237=ON	5.300	6.000	3.650	4.150	3.650	4.150	4.700	5.500
CJ	—		1.800	2.800	1.400	2.400	1.400	2.400	1.400	2.400
SCJ	—		1.800	2.800	1.400	2.400	1.400	2.400	1.400	2.400
JMP	—		1.800	2.800	1.100	2.400	1.100	2.400	1.100	2.400
WAND (s, d)	executed			0.060		0.0285		0.0285		0.0285
WAND (s1, s2, d)	executed			0.080		0.038		0.038		0.038
DAND (s, d)	executed			0.060		0.0285		0.0285		0.0285
DAND (s1, s2, d)	executed			0.080		0.038		0.038		0.038
WOR (s, d)	executed			0.060		0.0285		0.0285		0.0285
WOR (s1, s2, d)	executed			0.080		0.038		0.038		0.038
DOR (s, d)	executed			0.060		0.0285		0.0285		0.0285
DOR (s1, s2, d)	executed			0.080		0.038		0.038		0.038
WXOR (s, d)	executed			0.060		0.0285		0.0285		0.0285
WXOR (s1, s2, d)	executed			0.080		0.038		0.038		0.038
DXOR (s, d)	executed			0.060		0.0285		0.0285		0.0285
DXOR (s1, s2, d)	executed			0.080		0.038		0.038		0.038
WXNR (s, d)	executed			0.060		0.0285		0.0285		0.0285
WXNR (s1, s2, d)	executed			0.080		0.038		0.038		0.038
DXNR (s, d)	executed			0.060		0.0285		0.0285		0.0285
DXNR (s1, s2, d)	executed			0.080		0.038		0.038		0.038
ROR (d, n)	n = 1		2.300	3.100	1.700	2.500	1.700	2.500	1.700	2.500
	n = 15		2.400	3.100	1.800	2.500	1.800	2.500	1.800	2.500
RCR (d, n)	n = 1		2.300	3.900	1.700	3.200	1.700	3.200	1.700	3.200
	n = 15		2.400	4.100	1.700	3.200	1.700	3.200	1.700	3.200
ROL (d, n)	n = 1		2.400	3.300	1.800	3.200	1.800	3.200	1.800	3.200
	n = 15		2.400	3.300	1.800	3.200	1.800	3.200	1.800	3.200
RCL (d, n)	n = 1		2.400	2.700	1.800	2.100	1.800	2.100	1.800	2.100
	n = 15		2.400	2.800	1.800	2.200	1.800	2.200	1.800	2.200

[1] Can be used only for the Q03UDCPU, Q04UDHCPU and Q06UDHCPU whose first 5 digits of serial number is “10012” or higher.

[2] Can be used only for the Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q13UD(E)HCPU and Q26UD(E)HCPU whose first 5 digits of serial number is “10012” or higher.

Tab. A-17: *Processing times for Universal model QCPU (2)*

Instruction	Processing (Device)	Processing time (μs)							
		Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
DROR (d, n)	n = 1	2.400	3.400	1.900	2.700	1.900	2.700	1.900	2.700
	n = 31	2.500	3.400	1.900	2.700	1.900	2.700	1.900	2.700
DRCR (d, n)	n = 1	2.500	4.800	1.900	4.200	1.900	4.200	1.900	4.200
	n = 31	2.500	4.900	1.900	4.200	1.900	4.200	1.900	4.200
DROL (d, n)	n = 1	2.500	3.900	1.800	3.200	1.800	3.200	1.800	3.200
	n = 31	2.500	3.900	1.800	3.300	1.800	3.300	1.800	3.300
DRCL (d, n)	n = 1	2.500	4.800	1.900	3.800	1.900	3.800	1.900	3.800
	n = 31	2.500	4.600	1.900	3.800	1.900	3.800	1.900	3.800
SFR (d, n)	n = 1	2.400	3.900	1.700	2.600	1.700	2.600	1.700	2.600
	n = 15	2.300	3.900	1.800	2.600	1.800	2.600	1.800	2.600
SFL (d, n)	n = 1	2.400	4.300	1.800	2.700	1.800	2.700	1.800	2.700
	n = 15	2.400	4.300	1.800	2.700	1.800	2.700	1.800	2.700
DSFR (d, n)	n = 1	2.700	4.800	2.200	4.300	2.200	4.300	2.200	4.300
	n = 96	32.600	35.900	23.900	26.100	23.900	26.100	23.900	26.100
DSFL (d, n)	n = 1	2.700	4.600	2.100	4.000	2.100	4.000	2.100	4.000
	n = 96	32.600	35.300	23.700	25.800	23.700	25.800	23.700	25.800
SUM	s = 0	3.400	4.300	2.900	3.600	2.900	3.600	2.900	3.600
	s = FFFFH	3.500	4.200	2.900	3.600	2.900	3.600	2.900	3.600
SEG	executed	2.100	2.800	1.500	2.100	1.500	2.100	1.500	2.100
FOR	—	1.200	2.400	0.870	2.100	0.870	2.100	0.870	2.100
CALL pn	Intern file pointer	2.600	4.000	2.300	3.600	2.300	3.600	2.300	3.600
	Common pointer	4.000	5.300	3.200	4.900	3.200	4.900	3.200	4.900
CALL pn s1 to s5	—	28.700	33.400	26.100	29.300	26.100	29.300	26.100	29.300

Tab. A-17: *Processing times for Universal model QCPU (2)*

NOTE *For the instructions for which a leading edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.*

Example: MOVP instruction, WANDP instruction etc.

Table of the time to be added when file register, extended data register, extended link register and module access device are used

- Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Device Name		Data	Device Specification Location	Processing time (μs)			
				Q00UJ	Q00U	Q01U	Q02U
File register (R)	When standard RAM is used	Bit	Source	0.100	0.100	0.100	0.100
			Destination	0.220	0.220	0.220	0.220
		Word	Source	0.100	0.100	0.100	0.100
			Destination	0.100	0.100	0.100	0.100
		Double word	Source	0.200	0.200	0.200	0.200
			Destination	0.200	0.200	0.200	0.200
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.220
			Destination	—	—	—	0.420
		Word	Source	—	—	—	0.220
			Destination	—	—	—	0.180
		Double word	Source	—	—	—	0.440
			Destination	—	—	—	0.380
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.160
			Destination	—	—	—	0.320
		Word	Source	—	—	—	0.160
			Destination	—	—	—	0.140
		Double word	Source	—	—	—	0.320
			Destination	—	—	—	0.300
File register (ZR), Extended data register (D), Extended link register (W)	When standard RAM is used	Bit	Source	0.220	0.180	0.160	0.140
			Destination	0.280	0.320	0.300	0.280
		Word	Source	0.220	0.180	0.160	0.140
			Destination	0.220	0.180	0.160	0.140
		Double word	Source	0.320	0.280	0.260	0.240
			Destination	0.320	0.280	0.260	0.240
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.260
			Destination	—	—	—	0.480
		Word	Source	—	—	—	0.260
			Destination	—	—	—	0.220
		Double word	Source	—	—	—	0.480
			Destination	—	—	—	0.420
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.200
			Destination	—	—	—	0.380
		Word	Source	—	—	—	0.200
			Destination	—	—	—	0.180
		Double word	Source	—	—	—	0.360
			Destination	—	—	—	0.340
Module access device (Multiple CPU high speed transmission area) (U3En\G10000)		Bit	Source	—	—	—	—
			Destination	—	—	—	—
		Word	Source	—	—	—	—
			Destination	—	—	—	—
		Double word	Source	—	—	—	—
			Destination	—	—	—	—

Tab. A-18: *Processing times to be added for subset instructions for Universal model CPU (1)*

- Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UDE(H)CPU,Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU and Q100UDEHCPU

Device Name		Data	Device Specification Location	Processing time (μs)			
				Q03UD(E)	Q04/ Q06UD(E)H	Q10/Q13/ Q20/ Q26UD(E)H	Q50/ Q100UDEH
File register (R)	When standard RAM is used	Bit	Source	0.100	0.048	0.048	0.048
			Destination	0.100	0.038	0.038	0.038
		Word	Source	0.100	0.048	0.048	0.048
			Destination	0.100	0.038	0.038	0.038
		Double word	Source	0.200	0.095	0.095	0.095
			Destination	0.200	0.086	0.086	0.086
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.220	0.200	0.200	0.200
			Destination	0.180	0.162	0.162	0.162
		Word	Source	0.220	0.200	0.200	0.200
			Destination	0.180	0.162	0.162	0.162
		Double word	Source	0.440	0.399	0.399	0.399
			Destination	0.380	0.361	0.361	0.361
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.160	0.152	0.152	0.152
			Destination	0.140	0.133	0.133	0.133
		Word	Source	0.160	0.152	0.152	0.152
			Destination	0.140	0.133	0.133	0.133
		Double word	Source	0.320	0.304	0.304	0.304
			Destination	0.300	0.295	0.295	0.295
File register (ZR), Extended data register (D), Extended link register (W)	When standard RAM is used	Bit	Source	0.120	0.057	0.057	0.057
			Destination	0.120	0.048	0.048	0.048
		Word	Source	0.120	0.057	0.057	0.057
			Destination	0.120	0.048	0.048	0.048
		Double word	Source	0.220	0.105	0.105	0.105
			Destination	0.220	0.095	0.095	0.095
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.240	0.209	0.209	0.209
			Destination	0.200	0.171	0.171	0.171
		Word	Source	0.240	0.209	0.209	0.209
			Destination	0.200	0.171	0.171	0.171
		Double word	Source	0.460	0.409	0.409	0.409
			Destination	0.400	0.371	0.371	0.371
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.180	0.162	0.162	0.162
			Destination	0.160	0.143	0.143	0.143
		Word	Source	0.180	0.162	0.162	0.162
			Destination	0.160	0.143	0.143	0.143
		Double word	Source	0.340	0.314	0.314	0.314
			Destination	0.320	0.304	0.304	0.304
Module access device (Multiple CPU high speed transmission area) (U3En\G10000)		Bit	Source	0.220	0.181	0.181	0.181
			Destination	0.140	0.105	0.105	0.105
		Word	Source	0.220	0.181	0.181	0.181
			Destination	0.140	0.105	0.105	0.105
		Double word	Source	0.500	0.437	0.437	0.437
			Destination	0.340	0.285	0.285	0.285

Tab. A-19: *Processing times to be added for subset instructions for Universal model CPU (2)*

Table of the time to be added when F/T(ST)/C device is used in OUT/SET/RST instruction

- Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Instruction Name	Device Name	Condition		Processing time (μs)			
				Q00UJ	Q00U	Q01U	Q02U
OUT	F	not executed		2.900	2.900	2.900	2.100
		executed	when displayed	116.000	116.000	116.000	68.800
			display completed	116.000	116.000	116.000	61.600
	T(ST), C	not executed		0.360	0.240	0.180	0.120
		executed	after time up	0.360	0.240	0.180	0.120
			when added	0.360	0.240	0.180	0.120
SET	F	not executed		0.120	0.080	0.006	0.004
		executed	when displayed	116.000	116.000	116.000	68.600
			display completed	116.000	116.000	116.000	65.700
RST	F	not executed		0.120	0.080	0.006	0.004
		executed	when displayed	55.800	55.800	55.800	26.500
			display completed	29.200	29.200	29.200	21.600
	T(ST), C	not executed		0.360	0.240	0.180	0.120
		executed		0.360	0.240	0.180	0.120

Tab. A-20: *Processing times to be added for Universal model CPU and OUT/SET/RST instructions (1)*

- Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UDE(H)CPU,Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU and Q100UDEHCPU

Instruction Name	Device Name	Condition		Processing time (μs)			
				Q03UD(E)	Q04/ Q06UD(E)H	Q10/Q13/ Q20/ Q26UD(E)H	Q50/ Q100UDEH
OUT	F	not executed		1.940	1.570	1.570	1.570
		executed	when displayed	39.930	38.090	38.090	38.090
			display completed	39.750	37.980	37.980	37.980
	T(ST), C	not executed		0.060	0.030	0.030	0.030
		executed	after time up	0.060	0.030	0.030	0.030
			when added	0.000	0.000	0.000	0.000
SET	F	not executed		42.900	40.600	40.600	40.600
		executed	when displayed	39.270	37.900	37.900	37.900
			display completed	0.000	0.000	0.000	0.000
RST	F	not executed		45.260	36.600	36.600	36.600
		executed	when displayed	19.020	16.190	16.190	16.190
			display completed	0.060	0.030	0.030	0.030
	T(ST), C	not executed		0.060	0.030	0.030	0.030
		executed		1.940	1.570	1.570	1.570

Tab. A-21: *Processing times to be added for Universal model CPU and OUT/SET/RST instructions (2)*

A.3.2 Processing time of instructions other than subset instruction

NOTES

- *The processing time shown in tables A-22 and A-23 applies when the device used in an instruction does not meet the device condition for subset processing (For device condition that does not trigger subset processing, refer to section 3.8.1).*
- *For instructions not shown in the following table, refer to tables A-16 and A-17 in section A.3.1.*
- *When using a file register (R, ZR), extended data register (D), extended link register (W), module access device (Un\G□ and U3En\G0 to G4095), and link direct device (Jn\□), add the processing time shown in tables A-24 and A-25 to that of the instruction.*
- *Since the processing time of an instruction varies depending on that of the cache function, both the minimum and maximum values are described in the table.*

● Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Instruction	Processing (Device)			Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ANB ORB MPS MRD MPP	—				0.120		0.080		0.060		0.040
INV	not executed				0.120		0.080		0.060		0.040
	executed										
MEP MEF	not executed				0.120		0.080		0.060		0.040
	executed										
EGP EGF	not executed				0.120		0.080		0.060		0.040
	executed										
PLS	—			1.800	1.900	1.800	1.900	1.800	1.900	1.300	1.600
PLF	—			1.800	1.900	1.800	1.900	1.800	1.900	1.600	1.700
FF	not executed				0.240		0.160		0.120		0.080
	executed			1.700	1.800	1.700	1.800	1.700	1.800	1.200	1.500
DELTA	not executed				0.240		0.160		0.120		0.080
	executed			4.000	14.700	4.000	14.700	4.000	14.700	2.800	3.600
SFT	not executed				0.240		0.160		0.120		0.800
	executed			1.800	12.600	1.800	12.600	1.800	12.600	1.600	6.600
MC	—				0.240		0.160		0.120		0.080
MCR	—				0.120		0.080		0.060		0.040
FEND END	Error check performed			250.000	250.000	250.000	250.000	250.000	250.000	175.000	252.000
	No error check performed			250.000	250.000	250.000	250.000	250.000	250.000	175.000	221.000
NOP NOPLF PAGE	—				0.120		0.080		0.060		0.040
LDE=	single precision	continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.100
		no continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.100
ANDE=	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	19.600	4.200	19.600	4.200	19.600	4.200	12.500
		no continuity		4.200	19.600	4.200	19.600	4.200	19.600	4.400	11.900
ORE=	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	17.400	4.200	17.400	4.200	17.400	4.600	10.800
		no continuity		4.200	17.400	4.200	17.400	4.200	17.400	4.500	9.800
LDE< >	single precision	continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	7.700
		no continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.600	8.200

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)			Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ANDE< >	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	19.600	4.200	19.600	4.200	19.600	4.300	14.200
		no continuity		4.200	19.600	4.200	19.600	4.200	19.600	4.400	14.200
ORE< >	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	17.400	4.200	17.400	4.200	17.400	4.600	6.700
		no continuity		4.200	17.400	4.200	17.400	4.200	17.400	4.400	6.600
LDE>	single precision	continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	13.700
		no continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.600	13.700
ANDE>	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	19.600	4.200	19.600	4.200	19.600	4.300	8.100
		no continuity		4.200	19.600	4.200	19.600	4.200	19.600	4.200	8.100
ORE>	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	17.400	4.200	17.400	4.200	17.400	4.600	8.500
		no continuity		4.200	17.400	4.200	17.400	4.200	17.400	4.400	8.100
LDE<=	single precision	continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.100
		no continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	9.600
ANDE<=	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	19.600	4.200	19.600	4.200	19.600	4.100	7.800
		no continuity		4.200	19.600	4.200	19.600	4.200	19.600	4.400	8.200
ORE<=	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	17.400	4.200	17.400	4.200	17.400	4.500	10.300
		no continuity		4.200	17.400	4.200	17.400	4.200	17.400	4.400	9.800
LDE<	single precision	continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.500
		no continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.900
ANDE<	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	19.600	4.200	19.600	4.200	19.600	4.300	9.200
		no continuity		4.200	19.600	4.200	19.600	4.200	19.600	4.400	9.400
ORE<	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	17.400	4.200	17.400	4.200	17.400	4.600	10.400
		no continuity		4.200	17.400	4.200	17.400	4.200	17.400	4.400	9.800
?LDE>=	single precision	continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	12.200
		no continuity		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.800
ANDE>=	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	19.600	4.200	19.600	4.200	19.600	4.100	6.700
		no continuity		4.200	19.600	4.200	19.600	4.200	19.600	4.400	7.000
ORE>=	single precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.200	17.400	4.200	17.400	4.200	17.400	4.600	14.000
		no continuity		4.200	17.400	4.200	17.400	4.200	17.400	4.500	14.300
LDED=	double precision	continuity		4.700	37.400	4.700	37.400	4.700	37.400	4.200	21.000
		no continuity		4.700	37.400	4.700	37.400	4.700	37.400	5.100	21.900
ANDED=	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.500	34.700	4.500	34.700	4.500	34.700	3.800	17.800
			no continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.100	18.100
ORED=	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.700	33.200	4.700	33.200	4.700	33.200	4.100	23.800
			no continuity	4.700	33.200	4.700	33.200	4.700	33.200	4.900	25.500
LDED<>	double precision	continuity		4.700	37.400	4.700	37.400	4.700	37.400	5.100	23.500
		no continuity		4.700	37.400	4.700	37.400	4.700	37.400	4.200	22.600

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)			Processing time (μs)							
				Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ANDED<>	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.000	18.800
			no continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.000	18.700
ORED<>	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.200
			no continuity	4.700	33.200	4.700	33.200	4.700	33.200	4.100	23.400
LDED>	double precision	continuity		4.700	37.400	4.700	37.400	4.700	37.400	5.100	25.100
		no continuity		4.700	37.400	4.700	37.400	4.700	37.400	4.200	23.400
ANDED>	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.500
			no continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
ORED>	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.700	33.200	4.700	33.200	4.700	33.200	5.000	24.200
			no continuity	4.700	33.200	4.700	33.200	4.700	33.200	4.900	25.800
LDED<=	double precision	continuity		4.700	37.400	4.700	37.400	4.700	37.400	4.200	22.500
		no continuity		4.700	37.400	4.700	37.400	4.700	37.400	4.200	13.500
ANDED<=	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.600
			no continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
ORED<=	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.700	33.200	4.700	33.200	4.700	33.200	5.000	26.300
			no continuity	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.200
LDED<	double precision	continuity		4.700	37.400	4.700	37.400	4.700	37.400	5.100	25.000
		no continuity		4.700	37.400	4.700	37.400	4.700	37.400	4.200	24.100
ANDED<	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.400
			no continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
ORED<	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
			no continuity	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
LDED>=	double precision	continuity		4.700	37.400	4.700	37.400	4.700	37.400	4.200	13.100
		no continuity		4.700	37.400	4.700	37.400	4.700	37.400	4.300	13.100
ANDED>=	double precision	not executed			0.360		0.240		0.180		0.120
		executed	continuity	4.500	34.700	4.500	34.700	4.500	34.700	3.900	19.500
			no continuity	4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.800
ORED>=	double precision	not executed			0.360	0.240	0.180	0.120			
		executed	continuity	4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
			no continuity	4.700	33.200	4.700	33.200	4.700	33.200	4.200	18.500
LD$=	continuity			8.300	38.500	8.300	38.500	8.300	38.500	5.500	14.900
	no continuity			8.300	38.500	8.300	38.500	8.300	38.500	5.500	15.600
AND$=	not executed				0.360		0.240		0.180	0.120	
	executed	continuity		7.200	37.300	7.200	37.300	7.200	37.300	5.200	13.800
		no continuity		7.200	37.300	7.200	37.300	7.200	37.300	5.300	14.500
OR$=	not executed				0.360		0.240		0.180	0.120	
	executed	continuity		7.500	36.600	7.500	36.600	7.500	36.600	5.500	14.900
		no continuity		7.500	36.600	7.500	36.600	7.500	36.600	5.300	14.600
LD$< >	continuity			8.300	39.300	8.300	39.300	8.300	39.300	5.600	15.200
	no continuity			8.300	39.300	8.300	39.300	8.300	39.300	5.600	15.400

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
AND$< >	not executed			0.360		0.240		0.180		
	executed	continuity	8.000	38.200	8.000	38.200	8.000	38.200	4.300	21.500
		no continuity	8.000	38.200	8.000	38.200	8.000	38.200	4.500	23.400
OR$< >	not executed			0.360		0.240		0.180		
	executed	continuity	8.300	37.300	8.300	37.300	8.300	37.300	5.400	17.700
		no continuity	8.300	37.300	8.300	37.300	8.300	37.300	5.300	19.400
LD$>	continuity		8.300	41.600	8.300	41.600	8.300	41.600	6.400	19.200
	no continuity		8.300	41.600	8.300	41.600	8.300	41.600	5.600	20.100
AND$>	not executed			0.360		0.240		0.180		
	executed	continuity	8.000	38.100	8.000	38.100	8.000	38.100	4.500	15.400
		no continuity	8.000	38.100	8.000	38.100	8.000	38.100	4.600	15.300
OR$>	not executed			0.360		0.240		0.180		
	executed	continuity	8.200	35.700	8.200	35.700	8.200	35.700	5.400	20.000
		no continuity	8.200	35.700	8.200	35.700	8.200	35.700	5.400	22.100
LD$<=	continuity		8.300	39.200	8.300	39.200	8.300	39.200	5.800	12.800
	no continuity		8.300	39.200	8.300	39.200	8.300	39.200	6.300	13.900
AND$<=	not executed			0.360		0.240		0.180		
	executed	continuity	7.100	36.500	7.100	36.500	7.100	36.500	6.000	16.000
		no continuity	7.100	36.500	7.100	36.500	7.100	36.500	6.100	16.200
OR$<=	not executed			0.360		0.240		0.180		
	executed	continuity	7.400	35.600	7.400	35.600	7.400	35.600	4.700	14.600
		no continuity	7.400	35.600	7.400	35.600	7.400	35.600	4.600	14.400
LD$<	continuity		7.400	40.000	7.400	40.000	7.400	40.000	4.800	17.000
	no continuity		7.400	40.000	7.400	40.000	7.400	40.000	5.500	18.000
AND$<	not executed			0.360		0.240		0.180		
	executed	continuity	8.000	37.300	8.000	37.300	8.000	37.300	5.900	13.400
		no continuity	8.000	37.300	8.000	37.300	8.000	37.300	6.200	14.500
OR$<	not executed			0.360		0.240		0.180		
	executed	continuity	8.300	35.600	8.300	35.600	8.300	35.600	6.200	18.700
		no continuity	8.300	35.600	8.300	35.600	8.300	35.600	5.400	19.700
LD$>=	continuity		7.400	38.300	7.400	38.300	7.400	38.300	4.800	10.000
	no continuity		7.400	38.300	7.400	38.300	7.400	38.300	5.500	11.200
AND$>=	not executed			0.360		0.240		0.180		
	executed	continuity	7.200	37.300	7.200	37.300	7.200	37.300	4.400	21.600
		no continuity	7.200	37.300	7.200	37.300	7.200	37.300	4.500	21.800
OR$>=	not executed			0.360		0.240		0.180		
	executed	continuity	8.200	36.400	8.200	36.400	8.200	36.400	5.400	15.400
		no continuity	8.200	36.400	8.200	36.400	8.200	36.400	5.300	15.300
BKCMP = (s1, s2, d, n)	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.600
	n = 96		64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.500
BKCMP<> (s1, s2, d, n)	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500
	n = 96		66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.500
BKCMP> (s1, s2, d, n)	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	23.100
	n = 96		66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.400
BKCMP<= (s1, s2, d, n)	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500
	n = 96		64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.400
BKCMP< (s1, s2, d, n)	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.300	23.000
	n = 96		66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.500
BKCMP>= (s1, s2, d, n)	n = 1		15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500
	n = 96		64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.400

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
DBKCMP = (s1, s2, d, n)	n = 1		15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000
	n = 96		64.900	85.700	64.900	85.700	64.900	85.700	60.700	78.400
DBKCMP<> (s1, s2, d, n)	n = 1		15.700	36.300	15.700	36.300	15.700	36.300	9.350	28.900
	n = 96		67.000	87.700	67.000	87.700	67.000	87.700	62.500	80.300
DBKCMP> (s1, s2, d, n)	n = 1		15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000
	n = 96		67.000	87.700	67.000	87.700	67.000	87.700	62.600	80.300
DBKCMP<= (s1, s2, d, n)	n = 1		15.700	36.300	15.700	36.300	15.700	36.300	9.350	29.000
	n = 96		64.800	85.700	64.800	85.700	64.800	85.700	60.800	78.400
DBKCMP< (s1, s2, d, n)	n = 1		15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000
	n = 96		67.000	87.700	67.000	87.700	67.000	87.700	62.700	80.400
DBKCMP>= (s1, s2, d, n)	n = 1		15.700	36.300	15.700	36.300	15.700	36.300	9.300	29.000
	n = 96		64.800	85.700	64.800	85.700	64.800	85.700	60.700	78.400
DB + (s, d)	executed		5.750	13.300	5.750	13.300	5.750	13.300	4.900	7.500
DB + (s1, s2, d)	executed		5.650	13.200	5.650	13.200	5.650	13.200	5.200	11.000
DB - (s, d)	executed		5.750	12.700	5.750	12.700	5.750	12.700	4.900	10.200
DB - (s1, s2, d)	executed		5.650	12.600	5.650	12.600	5.650	12.600	5.200	8.600
DB * (s1, s2, d)	executed		8.750	40.200	8.750	40.200	8.750	40.200	8.300	22.200
DB/ (s1, s2, d)	executed		5.750	21.500	5.750	21.500	5.750	21.500	6.100	19.200
ED + (s, d)	double precision	s = 0, d = 0	4.500	26.700	4.500	26.700	4.500	26.700	4.800	16.800
		$s = 2^{1023}$, $d = 2^{1023}$	5.800	32.900	5.800	32.900	5.800	32.900	4.800	16.800
ED + (s1, s2, d)	double precision	s1 = 0, s2 = 0	5.450	35.400	5.450	35.400	5.450	35.400	7.100	20.100
		$s1 = 2^{1023}$, $s2 = 2^{1023}$	6.750	41.400	6.750	41.400	6.750	41.400	7.100	20.100
ED - (s, d)	double precision	s = 0, d = 0	5.200	25.900	5.200	25.900	5.200	25.900	5.000	17.300
		$s = 2^{1023}$, $d = 2^{1023}$	6.000	27.700	6.000	27.700	6.000	27.700	5.000	17.300
ED - (s1, s2, d)	double precision	s1 = 0, s2 = 0	5.550	32.900	5.550	32.900	5.550	32.900	6.000	16.300
		$s1 = 2^{1023}$, $s2 = 2^{1023}$	5.750	33.900	5.750	33.900	5.750	33.900	6.000	16.300
ED * (s1, s2, d)	double precision	s1 = 0, s2 = 0	5.550	34.400	5.550	34.400	5.550	34.400	10.500	22.300
		$s1 = 2^{1023}$, $s2 = 2^{1023}$	5.950	39.100	5.950	39.100	5.950	39.100	10.500	22.300
ED / (s1, s2, d)	double precision	$s1 = 2^{1023}$, $s2 = 2^{1023}$	8.050	44.200	8.050	44.200	8.050	44.200	7.500	27.200
BK + (s1, s2, d, n)	n = 1		13.500	28.500	13.500	28.500	13.500	28.500	12.100	19.700
	n = 96		63.100	78.200	63.100	78.200	63.100	78.200	61.700	69.300
BK - (s1, s2, d, n)	n = 1		13.500	28.500	13.500	28.500	13.500	28.500	12.100	20.600
	n = 96		63.100	78.200	63.100	78.200	63.100	78.200	61.700	70.200
DBK + (s1, s2, d, n)	n = 1		10.100	24.200	10.100	24.200	10.100	24.200	7.050	19.200
	n = 96		59.800	73.900	59.800	73.900	59.800	73.900	59.400	68.900
DBK - (s1, s2, d, n)	n = 1		10.100	24.200	10.100	24.200	10.100	24.200	7.050	19.900
	n = 96		59.800	73.900	59.800	73.900	59.800	73.900	59.400	69.600
$ + (s, d)	—		15.400	64.300	15.400	64.300	15.400	64.300	14.400	34.000
$ + (s1, s2, d)	—		19.700	71.000	19.700	71.000	19.700	71.000	9.200	22.900
FLTD	double precision	s = 0	3.100	19.600	3.100	19.600	3.100	19.600	4.000	8.900
		$s = 7FFF_H$	3.350	19.900	3.350	19.900	3.350	19.900	3.400	9.000
DFLTD	double precision	$s = 7FFF_H$	3.200	20.400	3.200	20.400	3.200	20.400	4.100	10.800
		$s = 7FFF_H$	3.450	20.500	3.450	20.500	3.450	20.500	3.600	10.800
INTD	double precision	$s = 7FFF_H$	3.200	22.900	3.200	22.900	3.200	22.900	3.500	9.300
		$s = 7FFF_H$	4.100	34.300	4.100	34.300	4.100	34.300	5.100	19.500
DINTD	double precision	$s = 7FFF_H$	3.200	23.000	3.200	23.000	3.200	23.000	2.600	6.800
		$s = 7FFF_H$	4.050	33.500	4.050	33.500	4.050	33.500	3.400	11.700
DBL	executed		3.300	5.900	3.300	5.900	3.300	5.900	2.700	3.800
WORD	executed		3.000	7.250	3.000	7.250	3.000	7.250	2.900	7.000
GRY	executed		3.350	7.500	3.350	7.500	3.350	7.500	2.700	6.100

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)	Processing time (μs)							
		Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
DGRY	executed	3.000	7.200	3.000	7.200	3.000	7.200	2.900	4.600
GBIN	executed	4.600	9.700	4.600	9.700	4.600	9.700	4.000	8.200
DGBIN	executed	5.550	10.700	5.550	10.700	5.550	10.700	5.500	8.000
NEG	executed	3.300	6.850	3.300	6.850	3.300	6.850	2.400	4.100
DNEG	executed	3.050	5.700	3.050	5.700	3.050	5.700	2.500	4.300
ENEG	Floating point = 0	3.100	7.350	3.100	7.350	3.100	7.350	2.500	3.400
	Floating point = -1.0	3.350	11.700	3.350	11.700	3.350	11.700	2.700	4.500
EDNEG	Floating point = 0	3.000	21.200	3.000	21.200	3.000	21.200	2.200	3.500
	Floating point = -1.0	3.100	22.900	3.100	22.900	3.100	22.900	2.400	3.500
BKBCD (s, d, n)	n = 1	8.700	27.600	8.700	27.600	8.700	27.600	9.700	22.000
	n = 96	84.200	104.000	84.200	104.000	84.200	104.000	74.200	86.500
BKBIN (s, d, n)	n = 1	8.450	28.100	8.450	28.100	8.450	28.100	8.900	16.300
	n = 96	56.100	75.800	56.100	75.800	56.100	75.800	58.500	65.100
ECON	—	3.100	21.300	3.100	21.300	3.100	21.300	4.300	6.800
EDCON	—	5.050	24.000	5.050	24.000	5.050	24.000	2.800	5.400
EDMOV	—	2.900	22.900	2.900	22.900	2.900	22.900	3.200	7.800
$MOV	Character string to be transferred = 0	6.250	30.100	6.250	30.100	6.250	30.100	4.500	13.900
	Character string to be transferred = 32	15.500	39.300	15.500	39.300	15.500	39.300	15.400	17.500
BXCH (d1, d2, n)	n = 1	8.400	20.900	8.400	20.900	8.400	20.900	8.700	15.200
	n = 96	67.100	79.900	67.100	79.900	67.100	79.900	67.200	74.000
SWAP	—	3.300	3.550	3.300	3.550	3.300	3.550	2.400	2.700
GOEND	—		0.550		0.550		0.550		
DI	—	2.800	8.400	2.800	8.400	2.800	8.400	1.800	2.200
EI	—	4.300	12.300	4.300	12.300	4.300	12.300	3.100	3.800
IMASK	—	12.900	40.600	12.900	40.600	12.900	40.600	9.800	25.000
IRET	—		1.000		1.000		1.000		
RSF X n	n = 1	7.500	26.500	7.500	26.500	7.500	26.500	4.300	16.100
	n = 96	11.400	30.400	11.400	30.400	11.400	30.400	11.400	23.700
RSF Y n	n = 1	7.300	26.300	7.300	26.300	7.300	26.300	3.800	10.000
	n = 96	10.900	29.900	10.900	29.900	10.900	29.900	8.500	15.200
UDCNT1	—	1.500	7.100	1.500	7.100	1.500	7.100	1.000	2.000
UDCNT2	—	1.500	6.300	1.500	6.300	1.500	6.300	1.000	4.000
TTMR	—	5.300	20.900	5.300	20.900	5.300	20.900	3.900	6.100
STMR	—	8.900	49.800	8.900	49.800	8.900	49.800	7.200	30.000
ROTC	—	52.300	52.600	52.300	52.600	52.300	52.600	15.200	16.100
RAMP	—	7.400	30.900	7.400	30.900	7.400	30.900	5.900	18.300
SPD	—	1.500	6.300	1.500	6.300	1.500	6.300	1.000	2.800
PLSY	—	6.400	7.100	6.400	7.100	6.400	7.100	3.500	4.700
PWM	—	3.900	4.600	3.900	4.600	3.900	4.600	3.400	3.400
MTR	—	10.100	61.400	10.100	61.400	10.100	61.400	20.500	28.400
BKAND (s1, s2, d, n)	n = 1	13.600	28.500	13.600	28.500	13.600	28.500	12.100	20.100
	n = 96	63.200	78.200	63.200	78.200	63.200	78.200	57.400	63.200
BKOR (s1, s2, d, n)	n = 1	13.500	28.500	13.500	28.500	13.500	28.500	7.700	13.200
	n = 96	63.100	78.200	63.100	78.200	63.100	78.200	57.400	62.800
BKXOR (s1, s2, d, n)	n = 1	13.600	28.300	13.600	28.300	13.600	28.300	7.800	13.200
	n = 96	63.100	78.000	63.100	78.000	63.100	78.000	57.300	62.800
BKXNR (s1, s2, d, n)	n = 1	13.500	28.300	13.500	28.300	13.500	28.300	7.800	14.100
	n = 96	63.100	78.000	63.100	78.000	63.100	78.000	57.400	62.900
BSFR (d, n)	n = 1	5.050	21.100	5.050	21.100	5.050	21.100	3.700	6.300
	n = 96	9.000	34.800	9.000	34.800	9.000	34.800	10.200	12.800

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
BSFL (d, n)	n = 1		4.800	19.100	4.800	19.100	4.800	19.100	4.500	8.900
	n = 96		8.550	34.300	8.550	34.300	8.550	34.300	10.100	14.300
SFTBR (n1, n2, d)	n1 = 16 / n2 = 1		10.300	46.500	10.300	46.500	10.300	46.500	8.800	43.400
	n1 = 16 / n2 = 15		10.300	46.400	10.300	46.400	10.300	46.400	8.750	43.400
SFTBL (n1, n2, d)	n1 = 16 / n2 = 1		10.500	49.800	10.500	49.800	10.500	49.800	8.050	45.100
	n1 = 16 / n2 = 15		10.500	49.800	10.500	49.800	10.500	49.800	8.050	45.100
SFTWR (n1, n2, d)	n1 = 16 / n2 = 1		7.950	24.000	7.950	24.000	7.950	24.000	6.500	22.800
	n1 = 16 / n2 = 15		7.950	24.100	7.950	24.100	7.950	24.100	6.500	22.800
SFTWL (n1, n2, d)	n1 = 16 / n2 = 1		8.700	23.600	8.700	23.600	8.700	23.600	7.350	23.600
	n1 = 16 / n2 = 15		8.650	23.700	8.650	23.700	8.650	23.700	7.300	23.700
BSET (d, n)	n = 1		4.550	4.750	4.550	4.750	4.550	4.750	3.000	3.400
	n = 15		4.550	4.750	4.550	4.750	4.550	4.750	3.000	3.500
BRST (d, n)	n = 1		4.600	4.750	4.600	4.750	4.600	4.750	3.000	3.400
	n = 15		4.600	4.750	4.600	4.750	4.600	4.750	3.000	3.400
TEST	executed		7.250	13.200	7.250	13.200	7.250	13.200	4.400	6.900
DTEST	executed		6.950	12.900	6.950	12.900	6.950	12.900	4.500	7.000
BKRST (d, n)	n = 1		7.350	11.600	7.350	11.600	7.350	11.600	4.300	5.200
	n = 96		10.100	22.600	10.100	22.600	10.100	22.600	6.500	13.200
SER (s1, s2, d, n)	n = 1	All match	6.650	6.800	6.650	6.800	6.650	6.800	5.000	5.300
		None match	6.650	6.800	6.650	6.800	6.650	6.800	5.000	5.300
	n = 96	All match	34.000	42.300	34.000	42.300	34.000	42.300	32.300	35.900
		None match	34.000	42.300	34.000	42.300	34.000	42.300	32.400	35.900
DSER (s1, s2, d, n)	n = 1	All match	8.000	16.300	8.000	16.300	8.000	16.300	6.800	10.200
		None match	8.000	16.300	8.000	16.300	8.000	16.300	6.800	10.200
	n = 96	All match	54.100	62.600	54.100	62.600	54.100	62.600	52.800	56.300
		None match	54.100	62.600	54.100	62.600	54.100	62.600	52.800	56.300
DSUM (s, d)	s = 0		4.100	4.200	4.100	4.200	4.100	4.200	3.700	4.100
	s = FFFFFFFFH		4.100	4.200	4.100	4.200	4.100	4.200	3.800	4.100
DECO (s, d, n)	n = 2		8.850	23.000	8.850	23.000	8.850	23.000	6.000	16.400
	n = 8		13.600	36.600	13.600	36.600	13.600	36.600	8.100	15.200
ENCO (s, d, n)	n = 2	M1 = ON	7.650	11.900	7.650	11.900	7.650	11.900	5.300	6.300
		M4 = ON	7.500	11.700	7.500	11.700	7.500	11.700	5.200	6.200
	n = 8	M1 = ON	14.600	27.800	14.600	27.800	14.600	27.800	10.400	17.900
		M256 = ON	10.600	23.700	10.600	23.700	10.600	23.700	5.700	13.300
DIS (s, d, n)	n = 1		6.500	14.800	6.500	14.800	6.500	14.800	5.000	10.900
	n = 4		6.900	15.200	6.900	15.200	6.900	15.200	5.400	11.300
UNI (s, d, n)	n = 1		6.800	15.100	6.800	15.100	6.800	15.100	5.500	8.900
	n = 4		7.500	15.900	7.500	15.900	7.500	15.900	6.200	9.600
NDIS	executed		4.750	18.700	4.750	18.700	4.750	18.700	11.000	16.300
NUNI	executed		4.750	18.700	4.750	18.700	4.750	18.700	10.600	16.000
WTOB (s, d, n)	n = 1		6.600	14.900	6.600	14.900	6.600	14.900	5.000	6.500
	n = 96		37.700	46.100	37.700	46.100	37.700	46.100	36.000	38.400
BTOW (s, d, n)	n = 1		7.350	15.600	7.350	15.600	7.350	15.600	5.100	6.100
	n = 96		32.100	40.500	32.100	40.500	32.100	40.500	29.900	32.000
MAX (s, d, n)	n = 1		8.250	24.900	8.250	24.900	8.250	24.900	4.300	6.900
	n = 96		34.200	51.600	34.200	51.600	34.200	51.600	32.000	34.300
MIN (s, d, n)	n = 1		8.250	24.800	8.250	24.800	8.250	24.800	4.400	6.800
	n = 96		34.200	51.600	34.200	51.600	34.200	51.600	30.300	34.800
DMAX (s, d, n)	n = 1		6.800	34.900	6.800	34.900	6.800	34.900	4.800	14.200
	n = 96		60.300	89.200	60.300	89.200	60.300	89.200	56.400	68.000

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)	Processing time (μs)							
		Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
DMIN (s, d, n)	n = 1	7.600	35.700	7.600	35.700	7.600	35.700	4.800	9.300
	n = 96	59.400	90.000	59.400	90.000	59.400	90.000	55.400	62.800
SORT (s1, s2, d1, d2, n)	n = 1	10.100	28.900	10.100	28.900	10.100	28.900	6.200	12.200
	n = 96	52.100	92.400	52.100	92.400	52.100	92.400	6.200	13.100
DSORT (s1, s2, d1, d2, n)	n = 1	9.300	29.000	9.300	29.000	9.300	29.000	6.200	10.500
	n = 96	43.600	89.600	43.600	89.600	43.600	89.600	6.100	10.500
WSUM (s, d, n)	n = 1	6.700	15.000	6.700	15.000	6.700	15.000	4.800	6.200
	n = 96	28.900	37.100	28.900	37.100	28.900	37.100	26.900	28.700
DWSUM (s, d, n)	n = 1	8.600	26.800	8.600	26.800	8.600	26.800	5.500	7.000
	n = 96	56.200	74.700	56.200	74.700	56.200	74.700	53.000	56.300
MEAN (s, d, n)	n = 1	5.850	19.800	5.850	19.800	5.850	19.800	4.300	17.300
	n = 96	17.300	38.200	17.300	38.200	17.300	38.200	16.000	35.500
DMEAN (s, d, n)	n = 1	6.900	23.300	6.900	23.300	6.900	23.300	5.750	21.900
	n = 96	29.400	49.900	29.400	49.900	29.400	49.900	29.200	48.600
NEXT	—	1.000	1.100	1.000	1.100	1.000	1.100	0.980	1.400
BREAK	—	4.700	25.000	4.700	25.000	4.700	25.000	21.300	17.900
RET	Return to original program	4.100	19.500	4.100	19.500	4.100	19.500	2.000	3.000
	Return to other program	4.700	16.700	4.700	16.700	4.700	16.700	2.300	4.900
FCALL pn	Internal file pointer	5.400	5.400	5.400	5.400	5.400	5.400	3.300	5.300
	Common pointer	7.600	30.500	7.600	30.500	7.600	30.500	4.900	6.600
FCALL pn s1 to s5	—	50.400	62.700	50.400	62.700	50.400	62.700	19.800	23.700
ECALL * pn *: Program name	—	105.000	214.000	105.000	214.000	105.000	214.000	75.700	134.000
ECALL * pn s1 to s5 *: Program name	—	164.000	271.000	164.000	271.000	164.000	271.000	109.000	173.000
EFCALL * pn *: Program name	—	105.000	214.000	105.000	214.000	105.000	214.000	76.200	134.000
EFCALL * pn s1 to s5 *: Program name	—	164.000	271.000	164.000	271.000	164.000	271.000	90.500	170.000
XCALL	—	5.100	6.700	5.100	6.700	5.100	6.700	3.800	6.400
COM CCOM	When selecting I/O refresh only	18.100	89.100	18.100	89.100	18.100	89.100	12.800	79.000
	When selecting CC-Link refresh only (master station side)	33.300	132.000	33.300	132.000	33.300	132.000	24.900	119.000
	When selecting CC-Link refresh only (local station side)	33.300	132.000	33.300	132.000	33.300	132.000	24.900	119.000
	• When selecting MELSECNET/ H refresh only (Control station side) • When selecting CC-Link IE refresh only (Control station/ Master station side)	78.600	231.000	78.600	231.000	78.600	231.000	54.000	212.000
	• When selecting MELSECNET/ H refresh only (Normal station side) • When selecting CC-Link IE refresh only (Normal station/ Local station side)	78.600	231.000	78.600	231.000	78.600	231.000	54.000	212.000
	When selecting intelli auto refresh only	18.100	89.000	18.100	89.000	18.100	89.000	12.800	79.000
	When selecting I/O outside the group only (Input only)	15.700	71.600	15.700	71.600	15.700	71.600	8.600	76.500
	When selecting I/O outside the group only (Output only)	40.200	152.000	40.200	152.000	40.200	152.000	26.300	135.000
	When selecting I/O outside the group only (Both I/O)	45.800	153.000	45.800	153.000	45.800	153.000	26.100	135.000
	When selecting refresh of multiple CPU high speed transmission area only	—	—	—	—	—	—	—	—
	When selecting communication with peripheral device	18.200	89.000	18.200	89.000	18.200	89.000	7.250	54.300
FIFW	Number of data points = 0	6.100	14.200	6.100	14.200	6.100	14.200	3.700	10.100
	Number of data points = 96	6.100	14.200	6.100	14.200	6.100	14.200	3.800	5.200
FIFR	Number of data points = 0	7.500	15.600	7.500	15.600	7.500	15.600	4.400	5.800
	Number of data points = 96	37.000	45.000	37.000	45.000	37.000	45.000	33.500	35.200

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)	Processing time (μs)							
		Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
FPOP	Number of data points = 0	7.600	15.600	7.600	15.600	7.600	15.600	4.400	10.800
	Number of data points = 96	7.600	15.600	7.600	15.600	7.600	15.600	4.400	10.800
FINS	Number of data points = 0	6.900	15.000	6.900	15.000	6.900	15.000	5.000	10.700
	Number of data points = 96	36.600	44.700	36.600	44.700	36.600	44.700	4.400	10.900
FDEL	Number of data points = 0	8.000	16.100	8.000	16.100	8.000	16.100	4.900	11.300
	Number of data points = 96	37.300	45.500	37.300	45.500	37.300	45.500	34.200	35.900
FROM (d, n1, n2, n3)	n3 = 1	17.400	74.700	17.400	74.700	17.400	74.700	12.100	71.300
	n3 = 1000	406.000	498.500	406.000	498.500	406.000	498.500	402.600	495.100
DFRO (d, n1, n2, n3)	n3 = 1	19.600	85.600	19.600	85.600	19.600	85.600	14.600	81.800
	n3 = 500	406.000	498.500	406.000	498.500	406.000	498.500	402.600	495.100
TO (s, n1, n2, n3)	n3 = 1	16.400	69.600	16.400	69.600	16.400	69.600	11.700	63.400
	n3 = 1000	381.300	471.200	381.300	471.200	381.300	471.200	375.900	464.300
DTO (s, n1, n2, n3)	n3 = 1	18.600	85.100	18.600	85.100	18.600	85.100	14.200	78.500
	n3 = 500	381.300	471.200	381.300	471.200	381.300	471.200	375.900	464.300
LEDR	No display ==> no display	1.500	7.100	1.500	7.100	1.500	7.100	5.100	5.100
	LED instruction execution ==> no display	38.900	109.000	38.900	109.000	38.900	109.000	35.700	89.200
BINDA (s, d)	s = 1	5.600	13.900	5.600	13.900	5.600	13.900	4.900	6.500
	s = -32768	7.800	16.200	7.800	16.200	7.800	16.200	7.200	8.700
DBINDA (s, d)	s = 1	6.200	14.500	6.200	14.500	6.200	14.500	5.700	7.100
	s = -2147483648	11.000	19.200	11.000	19.200	11.000	19.200	10.400	12.200
BINHA (s, d)	s = 1	5.050	13.400	5.050	13.400	5.050	13.400	4.400	5.900
	s = FFFFH	5.050	13.400	5.050	13.400	5.050	13.400	4.400	5.800
DBINHA (s, d)	s = 1	5.600	13.900	5.600	13.900	5.600	13.900	5.200	6.700
	s = FFFFFFFFH	5.600	13.900	5.600	13.900	5.600	13.900	5.100	6.500
BCDDA (s, d)	s = 1	4.850	13.200	4.850	13.200	4.850	13.200	4.300	5.800
	s = 9999	5.300	13.600	5.300	13.600	5.300	13.600	4.700	6.100
DBCDDA (s, d)	s = 1	5.300	13.600	5.300	13.600	5.300	13.600	4.800	6.300
	s = 99999999	6.200	14.500	6.200	14.500	6.200	14.500	5.600	7.100
DABIN (s, d)	s = 1	7.000	18.500	7.000	18.500	7.000	18.500	6.500	9.000
	s = -32768	6.950	18.500	6.950	18.500	6.950	18.500	6.300	8.900
DDABIN (s, d)	s = 1	9.450	21.000	9.450	21.000	9.450	21.000	9.400	12.000
	s = -2147483648	9.450	21.000	9.450	21.000	9.450	21.000	9.100	11.600
HABIN (s, d)	s = 1	5.650	17.100	5.650	17.100	5.650	17.100	4.900	7.500
	s = FFFFH	5.750	17.300	5.750	17.300	5.750	17.300	5.100	8.100
DHABIN(s, d)	s = 1	6.800	18.200	6.800	18.200	6.800	18.200	6.000	8.500
	s = FFFFFFFFH	7.100	18.600	7.100	18.600	7.100	18.600	6.300	8.900
DABCD (s, d)	s = 1	5.650	17.200	5.650	17.200	5.650	17.200	5.000	7.500
	s = 9999	5.700	17.200	5.700	17.200	5.700	17.200	5.000	7.500
DDABCD (s, d)	s = 1	6.850	18.300	6.850	18.300	6.850	18.300	6.200	8.800
	s = 99999999	6.850	18.300	6.850	18.300	6.850	18.300	6.200	8.800
COMRD	—	185.000	188.000	185.000	188.000	185.000	188.000	97.300	97.400
LEN	1 character	4.700	16.200	4.700	16.200	4.700	16.200	4.100	6.600
	96 characters	20.600	32.900	20.600	32.900	20.600	32.900	19.800	22.400
STR	—	9.800	36.500	9.800	36.500	9.800	36.500	6.900	14.400
DSTR	—	12.100	40.400	12.100	40.400	12.100	40.400	10.200	20.800
VAL	—	12.200	40.900	12.200	40.900	12.200	40.900	9.800	23.900
DVAL	—	19.400	45.600	19.400	45.600	19.400	45.600	14.000	33.100
ESTR	—	29.700	87.800	29.700	87.800	29.700	87.800	22.100	52.400
EVAL	Decimal point format all 2-digit specification	23.900	70.400	23.900	70.400	23.900	70.400	23.300	36.500
	Exponent format all 6-digit specification	23.700	70.300	23.700	70.300	23.700	70.300	23.300	36.400

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ASC (s, d, n)	n = 1		10.200	41.800	10.200	41.800	10.200	41.800	5.600	19.700
	n = 96		31.900	66.600	31.900	66.600	31.900	66.600	30.200	44.700
HEX (s, d, n)	n = 1		8.600	43.400	8.600	43.400	8.600	43.400	7.500	23.100
	n = 96		77.100	115.000	77.100	115.000	77.100	115.000	37.500	53.300
RIGHT (s, d, n)	n = 1		10.900	29.600	10.900	29.600	10.900	29.600	7.600	11.400
	n = 96		41.400	60.300	41.400	60.300	41.400	60.300	36.300	46.000
LEFT (s, d, n)	n = 1		10.600	29.300	10.600	29.300	10.600	29.300	6.500	16.100
	n = 96		41.300	60.200	41.300	60.200	41.300	60.200	36.200	46.200
MIDR	—		11.700	30.600	11.700	30.600	11.700	30.600	9.500	19.100
MIDW	—		12.400	24.000	12.400	24.000	12.400	24.000	10.300	18.200
INSTR	No match		22.000	38.200	22.000	38.200	22.000	38.200	19.300	29.000
	Match	Head	13.300	29.600	13.300	29.600	13.300	29.600	10.300	20.000
		End	21.900	38.100	21.900	38.100	21.900	38.100	51.100	60.800
EMOD	—		11.600	24.000	11.600	24.000	11.600	24.000	10.300	15.300
EREXP	—		19.700	28.000	19.700	28.000	19.700	28.000	19.300	22.300
STRINS (s, d, n)	s = 128 / d = 40 / n = 1		47.000	102.000	47.000	102.000	47.000	102.000	44.300	96.700
	s = 128 / d = 40 / n = 48		70.100	134.000	70.100	134.000	70.100	134.000	58.800	112.000
STRDEL (s, d, n)	s = 128 / d = 40 / n = 1		46.400	93.600	46.400	93.600	46.400	93.600	39.000	78.100
	s = 128 / d = 40 / n = 48		44.500	70.600	44.500	70.600	44.500	70.600	36.000	69.200
SIN	single precision		6.400	13.900	6.400	13.900	6.400	13.900	4.500	9.900
COS	single precision		6.100	13.500	6.100	13.500	6.100	13.500	4.300	8.200
TAN	single precision		8.300	15.000	8.300	15.000	8.300	15.000	5.100	7.200
ASIN	single precision		7.300	15.600	7.300	15.600	7.300	15.600	6.100	13.700
ACOS	single precision		8.100	16.500	8.100	16.500	8.100	16.500	6.800	11.100
ATAN	single precision		5.350	12.000	5.350	12.000	5.350	12.000	4.000	6.900
SIND	double precision		13.400	51.300	13.400	51.300	13.400	51.300	9.600	26.000
COSD	double precision		14.700	51.700	14.700	51.700	14.700	51.700	10.000	26.900
TAND	double precision		17.400	54.400	17.400	54.400	17.400	54.400	11.400	25.300
ASIND	double precision		22.600	60.300	22.600	60.300	22.600	60.300	12.100	30.800
ACOSD	double precision		19.700	60.000	19.700	60.000	19.700	60.000	11.700	28.000
ATAND	double precision		15.000	51.800	15.000	51.800	15.000	51.800	9.700	22.000
RAD	single precision		3.200	10.300	3.200	10.300	3.200	10.300	2.500	4.800
RADD	double precision		5.200	43.100	5.200	43.100	5.200	43.100	4.100	16.400
DEG	single precision		3.200	11.500	3.200	11.500	3.200	11.500	2.500	4.700
DEGD	double precision		5.150	43.800	5.150	43.800	5.150	43.800	5.000	18.100
SQR	single precision		3.900	12.300	3.900	12.300	3.900	12.300	3.500	9.300
SQRD	double precision		7.000	45.700	7.000	45.700	7.000	45.700	5.700	25.400
EXP (s, d)	single precision	s = -10	6.350	13.800	6.350	13.800	6.350	13.800	4.000	13.000
		s = 1	6.350	13.800	6.350	13.800	6.350	13.800	4.000	13.000
EXPD (s, d)	double precision	s = -10	15.800	52.700	15.800	52.700	15.800	52.700	8.800	27.600
		s = 1	15.400	52.500	15.400	52.500	15.400	52.500	8.500	27.300
LOG (s, d)	single precision	s = 1	5.800	14.900	5.800	14.900	5.800	14.900	4.100	8.100
		s = 10	7.450	16.500	7.450	16.500	7.450	16.500	6.200	10.300
LOGD (s, d)	double precision	s = 1	11.000	48.900	11.000	48.900	11.000	48.900	9.500	28.300
		s = 10	12.600	51.300	12.600	51.300	12.600	51.300	11.100	29.900
RND	—		1.950	5.450	1.950	5.450	1.950	5.450	1.200	2.300
SRND	—		2.750	4.550	2.750	4.550	2.750	4.550	1.400	2.400
BSQR (s, d)	s = 0		2.500	6.800	2.500	6.800	2.500	6.800	1.800	3.300
	s = 9999		6.400	15.500	6.400	15.500	6.400	15.500	5.100	8.800

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
BDSQR (s, d)	s = 0		2.600	6.050	2.600	6.050	2.600	6.050	1.900	3.700
	s = 99999999		8.450	17.600	8.450	17.600	8.450	17.600	7.500	10.900
BSIN	—		11.500	32.800	11.500	32.800	11.500	32.800	8.700	20.200
BCOS	—		10.400	32.500	10.400	32.500	10.400	32.500	7.800	14.400
BTAN	—		12.100	33.700	12.100	33.700	12.100	33.700	9.000	17.000
BASIN	—		13.300	32.800	13.300	32.800	13.300	32.800	12.200	15.100
BACOS	—		13.400	33.700	13.400	33.700	13.400	33.700	13.100	14.900
BATAN	—		12.600	31.400	12.600	31.400	12.600	31.400	11.400	15.700
POW (s1, s2, d)	single precision	s1 = 12.3 E + 5; s2 = 3.45 E + 0	12.200	22.100	12.200	22.100	12.200	22.100	8.950	19.500
POWD (s1, s2, d)	double precision	s1 = 12.3 E + 5; s2 = 3.45 E + 0	27.300	61.000	27.300	61.000	27.300	61.000	19.400	55.200
LOG10	single precision		8.200	16.500	8.200	16.500	8.200	16.500	5.950	14.800
LOG10D	double precision		15.100	48.000	15.100	48.000	15.100	48.000	12.400	46.500
LIMIT	—		5.350	5.500	5.350	5.500	5.350	5.500	5.200	5.400
DLIMIT	—		6.000	6.150	6.000	6.150	6.000	6.150	5.700	5.900
BAND	—		5.450	12.400	5.450	12.400	5.450	12.400	5.400	6.300
DBAND	—		6.050	11.900	6.050	11.900	6.050	11.900	5.800	6.900
ZONE	—		6.250	10.700	6.250	10.700	6.250	10.700	5.200	11.100
DZONE	—		6.000	11.900	6.000	11.900	6.000	11.900	5.700	10.800
SCL (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	14.900	50.100	14.900	50.100	14.900	50.100	14.700	48.000
		Point No. 9 < s1 < Point No. 10	15.800	50.900	15.800	50.900	15.800	50.900	19.600	50.400
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	13.900	53.100	13.900	53.100	13.900	53.100	13.700	51.000
		Point No. 9 < s1 < Point No. 10	16.600	56.600	16.600	56.600	16.600	56.600	20.400	56.200
DSCL (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	13.400	52.400	13.400	52.400	13.400	52.400	12.800	50.300
		Point No. 9 < s1 < Point No. 10	14.200	54.100	14.200	54.100	14.200	54.100	17.300	53.500
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	12.300	53.200	12.300	53.200	12.300	53.200	11.500	51.100
		Point No. 9 < s1 < Point No. 10	15.000	57.600	15.000	57.600	15.000	57.600	18.100	57.100
SCL2 (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	14.200	53.300	14.200	53.300	14.200	53.300	13.200	51.200
		Point No. 9 < s1 < Point No. 10	14.900	55.000	14.900	55.000	14.900	55.000	18.000	54.500
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	15.000	53.500	15.000	53.500	15.000	53.500	14.000	51.300
		Point No. 9 < s1 < Point No. 10	16.300	56.400	16.300	56.400	16.300	56.400	19.300	55.800
DSCL2 (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	13.400	52.700	13.400	52.700	13.400	52.700	13.100	50.500
		Point No. 9 < s1 < Point No. 10	14.200	54.300	14.200	54.300	14.200	54.300	18.100	53.700
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	12.300	53.200	12.300	53.200	12.300	53.200	12.100	51.000
		Point No. 9 < s1 < Point No. 10	15.000	57.600	15.000	57.600	15.000	57.600	18.900	57.100
RSET	Standard RAM		6.800	26.900	6.800	26.900	6.800	26.900	3.000	16.400
	SRAM card		—	—	—	—	—	—	3.000	16.400
QDRSET	SRAM card to standard RAM		—	—	—	—	—	—	230.000	327.000
	Standard RAM to SRAM card		—	—	—	—	—	—	997.000	1.066.000
QCDSET	SRAM card to standard ROM		—	—	—	—	—	—	525.000	690.000
	Standard ROM to SRAM card		—	—	—	—	—	—	490.000	655.000
DATERD	—		5.600	27.800	5.600	27.800	5.600	27.800	5.100	14.700
DATEWR	—		7.800	42.100	7.800	42.100	7.800	42.100	7.100	23.000
DATE +	No digit increase		14.200	41.200	14.200	41.200	14.200	41.200	6.500	13.100
	Digit increase		14.200	41.200	14.200	41.200	14.200	41.200	5.700	21.200
DATE -	No digit increase		15.100	41.200	15.100	41.200	15.100	41.200	6.500	11.500
	Digit increase		15.100	41.200	15.100	41.200	15.100	41.200	5.700	17.200
SECOND	—		5.800	20.500	5.800	20.500	5.800	20.500	2.600	5.900
HOUR	—		6.200	22.500	6.200	22.500	6.200	22.500	3.000	5.300

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
LDDT =	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	8.200	25.500
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ANDDT=	not executed		0.480	0.320	0.240	0.160				
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ORDT=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDDT <>	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ANDDT<>	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ORDT<>	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDDT>	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ANDDT>	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	8.200	25.500	7.200	23.400
		no continuity	8.200	25.500	8.200	25.500	8.200	25.500	7.200	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ORDT>	not executed			0.480	0.320	0.240	0.160			
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	8.200	25.500	7.400	23.300
		no continuity	8.200	25.500	8.200	25.500	8.200	25.500	7.400	23.300
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDDT<=	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ANDDT<=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ORDT<=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDDT<	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ANDDT<	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ORDT<	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDDT>=	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ANDDT>=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
ORDT>=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified date	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDTM=	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
ANDTM=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ORTM=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDTM<>	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
ANDTM<>	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
ORTM<>	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDTM>	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
ANDTM>	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
ORTM>	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDTM<=	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
ANDTM<=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
ORTM<=	not executed			0.480		0.320		0.240	0.160	
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
LDTM<	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
ANDTM<	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
ORTM<	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDTM>=	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
ANDTM>=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
ORTM>=	not executed			0.480		0.320		0.240		0.160
	Comparison of specified time	continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
	Comparison of current time	continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
S.DATERD	—		9.250	51.000	9.250	51.000	9.250	51.000	7.500	23.400
S.DATE +	No digit increase		16.800	75.400	16.800	75.400	16.800	75.400	9.100	23.400
	Digit increase		16.800	75.400	16.800	75.400	16.800	75.400	8.900	22.200
S.DATE -	No digit increase		17.600	75.300	17.600	75.300	17.600	75.300	9.000	22.200
	Digit increase		16.900	75.300	16.900	75.300	16.900	75.300	9.800	22.100
PSTOP	—		82.200	199.000	82.200	199.000	82.200	199.000	61.400	84.500
POFF	—		82.600	198.000	82.600	198.000	82.600	198.000	121.000	246.000
PSCAN	—		83.600	200.000	83.600	200.000	83.600	200.000	126.000	232.000
WDT	—		2.900	12.000	2.900	12.000	2.900	12.000	1.300	3.000
DUTY	—		7.700	27.500	7.700	27.500	7.700	27.500	4.900	24.300
TIMCHK	—		5.350	24.500	5.350	24.500	5.350	24.500	7.400	23.300
ZRRDB	File register of standard RAM		4.100	4.200	4.100	4.200	4.100	4.200	2.400	2.600
	File register of SRAM card		—	—	—	—	—	—	2.500	2.800
ZRWRB	File register of standard RAM		5.400	5.500	5.400	5.500	5.400	5.500	3.100	3.300
	File register of SRAM card		—	—	—	—	—	—	3.300	3.600
ADRSET	—		2.400	6.650	2.400	6.650	2.400	6.650	4.200	4.900
ZPUSH	—		9.200	20.500	9.200	20.500	9.200	20.500	6.900	14.000
ZPOP	—		9.000	15.500	9.000	15.500	9.000	15.500	7.500	12.500

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

Instruction	Processing (Device)		Processing time (μs)							
			Q00UJCPU		Q00UCPU		Q01UJCPU		Q02UJCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
S.ZCOM	When mounting CC-Link module (master station side)		29.400	91.700	29.400	91.700	29.400	91.700	20.600	55.000
	When mounting CC-Link module (local station side)		29.500	91.600	29.500	91.600	29.500	91.600	20.600	66.100
	When MELSECNET/H module or CC-Link IE module is mounted (Control station/Master station side)		79.900	214.000	79.900	214.000	79.900	214.000	102.000	180.000
	When MELSECNET/H module or CC-Link IE module is mounted (Normal station/Local station side)		79.900	214.000	79.900	214.000	79.900	214.000	55.600	168.100
S.RTREAD	—		9.200	57.700	9.200	57.700	9.200	57.700	6.700	33.500
S.RTWRITE	—		10.900	67.100	10.900	67.100	10.900	67.100	8.300	26.000
UNIRD (n1, d, n2)	n2 = 1		6.000	33.100	6.000	33.100	6.000	33.100	4.000	29.100
	n2 = 16		16.500	43.600	16.500	43.600	16.500	43.600	12.500	37.600
TYPERD			48.50	141.30	43.50	139.90	43.40	139.80	32.40	134.20
TRACE	Start		174.000	174.000	174.000	174.000	174.000	174.000	96.600	103.000
TRACER	—		5.100	15.500	5.100	15.500	5.100	15.500	3.800	13.600
RBMOV (s, d, n)	When standard RAM is used	1 point	—	—	12.200	34.900	12.200	34.900	9.400	31.300
		1000 points	—	—	121.500	145.100	121.500	145.100	118.500	141.300
	When SRAM card is used	1 point	—	—	—	—	—	—	9.400	31.400
		1000 points	—	—	—	—	—	—	178.500	201.300
SP.FWRITE	—		—	—	—	—	—	—	9.200	12.100
SP.FREAD	—		—	—	—	—	—	—	489.000	544.000
SP.DEVST	—		—	—	—	—	—	—	87.000	144.000
S.DEVLD	—		—	—	—	—	—	—	127.000	140.000
S.TO (n1, n2, n3, n4, d)	Writing to host CPU shared memory	n4 = 1	64.600	78.100	64.600	78.100	64.600	78.100	64.600	78.100
		n4 = 320	115.000	126.000	115.000	126.000	115.000	126.000	154.000	126.000
TO (n1, n2, s, n3)	Writing to host CPU shared memory	n3 = 1	12.700	62.200	12.700	62.200	12.700	62.200	8.300	58.200
		n3 = 320	63.500	112.300	63.500	112.300	63.500	112.300	56.200	107.800
DTO (n1, n2, s, n3)	Writing to host CPU shared memory	n3 = 1	13.500	62.300	13.500	62.300	13.500	62.300	8.600	58.300
		n3 = 320	112.900	160.800	112.900	160.800	112.900	160.800	106.800	157.300
FROM (n1, n2, d, n3)	Reading from host CPU shared memory	n3 = 1	12.100	58.700	12.100	58.700	12.100	58.700	8.400	52.600
		n3 = 320	56.000	101.700	56.000	101.700	56.000	101.700	51.700	96.600
	Reading from other CPU shared memory	n3 = 1	24.400	82.900	24.400	82.900	24.400	82.900	16.600	37.000
		n3 = 320	152.000	243.000	152.000	243.000	152.000	243.000	153.000	185.000
		n3 = 1000	418.000	518.000	418.000	518.000	418.000	518.000	432.000	485.000
DFRO (n1, n2, d, n3)	Reading from host CPU shared memory	n3 = 1	12.100	58.700	12.100	58.700	12.100	58.700	8.800	53.400
		n3 = 320	97.400	143.700	97.400	143.700	97.400	143.700	94.900	139.600
	Reading from other CPU shared memory	n3 = 1	24.800	94.200	24.800	94.200	24.800	94.200	16.600	47.300
		n3 = 320	276.000	367.000	276.000	367.000	276.000	367.000	278.000	339.000
		n3 = 1000	799.000	892.000	799.000	892.000	799.000	892.000	841.000	892.000

Tab. A-22: *Processing times for instructions other than subset instructions for Universal model QCPU (1)*

NOTE *For the instructions for which a leading edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.*

Example: WORDP instruction, TOP instruction etc.

- Q03UD(E)JCPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, and Q100UDEHCPU

Instruction	Processing (Device)			Processing time (μs)							
				Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ANB ORB MPS MRD MPP	—				0.020		0.0095		0.0095		0.0095
INV	not executed				0.020		0.0095		0.0095		0.0095
	executed										
MEP MEF	not executed				0.020		0.0095		0.0095		0.0095
	executed										
EGP EGF	not executed				0.020		0.0095		0.0095		0.0095
	executed										
PLS	—			1.300	1.600	0.890	1.100	0.890	1.100	0.890	1.100
PLF	—			1.500	1.600	0.940	1.200	0.940	1.200	0.940	1.200
FF	not executed				0.040		0.0185		0.0185		0.0185
	executed			1.200	1.500	0.790	0.910	0.790	0.910	0.790	0.910
DELTA	not executed				0.040		0.0185		0.0185		0.0185
	executed			2.800	3.600	2.400	3.200	2.400	3.200	2.400	3.200
SFT	not executed				0.040		0.0185		0.0185		0.0185
	executed			1.600	3.300	1.100	2.700	1.100	2.700	1.100	2.700
MC	—				0.040		0.0185		0.0185		0.0185
MCR	—				0.040		0.0185		0.0185		0.0185
FEND END	error check performed			108.000	130.000	75.800	89.300	75.800	89.300	75.800	89.300
	no error check performed			107.000	124.000	75.800	89.800	75.800	89.800	75.800	89.800
NOP NOPLF PAGE	—				0.020		0.0095		0.0095		0.0095
LDE=	single precision	continuity		3.700	4.700	3.300	4.300	3.300	4.300	3.300	4.300
		no continuity		3.800	5.000	3.400	4.500	3.400	4.500	3.400	4.500
ANDE=	single precision	not executed									
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.300	5.800	3.000	5.100	3.000	5.100	3.000	5.100
ORE=	single precision	not executed		3.500	5.600	3.000	5.200	3.000	5.200	3.000	5.200
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.600	4.500	3.200	4.200				
LDE< >	single precision	continuity		3.500	4.800	3.200	4.300				
		no continuity		4.000	4.700	3.600	4.200		0.0285		0.0285
ANDE< >	single precision	not executed		3.900	4.500	3.500	4.000				
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.300	5.100	3.000	4.800				
ORE< >	single precision	not executednot executed		3.500	5.000	3.100	4.600				
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.600	6.000	3.300	5.500				
LDE>	single precision	continuity		3.500	5.800	3.100	5.300				
		no continuity		3.800	5.000	3.300	4.600		0.0285		0.0285
ANDE>	single precision	not executed		3.700	4.900	3.300	4.400				
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.500	4.700	3.100	4.200				

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)			Processing time (μs)							
				Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ORE>	single precision	not executed		3.600	4.500	3.100	4.000				
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.600	5.100	3.300	4.600				
LDE<=	single precision	continuity		3.500	4.800	3.200	4.500				
		no continuity		3.800	5.600	3.400	5.200		0.0285		0.0285
ANDE<=	single precision	not executed		3.800	5.600	3.400	5.100				
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.200	4.600	2.800	4.200				
ORE<=	single precision	not executed		3.500	5.000	3.100	4.500				
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.700	5.800	3.400	5.400				
LDE<	single precision	continuity		3.800	5.700	3.300	5.300				
		no continuity		4.000	5.400	3.500	4.900		0.0285		0.0285
ANDE<	single precision	not executed		4.000	5.200	3.500	4.900				
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.400	4.600	3.000	4.200				
ORE<	single precision	not executed		3.500	4.900	3.100	4.400				
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.600	5.200	3.300	4.900				
LDE>=	single precision	continuity		3.800	6.000	3.300	5.500		0.0285		0.0285
		no continuity		3.800	5.900	3.400	5.400				
ANDE>=	single precision	not executed			0.060	0.0285	0.0285		0.0285		0.0285
		executed	continuity	3.200	4.800	2.900	4.600				
			no continuity	3.500	5.400	3.100	5.100				
ORE>=	single precision	not executed			0.060	0.0285	0.0285		0.0285		0.0285
		executed	continuity	3.600	5.200	3.300	4.700				
			no continuity	3.500	5.200	3.200	4.700				
LDED=	double precision	continuity		4.100	7.700	3.500	7.200	3.500	7.200	3.500	7.200
		no continuity		4.300	8.100	3.800	7.400	3.800	7.400	3.800	7.400
ANDED=	double precision	not executed									
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.600	7.600	3.200	7.000	3.200	7.000	3.200	7.000
ORED=	double precision	not executed		3.900	7.700	3.400	7.400	3.400	7.400	3.400	7.400
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.800	8.800	3.400	8.300	3.400	8.300	3.400	8.300
LDED<>	double precision	continuity		4.000	9.300	3.700	8.800	3.700	8.800	3.700	8.800
		no continuity		4.400	8.200	3.900	7.700	3.900	7.700	3.900	7.700
ANDED<>	double precision	not executed		4.100	7.900	3.500	7.500	3.500	7.500	3.500	7.500
		executed	continuity								
			no continuity		0.060		0.0285		0.0285		0.0285
ORED<>	double precision	not executed		3.800	7.600	3.300	7.200	3.300	7.200	3.300	7.200
		executed	continuity	3.800	7.700	3.400	7.300	3.400	7.300	3.400	7.300
			no continuity		0.060		0.0285		0.0285		0.0285
LDED>	double precision	continuity		4.100	9.300	3.700	8.900	3.700	8.900	3.700	8.900
		no continuity		3.800	8.900	3.400	8.400	3.400	8.400	3.400	8.400
ANDED>	double precision	not executed		4.300	8.100	3.800	7.500	3.800	7.500	3.800	7.500
		executed	continuity	4.100	7.800	3.500	7.200	3.500	7.200	3.500	7.200
			no continuity								

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)			Processing time (μs)							
				Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ORED>	double precision	not executed			0.060		0.0285		0.0285		0.0285
		executed	continuity	3.800	7.700	3.300	7.300	3.300	7.300	3.300	7.300
			no continuity	4.000	7.900	3.500	7.500	3.500	7.500	3.500	7.500
LDED<=	double precision	continuity			0.060		0.0285		0.0285		0.0285
		no continuity		4.100	9.300	3.700	8.800	3.700	8.800	3.700	8.800
ANDED<=	double precision	not executed		4.100	9.300	3.700	8.800	3.700	8.800	3.700	8.800
		executed	continuity	4.000	8.000	3.500	7.400	3.500	7.400	3.500	7.400
			no continuity	4.100	9.400	3.600	8.800	3.600	8.800	3.600	8.800
ORED<=	double precision	not executed									
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.800	7.700	3.300	7.200	3.300	7.200	3.300	7.200
LDED<	double precision	continuity		4.300	8.300	3.800	7.600	3.800	7.600	3.800	7.600
		no continuity		3.700	7.900	3.500	7.400	3.500	7.400	3.500	7.400
ANDED<	double precision	not executed									
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	3.800	7.800	3.300	7.300	3.300	7.300	3.300	7.300
ORED<	double precision	not executed		3.900	7.900	3.400	3.900	3.400	3.900	3.400	3.900
		executed	continuity		0.060		0.0285		0.0285		0.0285
			no continuity	4.100	9.600	3.700	9.200	3.700	9.200	3.700	9.200
LDED>=	double precision	continuity		4.000	9.600	3.700	9.200	3.700	9.200	3.700	9.200
		no continuity		4.100	9.600	3.600	9.000	3.600	9.000	3.600	9.000
ANDED>=	double precision	not executed		4.100	9.600	3.600	8.900	3.600	8.900	3.600	8.900
		executed	continuity								
			no continuity		0.060		0.0285		0.0285		0.0285
ORED>=	double precision	not executed		3.800	7.900	3.400	7.400	3.400	7.400	3.400	7.400
		executed	continuity	3.900	8.100	3.400	7.500	3.400	7.500	3.400	7.500
			no continuity		0.060		0.0285		0.0285		0.0285
LD$=	continuity			4.100	9.600	3.700	9.200	3.700	9.200	3.700	9.200
	no continuity			4.000	7.200	3.600	6.600	3.600	6.600	3.600	6.600
AND$=	not executed			5.300	8.900	4.700	8.100	4.700	8.100	4.700	8.100
	executed	continuity		4.700	9.000	4.200	8.200	4.200	8.200	4.200	8.200
		no continuity			0.060		0.0285		0.0285		0.0285
OR$=	not executed			4.400	6.800	3.900	6.400	3.900	6.400	3.900	6.400
	executed	continuity		4.500	6.700	4.000	6.300	4.000	6.300	4.000	6.300
		no continuity			0.060		0.0285		0.0285		0.0285
LD$< >	continuity			5.100	8.200	4.200	7.600	4.200	7.600	4.200	7.600
	no continuity			5.000	8.100	4.000	7.200	4.000	7.200	4.000	7.200
AND$< >	not executed			4.800	8.100	4.300	7.500	4.300	7.500	4.300	7.500
	executed	continuity		4.700	8.400	4.200	7.800	4.200	7.800	4.200	7.800
		no continuity			0.060		0.0285		0.0285		0.0285
OR$< >	not executed			4.300	5.500	4.100	5.100	4.100	5.100	4.100	5.100
	executed	continuity		4.500	5.900	4.400	5.400	4.400	5.400	4.400	5.400
		no continuity			0.060		0.0285		0.0285		0.0285
LD$>	continuity			5.200	7.300	4.100	6.700	4.100	6.700	4.100	6.700
	no continuity			5.100	7.200	4.100	6.700	4.100	6.700	4.100	6.700
AND$>	not executed			4.800	7.200	4.300	6.700	4.300	6.700	4.300	6.700
	executed	continuity		4.800	7.700	4.200	7.100	4.200	7.100	4.200	7.100
		no continuity			0.060		0.0285		0.0285		0.0285

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
OR$>	not executed		4.500	7.100	4.000	6.700	4.000	6.700	4.000	6.700
	executed	continuity	4.600	7.600	4.300	7.000	4.300	7.000	4.300	7.000
		no continuity		0.060		0.0285		0.0285		0.0285
LD$<=	continuity		5.100	6.800	4.300	6.200	4.300	6.200	4.300	6.200
	no continuity		5.200	7.200	4.300	6.600	4.300	6.600	4.300	6.600
AND$<=	not executed		5.000	6.300	4.400	5.700	4.400	5.700	4.400	5.700
	executed	continuity	4.800	6.400	4.200	5.800	4.200	5.800	4.200	5.800
		no continuity		0.060		0.0285		0.0285		0.0285
OR$<=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity	4.700	7.700	4.400	7.200	4.400	7.200	4.400	7.200
		no continuity	4.600	7.600	4.400	7.100	4.400	7.100	4.400	7.100
LD$<	continuity		4.800	8.100	4.500	7.500	4.500	7.500	4.500	7.500
	no continuity		5.000	8.300	4.500	7.900	4.500	7.900	4.500	7.900
AND$<	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity	4.500	7.100	4.000	6.600	4.000	6.600	4.000	6.600
		no continuity	4.900	7.500	4.400	7.100	4.400	7.100	4.400	7.100
OR$<	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity	5.100	7.800	4.100	7.200	4.100	7.200	4.100	7.200
		no continuity	5.000	8.100	4.100	7.600	4.100	7.600	4.100	7.600
LD$>=	continuity		4.800	6.700	4.500	6.200	4.500	6.200	4.500	6.200
	no continuity		5.000	6.700	4.400	6.300	4.400	6.300	4.400	6.300
AND$>=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity	4.400	6.800	4.100	6.300	4.100	6.300	4.100	6.300
		no continuity	4.500	7.000	4.200	6.600	4.200	6.600	4.200	6.600
OR$>=	not executed			0.060		0.0285		0.0285		0.0285
	executed	continuity	5.400	6.600	4.100	5.800	4.100	5.800	4.100	5.800
		no continuity	5.300	6.300	4.100	5.700	4.100	5.700	4.100	5.700
BKCMP = (s1, s2, d, n)	n = 1		8.200	10.700	7.500	10.000	7.500	10.000	7.500	10.000
	n = 96		57.400	61.800	46.400	48.700	46.400	48.700	46.400	48.700
BKCMP<> (s1, s2, d, n)	n = 1		8.200	10.700	7.500	10.000	7.500	10.000	7.500	10.000
	n = 96		59.500	63.300	45.600	50.400	45.600	50.400	45.600	50.400
BKCMP> (s1, s2, d, n)	n = 1		8.200	10.800	7.500	10.100	7.500	10.100	7.500	10.100
	n = 96		59.500	63.400	47.700	50.500	47.700	50.500	47.700	50.500
BKCMP<= (s1, s2, d, n)	n = 1		8.200	10.600	7.500	10.000	7.500	10.000	7.500	10.000
	n = 96		57.400	61.700	46.400	49.000	46.400	49.000	46.400	49.000
BKCMP< (s1, s2, d, n)	n = 1		8.300	10.600	7.500	10.000	7.500	10.000	7.500	10.000
	n = 96		59.500	63.600	47.600	50.500	47.600	50.500	47.600	50.500
BKCMP>= (s1, s2, d, n)	n = 1		8.200	10.900	7.500	10.000	7.500	10.000	7.500	10.000
	n = 96		57.400	62.000	46.400	48.900	46.400	48.900	46.400	48.900
DBKCMP = (s1, s2, d, n)	n = 1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
	n = 96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
DBKCMP<> (s1, s2, d, n)	n = 1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
	n = 96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
DBKCMP> (s1, s2, d, n)	n = 1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
	n = 96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
DBKCMP<= (s1, s2, d, n)	n = 1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
	n = 96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
DBKCMP< (s1, s2, d, n)	n = 1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
	n = 96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
DBKCMP>= (s1, s2, d, n)	n = 1		9.250	14.000	8.600	13.000	8.600	13.000	8.600	13.000
	n = 96		60.700	67.500	47.900	52.800	47.900	52.800	47.900	52.800
DB + (s, d)	executed		4.900	7.000	4.600	6.400	4.600	6.400	4.600	6.400
DB + (s1, s2, d)	executed		5.200	7.300	4.800	6.700	4.800	6.700	4.800	6.700
DB - (s, d)	executed		4.900	6.600	4.700	6.000	4.700	6.000	4.700	6.000
DB - (s1, s2, d)	executed		5.200	7.500	4.800	6.600	4.800	6.600	4.800	6.600
DB * (s1, s2, d)	executed		8.300	12.100	8.100	11.600	8.100	11.600	8.100	11.600
DB/ (s1, s2, d)	executed		6.100	9.100	5.800	8.800	5.800	8.800	5.800	8.800
ED + (s, d)	double precision	s = 0, d = 0	4.800	8.000	4.300	7.200	4.300	7.200	4.300	7.200
		$s = 2^{1023}$, $d = 2^{1023}$	4.800	8.000	4.300	7.200	4.300	7.200	4.300	7.200
ED + (s1, s2, d)	double precision	s1 = 0, s2 = 0	5.500	9.800	4.800	9.200	4.800	9.200	4.800	9.200
		$s1 = 2^{1023}$, $s2 = 2^{1023}$	5.500	9.800	4.800	9.200	4.800	9.200	4.800	9.200
ED - (s, d)	double precision	s = 0, d = 0	5.000	8.200	4.400	7.500	4.400	7.500	4.400	7.500
		$s = 2^{1023}$, $d = 2^{1023}$	5.000	8.200	4.400	7.500	4.400	7.500	4.400	7.500
ED - (s1, s2, d)	double precision	s1 = 0, s2 = 0	4.400	8.100	3.800	7.500	3.800	7.500	3.800	7.500
		$s1 = 2^{1023}$, $s2 = 2^{1023}$	4.400	8.100	3.800	7.500	3.800	7.500	3.800	7.500
ED * (s1, s2, d)	double precision	s1 = 0, s2 = 0	5.800	9.500	5.100	8.800	5.100	8.800	5.100	8.800
		$s1 = 2^{1023}$, $s2 = 2^{1023}$	5.800	9.500	5.100	8.800	5.100	8.800	5.100	8.800
ED / (s1, s2, d)	double precision	$s1 = 2^{1023}$, $s2 = 2^{1023}$	6.600	10.600	5.900	10.000	5.900	10.000	5.900	10.000
BK + (s1, s2, d, n)	n = 1		9.100	11.200	8.500	10.600	8.500	10.600	8.500	10.600
	n = 96		60.700	62.900	44.600	47.000	44.600	47.000	44.600	47.000
BK - (s1, s2, d, n)	n = 1		9.700	12.000	8.900	11.300	8.900	11.300	8.900	11.300
	n = 96		61.300	63.600	45.600	47.900	45.600	47.900	45.600	47.900
DBK + (s1, s2, d, n)	n = 1		7.000	10.700	6.450	9.950	6.450	9.950	6.450	9.950
	n = 96		59.400	63.100	43.700	47.500	43.700	47.500	43.700	47.500
DBK - (s1, s2, d, n)	n = 1		7.000	10.700	6.450	9.950	6.450	9.950	6.450	9.950
	n = 96		59.400	63.100	43.700	47.500	43.700	47.500	43.700	47.500
$ + (s, d)	—		8.800	14.600	8.100	13.900	8.100	13.900	8.100	13.900
$ + (s1, s2, d)	—		7.300	11.100	6.500	10.300	6.500	10.300	6.500	10.300
FLTD	double precision	s = 0	2.300	5.000	1.800	4.700	1.800	4.700	1.800	4.700
		$s = 7FFF_H$	2.500	5.200	2.200	4.800	2.200	4.800	2.200	4.800
DFLTD	double precision	s = 0	2.400	5.200	2.000	4.900	2.000	4.900	2.000	4.900
		$s = 7FFFFFFF_H$	2.700	5.400	2.300	5.100	2.300	5.100	2.300	5.100
INTD	double precision	s = 0	2.700	4.100	2.200	4.100	2.200	4.100	2.200	4.100
		s = 32766.5	3.700	5.900	3.200	5.600	3.200	5.600	3.200	5.600
DINTD	double precision	s = 0	2.600	3.900	2.200	3.400	2.200	3.400	2.200	3.400
		s = 1234567890.3	3.400	5.600	3.000	5.100	3.000	5.100	3.000	5.100
DBL	executed		2.700	3.400	2.300	2.700	2.300	2.700	2.300	2.700
WORD	executed		2.900	4.300	2.600	3.600	2.600	3.600	2.600	3.600
GRY	executed		2.700	3.900	2.300	3.400	2.300	3.400	2.300	3.400
DGRY	executed		2.900	3.500	2.500	3.000	2.500	3.000	2.500	3.000
GBIN	executed		4.000	4.800	3.800	4.300	3.800	4.300	3.800	4.300
DGBIN	executed		5.500	6.100	5.000	5.900	5.000	5.900	5.000	5.900
NEG	executed		2.400	3.900	2.000	3.300	2.000	3.300	2.000	3.300
DNEG	executed		2.500	3.700	2.500	3.300	2.500	3.300	2.500	3.300
ENEG	Floating point = 0		2.500	3.300	2.300	2.800	2.300	2.800	2.300	2.800
	Floating point = -1.0		2.700	4.500	2.500	3.900	2.500	3.900	2.500	3.900
EDNEG	Floating point = 0		2.200	3.500	1.800	3.100	1.800	3.100	1.800	3.100
	Floating point = -1.0		2.400	3.500	1.900	3.000	1.900	3.000	1.900	3.000

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)	Processing time (μs)							
		Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
BKBCD (s, d, n)	n = 1	6.600	8.900	5.900	8.200	5.900	8.200	5.900	8.200
	n = 96	71.300	74.100	61.000	63.400	61.000	63.400	61.000	63.400
BKBIN (s, d, n)	n = 1	6.500	9.800	5.600	9.300	5.600	9.300	5.600	9.300
	n = 96	56.300	59.500	49.200	52.500	49.200	52.500	49.200	52.500
ECON	—	2.600	5.400	2.100	4.500	2.100	4.500	2.100	4.500
EDCON	—	2.800	5.400	2.500	5.400	2.500	5.400	2.500	5.400
EDMOV	—	2.300	5.500	1.700	5.000	1.700	5.000	1.700	5.000
$MOV	Character string to be transferred = 0	4.000	6.300	3.400	5.600	3.400	5.600	3.400	5.600
	Character string to be transferred = 32	14.600	16.500	11.400	13.300	11.400	13.300	11.400	13.300
BXCH (d1, d2, n)	n = 1	6.200	7.900	5.500	7.300	5.500	7.300	5.500	7.300
	n = 96	67.000	68.800	47.300	49.300	47.300	49.300	47.300	49.300
SWAP	—	2.400	2.700	1.900	2.200	1.900	2.200	1.900	2.200
GOEND	—		0.500		0.500		0.500		0.500
DI	—	1.800	2.200	1.500	1.800	1.500	1.800	1.500	1.800
EI	—	3.100	3.800	3.000	3.300	3.000	3.300	3.000	3.300
IMASK	—	9.800	13.300	7.200	10.500	7.200	10.500	7.200	10.500
IRET	—		1.000		1.000		1.000		1.000
RSF X n	n = 1	4.200	5.900	3.700	5.600	3.700	5.600	3.700	5.600
	n = 96	11.400	13.800	10.700	12.400	10.700	12.400	10.700	12.400
RSF Y n	n = 1	3.800	4.800	3.400	4.800	3.400	4.800	3.400	4.800
	n = 96	8.500	9.500	8.100	8.900	8.100	8.900	8.100	8.900
UDCNT1	—	0.900	1.500	0.500	0.983	0.500	0.983	0.500	0.983
UDCNT2	—	0.900	1.700	0.600	1.300	0.600	1.300	0.600	1.300
TTMR	—	3.900	6.100	3.400	5.400	3.400	5.400	3.400	5.400
STMR	—	6.800	13.500	5.800	12.500	5.800	12.500	5.800	12.500
ROTC	—	9.000	10.500	8.000	9.400	8.000	9.400	8.000	9.400
RAMP	—	5.900	8.800	5.200	8.400	5.200	8.400	5.200	8.400
SPD	—	0.900	1.900	0.500	1.400	0.500	1.400	0.500	1.400
PLSY	—	1.900	2.200	1.500	1.800	1.500	1.800	1.500	1.800
PWM	—	1.200	1.600	0.900	1.200	0.900	1.200	0.900	1.200
MTR	—	10.400	19.800	9.400	10.000	9.400	10.000	9.400	10.000
BKAND (s1, s2, d, n)	n = 1	9.000	11.700	8.300	11.000	8.300	11.000	8.300	11.000
	n = 96	57.400	63.100	43.800	47.300	43.800	47.300	43.800	47.300
BKOR (s1, s2, d, n)	n = 1	7.700	10.000	7.700	9.500	7.700	9.500	7.700	9.500
	n = 96	57.400	61.900	44.300	45.800	44.300	45.800	44.300	45.800
BKXOR (s1, s2, d, n)	n = 1	7.800	10.100	7.300	9.200	7.300	9.200	7.300	9.200
	n = 96	57.300	61.500	43.800	45.800	43.800	45.800	43.800	45.800
BKXNR (s1, s2, d, n)	n = 1	7.800	9.600	7.600	8.900	7.600	8.900	7.600	8.900
	n = 96	57.400	61.400	43.900	45.300	43.900	45.300	43.900	45.300
BSFR (d, n)	n = 1	3.700	5.400	3.200	4.800	3.200	4.800	3.200	4.800
	n = 96	6.900	9.000	5.800	7.700	5.800	7.700	5.800	7.700
BSFL (d, n)	n = 1	4.100	5.900	3.400	5.100	3.400	5.100	3.400	5.100
	n = 96	7.100	9.100	6.000	7.900	6.000	7.900	6.000	7.900
SFTBR (d, n1, n2)	n1 = 16 / n2 = 1	7.950	17.500	7.600	16.900	7.600	16.900	7.600	16.900
	n1 = 16 / n2 = 15	7.950	17.500	7.550	16.900	7.550	16.900	7.550	16.900
SFTBL (d, n1, n2)	n1 = 16 / n2 = 1	7.950	17.900	7.500	17.400	7.500	17.400	7.500	17.400
	n1 = 16 / n2 = 15	7.900	17.800	7.500	17.300	7.500	17.300	7.500	17.300
SFTWR (d, n1, n2)	n1 = 16 / n2 = 1	5.950	10.600	4.600	8.700	4.600	8.700	4.600	8.700
	n1 = 16 / n2 = 15	5.900	10.600	4.600	8.700	4.600	8.700	4.600	8.700

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
SFTWL (d, n1, n2)	n1 = 16 / n2 = 1		5.950	10.700	4.550	8.700	4.550	8.700	4.550	8.700
	n1 = 16 / n2 = 15		5.950	10.700	4.600	8.800	4.600	8.800	4.600	8.800
BSET (d, n)	n = 1		3.000	3.400	2.500	2.800	2.500	2.800	2.500	2.800
	n = 15		3.000	3.500	2.500	2.800	2.500	2.800	2.500	2.800
BRST (d, n)	n = 1		3.000	3.400	2.600	2.800	2.600	2.800	2.600	2.800
	n = 15		3.000	3.400	2.500	2.800	2.500	2.800	2.500	2.800
TEST	executed		4.400	5.300	3.700	4.700	3.700	4.700	3.700	4.700
DTEST	executed		4.500	5.400	3.900	4.800	3.900	4.800	3.900	4.800
BKRST (d, n)	n = 1		4.300	4.600	3.700	4.100	3.700	4.100	3.700	4.100
	n = 96		6.000	6.800	5.100	6.000	5.100	6.000	5.100	6.000
SER (s1, s2, d, n)	n = 1	All match	4.900	5.300	4.200	4.600	4.200	4.600	4.200	4.600
		None match	5.000	5.300	4.200	4.600	4.200	4.600	4.200	4.600
	n = 96	All match	32.300	32.900	25.900	26.300	25.900	26.300	25.900	26.300
		None match	32.400	32.900	25.900	26.300	25.900	26.300	25.900	26.300
DSER (s1, s2, d, n)	n = 1	All match	6.100	6.500	5.400	5.700	5.400	5.700	5.400	5.700
		None match	6.200	6.600	5.500	5.900	5.500	5.900	5.500	5.900
	n = 96	All match	52.800	54.200	41.200	41.800	41.200	41.800	41.200	41.800
		None match	52.800	54.200	41.200	41.800	41.200	41.800	41.200	41.800
DSUM (s, d)	s = 0		3.700	4.100	3.300	3.600	3.300	3.600	3.300	3.600
	s = FFFFFFFFH		3.800	4.100	3.200	3.700	3.200	3.700	3.200	3.700
DECO (s, d, n)	n = 2		6.000	7.500	5.300	6.900	5.300	6.900	5.300	6.900
	n = 8		8.100	9.300	6.800	7.800	6.800	7.800	6.800	7.800
ENCO (s, d, n)	n = 2	M1 = ON	5.300	5.700	4.700	5.100	4.700	5.100	4.700	5.100
		M4 = ON	5.200	5.700	4.600	5.000	4.600	5.000	4.600	5.000
	n = 8	M1 = ON	10.400	11.400	9.000	10.000	9.000	10.000	9.000	10.000
		M256 = ON	5.700	6.800	5.100	6.100	5.100	6.100	5.100	6.100
DIS (s, d, n)	n = 1		4.400	5.300	3.800	4.600	3.800	4.600	3.800	4.600
	n = 4		4.800	5.700	4.000	5.000	4.000	5.000	4.000	5.000
UNI (s, d, n)	n = 1		5.000	5.300	3.500	4.800	3.500	4.800	3.500	4.800
	n = 4		5.600	6.000	4.000	5.100	4.000	5.100	4.000	5.100
NDIS	executed		11.000	13.100	11.000	13.200	11.000	13.200	11.000	13.200
NUNI	executed		10.600	12.700	7.300	13.200	7.300	13.200	7.300	13.200
WTOB (s, d, n)	n = 1		5.000	6.500	4.400	5.800	4.400	5.800	4.400	5.800
	n = 96		36.000	38.400	28.200	29.300	28.200	29.300	28.200	29.300
BTOW (s, d, n)	n = 1		5.100	6.100	4.600	5.500	4.600	5.500	4.600	5.500
	n = 96		29.900	32.000	22.800	23.800	22.800	23.800	22.800	23.800
MAX (s, d, n)	n = 1		4.300	6.900	4.000	6.100	4.000	6.100	4.000	6.100
	n = 96		31.200	33.500	24.700	27.000	24.700	27.000	24.700	27.000
MIN (s, d, n)	n = 1		4.400	6.800	4.000	6.000	4.000	6.000	4.000	6.000
	n = 96		30.300	34.800	26.500	28.300	26.500	28.300	26.500	28.300
DMAX (s, d, n)	n = 1		4.800	9.100	4.800	8.100	4.800	8.100	4.800	8.100
	n = 96		56.400	62.200	47.100	49.600	47.100	49.600	47.100	49.600
DMIN (s, d, n)	n = 1		4.800	6.800	4.300	5.900	4.300	5.900	4.300	5.900
	n = 96		55.400	60.200	45.400	47.400	45.400	47.400	45.400	47.400
SORT (s1, n, s2, d1, d2)	n = 1		6.200	9.300	5.600	8.800	5.600	8.800	5.600	8.800
	n = 96		6.200	9.400	5.600	8.600	5.600	8.600	5.600	8.600
DSORT (s1, n, s2, d1, d2)	n = 1		6.200	9.300	5.600	8.200	5.600	8.200	5.600	8.200
	n = 96		6.100	9.100	5.600	8.400	5.600	8.400	5.600	8.400
WSUM (s, d, n)	n = 1		4.800	6.200	4.200	5.500	4.200	5.500	4.200	5.500
	n = 96		26.900	28.700	21.300	22.300	21.300	22.300	21.300	22.300

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)	Processing time (μs)							
		Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
DWSUM (s, d, n)	n = 1	5.500	7.000	4.800	6.100	4.800	6.100	4.800	6.100
	n = 96	53.000	56.300	42.700	44.000	42.700	44.000	42.700	44.000
MEAN (s, d, n)	n = 1	4.300	8.650	3.900	7.800	3.900	7.800	3.900	7.800
	n = 96	16.000	21.400	12.900	18.000	12.900	18.000	12.900	18.000
DMEAN (s, d, n)	n = 1	5.700	10.600	5.300	9.950	5.300	9.950	5.300	9.950
	n = 96	29.200	35.200	23.000	28.800	23.000	28.800	23.000	28.800
NEXT	—	0.940	1.400	0.770	1.200	0.770	1.200	0.770	1.200
BREAK	—	10.400	5.500	9.100	5.000	9.100	5.000	9.100	5.000
RET	Return to original program	2.000	3.000	1.600	2.600	1.600	2.600	1.600	2.600
	Return to other program	2.300	3.700	2.000	3.100	2.000	3.100	2.000	3.100
FCALL pn	Internal file pointer	3.100	4.400	2.700	3.600	2.700	3.600	2.700	3.600
	Common pointer	4.000	5.700	3.600	5.100	3.600	5.100	3.600	5.100
FCALL pn s1 to s5	—	19.300	21.500	16.500	18.600	16.500	18.600	16.500	18.600
ECALL * pn *: Program name	—	70.300	82.300	65.900	77.600	65.900	77.600	65.900	77.600
ECALL * pn s1 to s5 *: Program name	—	101.000	114.000	91.800	105.000	91.800	105.000	91.800	105.000
EFCALL * pn *: Program name	—	70.700	82.800	66.200	78.100	66.200	78.100	66.200	78.100
EFCALL * pn s1 to s5 *: Program name	—	86.500	107.000	78.800	91.600	78.800	91.600	78.800	91.600
XCALL	—	3.800	5.700	3.700	5.200	3.700	5.200	3.700	5.200
COM CCOM	When selecting I/O refresh only	12.800	29.100	12.400	28.600	12.400	28.600	12.400	28.600
	When selecting CC-Link refresh only (master station side)	16.000	39.500	15.500	39.100	15.500	39.100	15.500	39.100
	When selecting CC-Link refresh only (local station side)	16.100	39.500	15.500	39.100	15.500	39.100	15.500	39.100
	• When selecting MELSECNET/ H refresh only (Control station side) • When selecting CC-Link IE refresh only (Control station/ Master station side)	34.700	70.400	34.400	69.800	34.400	69.800	34.400	69.800
	• When selecting MELSECNET/ H refresh only (Normal station side) • When selecting CC-Link IE refresh only (Normal station/ Local station side)	34.700	70.400	34.400	69.800	34.400	69.800	34.400	69.800
	When selecting intelli auto refresh only	12.800	33.200	12.800	33.200	12.800	33.200	12.800	33.200
	When selecting I/O outside the group only (Input only)	7.900	21.100	7.700	20.700	7.700	20.700	7.700	20.700
	When selecting I/O outside the group only (Output only)	16.900	44.800	16.500	44.200	16.500	44.200	16.500	44.200
	When selecting I/O outside the group only (Both I/O)	22.600	52.600	22.400	52.600	22.400	52.600	22.400	52.600
	When selecting refresh of multiple CPU high speed transmission area only	13.000	33.800	12.700	33.200	12.700	33.200	12.700	33.200
	When selecting communication with peripheral device	7.250	18.800	7.100	18.500	7.100	18.500	7.100	18.500
FIFW	Number of data points = 0	3.700	5.300	3.200	4.600	3.200	4.600	3.200	4.600
	Number of data points = 96	3.800	4.400	3.300	3.800	3.300	3.800	3.300	3.800
FIFR	Number of data points = 01	4.300	5.000	3.800	4.400	3.800	4.400	3.800	4.400
	Number of data points = 96	33.500	35.500	24.800	25.700	24.800	25.700	24.800	25.700
FPOP	Number of data points = 01	4.300	5.900	3.800	5.300	3.800	5.300	3.800	5.300
	Number of data points = 96	4.300	5.900	3.700	5.400	3.700	5.400	3.700	5.400
FINS	Number of data points = 0	4.800	5.900	3.700	5.300	3.700	5.300	3.700	5.300
	Number of data points = 96	4.300	5.900	3.700	5.300	3.700	5.300	3.700	5.300
FDEL	Number of data points = 01	4.900	6.500	4.200	5.800	4.200	5.800	4.200	5.800
	Number of data points = 96	34.200	35.900	25.400	25.900	25.400	25.900	25.400	25.900
FROM (n1, n2, d, n3)	n3 = 1	10.800	24.100	10.700	23.600	10.700	23.600	10.700	23.600
	n3 = 1000	392.600	413.300	390.900	410.200	390.900	410.200	390.900	410.200

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)	Processing time (μs)							
		Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
DFRO (n1, n2, d, n3)	n3 = 1	13.600	27.700	12.600	26.700	12.600	26.700	12.600	26.700
	n3 = 500	392.600	413.300	390.900	410.200	390.900	410.200	390.900	410.200
TO (n1, n2, s, n3)	n3 = 1	10.200	21.900	9.600	21.300	9.600	21.300	9.600	21.300
	n3 = 1000	373.700	394.100	372.500	390.800	372.500	390.800	372.500	390.800
DTO (n1, n2, s, n3)	n3 = 1	13.000	26.700	12.000	25.700	12.000	25.700	12.000	25.700
	n3 = 500	373.700	394.100	372.500	390.800	372.500	390.800	372.500	390.800
LEDR	No display ==> no display	2.400	2.600	1.900	2.000	1.900	2.000	1.900	2.000
	LED instruction execution ==> no display	28.100	39.400	24.400	35.800	24.400	35.800	24.400	35.800
BINDA (s, d)	s = 1	4.900	6.500	4.300	5.600	4.300	5.600	4.300	5.600
	s = -32768	7.200	8.700	6.500	8.000	6.500	8.000	6.500	8.000
DBINDA (s, d)	s = 1	5.700	7.100	4.900	6.300	4.900	6.300	4.900	6.300
	s = -2147483648	10.400	12.000	9.600	11.000	9.600	11.000	9.600	11.000
BINHA(s, d)	s = 1	4.400	5.900	3.800	5.200	3.800	5.200	3.800	5.200
	s = FFFFH	4.400	5.800	3.700	5.200	3.700	5.200	3.700	5.200
DBINHA (s, d)	s = 1	5.200	6.700	4.600	6.000	4.600	6.000	4.600	6.000
	s = FFFFFFFFH	5.100	6.500	4.600	6.000	4.600	6.000	4.600	6.000
BCDDA (s, d)	s = 1	4.300	5.800	3.600	5.000	3.600	5.000	3.600	5.000
	s = 9999	4.700	6.100	4.100	5.400	4.100	5.400	4.100	5.400
DBCDDA(s, d)	s = 1	4.800	6.300	4.000	5.500	4.000	5.500	4.000	5.500
	s = 99999999	5.600	7.100	4.900	6.300	4.900	6.300	4.900	6.300
DABIN (s, d)	s = 1	6.500	8.500	5.800	7.800	5.800	7.800	5.800	7.800
	s = -32768	6.300	8.300	5.600	7.700	5.600	7.700	5.600	7.700
DDABIN (s, d)	s = 1	9.400	11.500	8.500	10.500	8.500	10.500	8.500	10.500
	s = -2147483648	9.100	11.200	8.100	10.200	8.100	10.200	8.100	10.200
HABIN (s, d)	s = 1	4.900	7.100	4.400	6.400	4.400	6.400	4.400	6.400
	s = FFFFH	5.100	7.300	4.600	6.500	4.600	6.500	4.600	6.500
DHABIN (s, d)	s = 1	6.000	8.100	5.300	7.300	5.300	7.300	5.300	7.300
	s = FFFFFFFFH	6.300	8.500	5.600	7.700	5.600	7.700	5.600	7.700
DABCD (s, d)	s = 1	5.000	7.100	4.400	6.300	4.400	6.300	4.400	6.300
	s = 9999	5.000	7.100	4.300	6.300	4.300	6.300	4.300	6.300
DDABCD (s, d)	s = 1	6.200	8.300	5.500	7.400	5.500	7.400	5.500	7.400
	s = 99999999	6.200	8.300	5.500	7.500	5.500	7.500	5.500	7.500
COMRD	—	51.600	52.400	50.900	51.200	50.900	51.200	50.900	51.200
LEN	1 character	4.100	6.200	3.600	5.500	3.600	5.500	3.600	5.500
	96 characters	19.800	22.200	16.800	18.700	16.800	18.700	16.800	18.700
STR	—	6.900	11.100	6.600	10.400	6.600	10.400	6.600	10.400
DSTR	—	10.200	12.500	9.600	11.500	9.600	11.500	9.600	11.500
VAL	—	9.800	14.200	8.900	13.000	8.900	13.000	8.900	13.000
DVAL	—	14.000	18.700	12.700	16.800	12.700	16.800	12.700	16.800
ESTR	—	18.700	24.100	17.900	23.100	17.900	23.100	17.900	23.100
EVAL	Decimal point format all 2-digit specification	23.300	30.400	22.800	29.000	22.800	29.000	22.800	29.000
	Exponent format all 6-digit specification	23.300	30.500	22.500	29.000	22.500	29.000	22.500	29.000
ASC (s, d, n)	n = 1	5.600	9.000	5.400	8.300	5.400	8.300	5.400	8.300
	n = 96	28.700	32.100	25.200	28.400	25.200	28.400	25.200	28.400
HEX (s, d, n)	n = 1	6.000	9.700	5.400	9.000	5.400	9.000	5.400	9.000
	n = 96	35.600	39.800	31.300	35.000	31.300	35.000	31.300	35.000
RIGHT (s, d, n)	n = 1	7.600	9.400	7.300	6.600	7.300	6.600	7.300	6.600
	n = 96	36.300	40.000	29.200	31.600	29.200	31.600	29.200	31.600
LEFT (s, d, n)	n = 1	6.500	8.900	5.900	8.200	5.900	8.200	5.900	8.200
	n = 96	36.200	39.700	29.200	31.500	29.200	31.500	29.200	31.500

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
MIDR	—		9.500	12.100	8.100	10.300	8.100	10.300	8.100	10.300
MIDW	—		10.300	12.000	8.800	10.200	8.800	10.200	8.800	10.200
INSTR	No match		19.300	21.800	16.600	18.400	16.600	18.400	16.600	18.400
	Match	Head	10.300	12.800	9.100	10.900	9.100	10.900	9.100	10.900
		End	51.100	54.200	42.700	44.900	42.700	44.900	42.700	44.900
EMOD	—		10.300	11.800	9.600	11.000	9.600	11.000	9.600	11.000
EREXP	—		19.300	21.000	18.800	20.100	18.800	20.100	18.800	20.100
STRINS (s, d, n)	s = 128 / d = 40 / n = 1		41.100	54.200	35.300	47.600	35.300	47.600	35.300	47.600
	s = 128 / d = 40 / n = 48		56.700	81.400	48.600	61.700	48.600	61.700	48.600	61.700
STRDEL (s, d, n)	s = 128 / d = 40 / n = 1		39.000	49.500	34.800	44.600	34.800	44.600	34.800	44.600
	s = 128 / d = 40 / n = 48		36.000	45.200	29.200	38.100	29.200	38.100	29.200	38.100
SIN	single precision		4.500	6.200	4.100	5.700	4.100	5.700	4.100	5.700
COS	single precision		4.300	6.000	4.000	5.600	4.000	5.600	4.000	5.600
TAN	single precision		5.100	7.200	5.100	6.700	5.100	6.700	5.100	6.700
ASIN	single precision		6.100	8.900	5.900	8.500	5.900	8.500	5.900	8.500
ACOS	single precision		6.800	9.300	6.700	8.900	6.700	8.900	6.700	8.900
ATAN	single precision		4.000	6.500	3.900	6.000	3.900	6.000	3.900	6.000
SIND	double precision		8.800	14.300	8.500	13.800	8.500	13.800	8.500	13.800
COSD	double precision		9.300	15.100	8.800	14.600	8.800	14.600	8.800	14.600
TAND	double precision		11.200	16.900	10.800	16.500	10.800	16.500	10.800	16.500
ASIND	double precision		12.000	17.100	11.600	16.600	11.600	16.600	11.600	16.600
ACOSD	double precision		11.700	16.500	11.200	16.200	11.200	16.200	11.200	16.200
ATAND	double precision		9.500	14.200	9.100	13.800	9.100	13.800	9.100	13.800
RAD	single precision		2.500	4.800	2.100	4.300	2.100	4.300	2.100	4.300
RADD	double precision		4.000	9.600	3.600	9.200	3.600	9.200	3.600	9.200
DEG	single precision		2.500	4.700	2.200	4.400	2.200	4.400	2.200	4.400
DEGD	double precision		4.300	9.000	3.800	9.000	3.800	9.000	3.800	9.000
SQR	single precision		3.000	4.600	2.600	4.300	2.600	4.300	2.600	4.300
SQRD	double precision		5.600	11.500	5.200	11.000	5.200	11.000	5.200	11.000
EVAL	Decimal point format all 2-digit specification		23.300	30.400	22.800	29.000	22.800	29.000	22.800	29.000
	Exponent format all 6-digit specification		23.300	30.500	22.500	29.000	22.500	29.000	22.500	29.000
ASC (s, d, n)	n = 1		5.600	9.000	5.400	8.300	5.400	8.300	5.400	8.300
	n = 96		28.700	32.100	25.200	28.400	25.200	28.400	25.200	28.400
HEX (s, d, n)	n = 1		6.000	9.700	5.400	9.000	5.400	9.000	5.400	9.000
	n = 96		35.600	39.800	31.300	35.000	31.300	35.000	31.300	35.000
RIGHT (s, d, n)	n = 1		7.600	9.400	7.300	6.600	7.300	6.600	7.300	6.600
	n = 96		36.300	40.000	29.200	31.600	29.200	31.600	29.200	31.600
LEFT (s, d, n)	n = 1		6.500	8.900	5.900	8.200	5.900	8.200	5.900	8.200
	n = 96		36.200	39.700	29.200	31.500	29.200	31.500	29.200	31.500
MIDR	—		9.500	12.100	8.100	10.300	8.100	10.300	8.100	10.300
MIDW	—		10.300	12.000	8.800	10.200	8.800	10.200	8.800	10.200
INSTR	No match		19.300	21.800	16.600	18.400	16.600	18.400	16.600	18.400
	Match	Head	10.300	12.800	9.100	10.900	9.100	10.900	9.100	10.900
		End	51.100	54.200	42.700	44.900	42.700	44.900	42.700	44.900
EMOD	—		10.300	11.800	9.600	11.000	9.600	11.000	9.600	11.000
EREXP	—		19.300	21.000	18.800	20.100	18.800	20.100	18.800	20.100
STRINS (s, d, n)	s = 128 / d = 40 / n = 1		41.100	54.200	35.300	47.600	35.300	47.600	35.300	47.600
	s = 128 / d = 40 / n = 48		56.700	81.400	48.600	61.700	48.600	61.700	48.600	61.700
STRDEL (s, d, n)	s = 128 / d = 40 / n = 1		39.000	49.500	34.800	44.600	34.800	44.600	34.800	44.600
	s = 128 / d = 40 / n = 48		36.000	45.200	29.200	38.100	29.200	38.100	29.200	38.100

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
SIN	single precision		4.500	6.200	4.100	5.700	4.100	5.700	4.100	5.700
COS	single precision		4.300	6.000	4.000	5.600	4.000	5.600	4.000	5.600
TAN	single precision		5.100	7.200	5.100	6.700	5.100	6.700	5.100	6.700
ASIN	single precision		6.100	8.900	5.900	8.500	5.900	8.500	5.900	8.500
ACOS	single precision		6.800	9.300	6.700	8.900	6.700	8.900	6.700	8.900
ATAN	single precision		4.000	6.500	3.900	6.000	3.900	6.000	3.900	6.000
SIND	double precision		8.800	14.300	8.500	13.800	8.500	13.800	8.500	13.800
COSD	double precision		9.300	15.100	8.800	14.600	8.800	14.600	8.800	14.600
TAND	double precision		11.200	16.900	10.800	16.500	10.800	16.500	10.800	16.500
ASIND	double precision		12.000	17.100	11.600	16.600	11.600	16.600	11.600	16.600
ACOSD	double precision		11.700	16.500	11.200	16.200	11.200	16.200	11.200	16.200
ATAND	double precision		9.500	14.200	9.100	13.800	9.100	13.800	9.100	13.800
RAD	single precision		2.500	4.800	2.100	4.300	2.100	4.300	2.100	4.300
RADD	double precision		4.000	9.600	3.600	9.200	3.600	9.200	3.600	9.200
DEG	single precision		2.500	4.700	2.200	4.400	2.200	4.400	2.200	4.400
DEGD	double precision		4.300	9.000	3.800	9.000	3.800	9.000	3.800	9.000
SQR	single precision		3.000	4.600	2.600	4.300	2.600	4.300	2.600	4.300
SQRD	double precision		5.600	11.500	5.200	11.000	5.200	11.000	5.200	11.000
EXP (s, d)	single precision	s = -10	4.000	6.100	3.800	5.500	3.800	5.500	3.800	5.500
		s = 1	4.000	6.100	3.800	5.600	3.800	5.600	3.800	5.600
EXPD (s, d)	double precision	s = -10	8.700	13.900	8.200	13.500	8.200	13.500	8.200	13.500
		s = 1	8.400	13.600	8.000	13.200	8.000	13.200	8.000	13.200
LOG (s, d)	single precision	s = 1	4.100	6.900	3.800	6.400	3.800	6.400	3.800	6.400
		s = 10	5.600	8.200	5.200	7.700	5.200	7.700	5.200	7.700
LOGD (s, d)	double precision	s = 1	8.100	13.000	7.700	12.500	7.700	12.500	7.700	12.500
		s = 10	9.700	14.800	9.200	14.300	9.200	14.300	9.200	14.300
RND	—		1.200	2.300	0.800	1.800	0.800	1.800	0.800	1.800
SRND	—		1.400	2.400	1.100	2.000	1.100	2.000	1.100	2.000
BSQR (s, d)	s = 0		1.800	3.300	1.600	2.800	1.600	2.800	1.600	2.800
	s = 9999		5.100	8.800	5.100	8.000	5.100	8.000	5.100	8.000
BDSQR (s, d)	s = 0		1.900	3.400	1.500	3.000	1.500	3.000	1.500	3.000
	s = 99999999		7.500	10.200	7.500	9.900	7.500	9.900	7.500	9.900
BSIN	—		8.600	15.100	8.100	14.500	8.100	14.500	8.100	14.500
BCOS	—		7.800	14.400	7.800	13.700	7.800	13.700	7.800	13.700
BTAN	—		9.000	13.800	9.000	13.300	9.000	13.300	9.000	13.300
BASIN	—		10.600	13.400	10.100	12.800	10.100	12.800	10.100	12.800
BACOS	—		11.600	14.400	11.100	14.100	11.100	14.100	11.100	14.100
BATAN	—		9.800	11.700	9.100	10.900	9.100	10.900	9.100	10.900
POW (s1, s2, d)	single precision	s1 = 12.3 E + 5; s2 = 3.45 E + 0	8,750	11.400	8.400	10.900	8.400	10.900	8.400	10.900
POWD (s1, s2, d)	double precision	s1 = 12.3 E + 5;								
LOG10	single precision		18.600	27.200	18.200	26.500	18.200	26.500	18.200	26.500
LOG10D	double precision									
LIMIT	—		5.900	8.550	5.700	8.050	5.700	8.050	5.700	8.050
DLIMIT	—		11.500	19.400	11.100	18.600	11.100	18.600	11.100	18.600
BAND	—		2.800	3.100	2.400	2.700	2.400	2.700	2.400	2.700
DBAND	—		3.200	3.500	2.800	3.000	2.800	3.000	2.800	3.000
ZONE	—		3.000	4.300	2.700	3.800	2.700	3.800	2.700	3.800
DZONE	—		3.600	5.100	3.300	4.600	3.300	4.600	3.300	4.600

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
SCL (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	13.200	23.600	12.300	22.500	12.300	22.500	12.300	22.500
		Point No. 9 < s1 < Point No. 10	13.300	23.600	12.600	22.700	12.600	22.700	12.600	22.700
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	12.000	23.100	11.400	22.200	11.400	22.200	11.400	22.200
		Point No. 9 < s1 < Point No. 10	14.100	25.300	12.800	23.900	12.800	23.900	12.800	23.900
DSCL (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	12.800	23.800	11.900	23.000	11.900	23.000	11.900	23.000
		Point No. 9 < s1 < Point No. 10	12.900	23.900	12.100	23.000	12.100	23.000	12.100	23.000
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	11.500	22.400	10.900	21.500	10.900	21.500	10.900	21.500
		Point No. 9 < s1 < Point No. 10	13.800	24.900	12.700	23.600	12.700	23.600	12.700	23.600
SCL2 (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	12.700	24.200	11.900	23.300	11.900	23.300	11.900	23.300
		Point No. 9 < s1 < Point No. 10	12.900	24.600	12.100	23.300	12.100	23.300	12.100	23.300
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	12.300	23.400	11.500	22.600	11.500	22.600	11.500	22.600
		Point No. 9 < s1 < Point No. 10	13.700	25.000	12.600	23.900	12.600	23.900	12.600	23.900
DSCL2 (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	12.600	23.800	11.800	22.900	11.800	22.900	11.800	22.900
		Point No. 9 < s1 < Point No. 10	13.000	23.900	12.200	22.800	12.200	22.800	12.200	22.800
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	11.500	22.400	11.000	21.400	11.000	21.400	11.000	21.400
		Point No. 9 < s1 < Point No. 10	13.900	24.900	12.800	23.600	12.800	23.600	12.800	23.600
RSET	Standard RAM		3.000	6.300	2.700	5.900	2.700	5.900	2.700	5.900
	SRAM card		3.000	6.400	2.600	5.800	2.600	5.800	2.600	5.800
QDRSET	SRAM card to standard RAM		120.000	134.000	115.000	134.000	115.000	134.000	115.000	134.000
	Standard RAM to SRAM card		533.000	560.000	520.000	553.000	520.000	553.000	520.000	553.000
QCDSET	SRAM card to standard ROM		306.000	346.000	305.000	346.000	305.000	346.000	305.000	346.000
	Standard ROM to SRAM card		311.000	342.000	300.000	334.000	300.000	334.000	300.000	334.000
DATERD	—		3.200	5.000	2.500	4.200	2.500	4.200	2.500	4.200
DATEWR	—		4.900	9.700	4.100	8.900	4.100	8.900	4.100	8.900
DATE +	No digit increase		5.100	8.000	4.700	6.600	4.700	6.600	4.700	6.600
	Digit increase		5.700	8.000	4.600	6.500	4.600	6.500	4.600	6.500
DATE -	No digit increase		5.800	8.500	4.600	7.000	4.600	7.000	4.600	7.000
	Digit increase		5.700	7.400	4.600	6.500	4.600	6.500	4.600	6.500
SECOND	—		2.600	3.900	2.200	3.400	2.200	3.400	2.200	3.400
HOUR	—		2.900	4.800	2.400	4.300	2.400	4.300	2.400	4.300
LDDT =	Comparison of specified date	continuity	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
		no continuity	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
	Comparison of current date	continuity	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
		no continuity	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
ANDDT=	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		no continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
	Comparison of current date	continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
		no continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
ORDT=	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current date	continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
		no continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
LDDT <>	Comparison of specified date	continuity	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
		no continuity	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
	Comparison of current date	continuity	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
		no continuity	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ANDDT<>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		no continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
	Comparison of current date	continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
		no continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
ORDT<>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current date	continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
		no continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
LDDT>	Comparison of specified date	continuity	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
		no continuity	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
	Comparison of current date	continuity	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
		no continuity	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
ANDDT>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		no continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
	Comparison of current date	continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
		no continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
ORDT>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current date	continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
		no continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
LDDT<=	Comparison of specified date	continuity	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
		no continuity	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
	Comparison of current date	continuity	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
		no continuity	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
ANDDT<=	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		no continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
	Comparison of current date	continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
		no continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
ORDT<=	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current date	continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
		no continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
LDDT<	Comparison of specified date	continuity	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
		no continuity	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
	Comparison of current date	continuity	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
		no continuity	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
ANDDT<	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		no continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
	Comparison of current date	continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
		no continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ORDT<	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current date	continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
		no continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
LDDT>=	Comparison of specified date	continuity	7.400	11.400	6.800	10.900	6.800	10.900	6.800	10.900
		no continuity	7.400	11.600	6.800	10.900	6.800	10.900	6.800	10.900
	Comparison of current date	continuity	5.900	10.000	5.500	9.700	5.500	9.700	5.500	9.700
		no continuity	5.900	10.100	5.500	9.700	5.500	9.700	5.500	9.700
ANDDT>=	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
		no continuity	7.200	11.400	6.500	10.700	6.500	10.700	6.500	10.700
	Comparison of current date	continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
		no continuity	5.700	9.900	5.300	9.300	5.300	9.300	5.300	9.300
ORDT>=	not executed			0.008		0.038		0.038		0.038
	Comparison of specified date	continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.400	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current date	continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
		no continuity	5.900	10.000	5.400	9.600	5.400	9.600	5.400	9.600
LDTM=	Comparison of specified clock	continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current clock	continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
		no continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
ANDTM=	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		no continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
	Comparison of current clock	continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
		no continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
ORTM=	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		no continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
	Comparison of current clock	continuity	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
LDTM<>	Comparison of specified clock	continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current clock	continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
		no continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
ANDTM<>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		no continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
	Comparison of current clock	continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
		no continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
ORTM<>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		no continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
	Comparison of current clock	continuity	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
		no continuity	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
LDTM>	Comparison of specified clock	continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current clock	continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
		no continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
ANDTM>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		no continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
	Comparison of current clock	continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
		no continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
ORTM>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		no continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
	Comparison of current clock	continuity	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
		no continuity	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
LDTM<>	Comparison of specified clock	continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current clock	continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
		no continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
ANDTM<>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		no continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
	Comparison of current clock	continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
		no continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
ORTM<>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		no continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
	Comparison of current clock	continuity	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
		no continuity	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
LDTM>	Comparison of specified clock	continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current clock	continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
		no continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
ANDTM>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
		no continuity	7.000	11.500	6.300	10.800	6.300	10.800	6.300	10.800
	Comparison of current clock	continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
		no continuity	5.500	9.900	5.100	9.500	5.100	9.500	5.100	9.500
ORTM>	not executed			0.008		0.038		0.038		0.038
	Comparison of specified clock	continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
		no continuity	7.300	11.500	6.600	10.800	6.600	10.800	6.600	10.800
	Comparison of current clock	continuity	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
		no continuity	5.900	9.900	5.300	9.500	5.300	9.500	5.300	9.500
LDTM<	Comparison of specified clock	continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
		no continuity	7.300	11.500	6.700	10.800	6.700	10.800	6.700	10.800
	Comparison of current clock	continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500
		no continuity	5.800	9.900	5.400	9.500	5.400	9.500	5.400	9.500

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
ANDTM<	not executed			0.480		0.320		0.240		0.240
	Comparison of specified clock	continuity	8.200	25.500	8.200	25.500	6.500	25.500	6.500	25.500
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	6.500	25.500
	Comparison of current clock	continuity	6.500	23.100	6.500	23.100	6.500	23.100	6.500	23.100
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	6.500	23.100
ORTM<	not executed			0.480		0.320		0.240		0.240
	Comparison of specified clock	continuity	8.200	25.500	8.200	25.500	6.500	25.500	6.500	25.500
		no continuity	8.200	25.500	8.200	25.500	6.500	25.500	6.500	25.500
	Comparison of current clock	continuity	6.500	23.100	6.500	23.100	6.500	23.100	6.500	23.100
		no continuity	6.500	23.100	6.500	23.100	6.500	23.100	6.500	23.100
S.DATERD	—		9.250	51.000	9.250	51.000	9.250	51.000	9.250	51.000
S.DATE +	No digit increase		16.800	75.400	16.800	75.400	16.800	75.400	16.800	75.400
	Digit increase		16.800	75.400	16.800	75.400	16.800	75.400	16.800	75.400
S.DATE -	No digit increase		17.600	75.300	17.600	75.300	17.600	75.300	17.600	75.300
	Digit increase		16.900	75.300	16.900	75.300	16.900	75.300	16.900	75.300
PSTOP	—		82.200	199.000	82.200	199.000	82.200	199.000	82.200	199.000
POFF	—		82.600	198.000	82.600	198.000	82.600	198.000	82.600	198.000
PSCAN	—		83.600	200.000	83.600	200.000	83.600	200.000	83.600	200.000
WDT	—		2.900	12.000	2.900	12.000	2.900	12.000	2.900	12.000
DUTY	—		7.700	27.500	7.700	27.500	7.700	27.500	7.700	27.500
TIMCHK	—		5.350	24.500	5.350	24.500	5.350	24.500	5.350	24.500
ZRRDB	File register of standard RAM		4.100	4.200	4.100	4.200	4.100	4.200	4.100	4.200
	File register of SRAM card		—	—	—	—	—	—	—	—
ZRWRB	File register of standard RAM		5.400	5.500	5.400	5.500	5.400	5.500	5.400	5.500
	File register of SRAM card		—	—	—	—	—	—	—	—
ADRSET	—		2.400	6.650	2.400	6.650	2.400	6.650	2.400	6.650
ZPUSH	—		9.200	20.500	9.200	20.500	9.200	20.500	9.200	20.500
ZPOP	—		9.000	15.500	9.000	15.500	9.000	15.500	9.000	15.500
S.ZCOM	When mounting CC-Link module (Master station side)		19.600	26.500	19.300	26.000	19.300	26.000	19.300	26.000
	When mounting CC-Link module (Local station side)		19.600	26.500	19.100	26.200	19.100	26.200	19.100	26.200
	When mounting MELSECNET/H module, CC-Link IE module (Control station side)		53.500	73.500	53.000	72.700	53.000	72.700	53.000	72.700
	When mounting MELSECNET/H module, CC-Link IE module (Normal station side)		29.800	41.200	29.800	40.600	29.800	40.600	29.800	40.600
S.RTREAD	—		5.900	11.000	5.400	10.500	5.400	10.500	5.400	10.500
S.RTWRITE	—		6.700	11.100	6.000	10.400	6.000	10.400	6.000	10.400
UNIRD (n1, d, n2)	n2 = 1		4.000	8.400	3.700	8.000	3.700	8.000	3.700	8.000
	n2 = 16		12.500	17.000	12.200	16.600	12.200	16.600	12.200	16.600
TYPERD			29.800	53.000	29.500	52.300	29.500	52.300	29.500	52.300
TRACE	Start		46.600	48.300	43.800	44.700	43.800	44.700	43.800	44.700
TRACER	—		3.300	6.800	2.600	6.000	2.600	6.000	2.600	6.000
RBMOV (s, d, n)	When standard RAM is used	1 point	11.300	16.800	9.200	15.100	9.200	15.100	9.200	15.100
		1000 points	120.700	127.100	61.000	68.600	61.000	68.600	61.000	68.600
	When SRAM card is used	1 point	11.200	16.700	9.400	15.600	9.400	15.600	9.400	15.600
		1000 points	180.700	187.100	165.000	172.600	165.000	172.600	165.000	172.600
SP.FWRITE	—		6.700	11.100	6.000	10.400	6.000	10.400	6.000	10.400
SP.FREAD	—		5.900	11.000	5.400	10.500	5.400	10.500	5.400	10.500
SP.DEVST	—		4.500	36.500	4.000	34.500	4.000	34.500	4.000	34.500
S.DEVLD	—		11.000	17.800	10.000	17.000	10.000	17.000	10.000	17.000
S.TO (n1, n2, n3, n4, d)	Writing to host CPU shared memory	n4 = 1	34.700	34.900	33.500	34.400	33.500	34.400	33.500	34.400
		n4 = 320	85.900	87.600	75.200	75.500	75.200	75.500	75.200	75.500

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Instruction	Processing (Device)		Processing time (μs)							
			Q03UD(E)		Q04/Q06UD(E)H		Q10/Q13/Q20/ Q26UD(E)H		Q50/Q100UDEH	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
TO (n1, n2, s, n3)	Writing to host CPU shared memory	n3 = 1	4.700	23.800	5.200	23.300	5.200	23.300	5.200	23.300
		n3 = 320	57.500	76.200	47.100	64.500	47.100	64.500	47.100	64.500
DTO (n1, n2, s, n3)	Writing to host CPU shared memory	n3 = 1	5.300	23.800	5.800	23.300	5.800	23.300	5.800	23.300
		n3 = 320	111.300	128.400	91.500	108.500	91.500	108.500	91.500	108.500
FROM (n1, n2, d, n3)	Reading from host CPU shared memory	n3 = 1	5.000	23.800	4.300	23.300	4.300	23.300	4.300	23.300
		n3 = 320	51.400	65.600	44.400	60.700	44.400	60.700	44.400	60.700
	Reading from other CPU shared memory	n3 = 1	11.600	17.700	10.600	13.900	10.600	13.900	10.600	13.900
		n3 = 320	142.000	160.000	142.000	149.000	142.000	149.000	142.000	149.000
		n3 = 1000	431.000	463.000	422.000	448.000	422.000	448.000	422.000	448.000
DFRO (n1, n2, d, n3)	Reading from host CPU shared memory	n3 = 1	5.200	23.800	5.600	23.300	5.600	23.300	5.600	23.300
		n3 = 320	96.400	113.200	83.600	100.800	83.600	100.800	83.600	100.800
	Reading from other CPU shared memory	n3 = 1	12.900	20.800	12.200	17.100	12.200	17.100	12.200	17.100
		n3 = 320	277.000	299.000	274.000	291.000	274.000	291.000	274.000	291.000
		n3 = 1000	838.000	860.000	835.000	857.000	835.000	857.000	835.000	857.000
D.DDWR (n, s1, s2, d1, d2)	Writes devices to another CPU	n=1	34.700	34.900	33.500	34.400	33.500	34.400	33.500	34.400
		n=16	85.900	87.600	75.200	75.500	75.200	75.500	75.200	75.500
		n=96	5.600	10.200	3.300	9.900	3.300	9.900	3.300	9.900
DP.DDWR (n, s1, s2, d1, d2)		n=1	36.700	42.400	34.300	39.200	34.300	39.200	34.300	39.200
		n=16	5.000	12.100	3.100	10.500	3.100	10.500	3.100	10.500
		n=96	59.100	66.800	55.300	65.100	55.300	65.100	55.300	65.100
D.DDRD (n, s1, s2, d1, d2)	Reads devices from another CPU	n=1	3.300	12.700	2.400	9.600	2.400	9.600	2.400	9.600
		n=16	50.900	64.400	45.200	48.200	45.200	48.200	45.200	48.200
		n=96	11.600	17.700	10.600	13.900	10.600	13.900	10.600	13.900
DP.DDRD (n, s1, s2, d1, d2)		n=1	142.000	160.000	142.000	149.000	142.000	149.000	142.000	149.000
		n=16	431.000	463.000	422.000	448.000	422.000	448.000	422.000	448.000
		n=96	6.700	12.600	2.800	9.900	2.800	9.900	2.800	9.900

Tab. A-23: *Processing times for instructions other than subset instructions for Universal model QCPU (2)*

Table of the time to be added when file register, extended data register, extended link register, module access device, and link direct device are used

- Q03UD(E) Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Device Name		Data	Device Specification Location	Processing time (μs)			
				Q00UJ	Q00U	Q01U	Q02U
File register (R)	When standard RAM is used	Bit	Source	0.100	0.100	0.100	0.100
			Destination	0.100	0.100	0.100	0.100
		Word	Source	0.100	0.100	0.100	0.100
			Destination	0.100	0.100	0.100	0.100
		Double word	Source	0.100	0.100	0.100	0.200
			Destination	0.100	0.100	0.100	0.200
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.220
			Destination	—	—	—	0.180
		Word	Source	—	—	—	0.220
			Destination	—	—	—	0.180
		Double word	Source	—	—	—	0.440
			Destination	—	—	—	0.380
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.160
			Destination	—	—	—	0.140
		Word	Source	—	—	—	0.160
			Destination	—	—	—	0.140
		Double word	Source	—	—	—	0.320
			Destination	—	—	—	0.300
File register (ZR), Extended data register (D), Extended link register (W)	When standard RAM is used	Bit	Source	0.120	0.120	0.120	0.120
			Destination	0.120	0.120	0.120	0.120
		Word	Source	0.120	0.120	0.120	0.120
			Destination	0.120	0.120	0.120	0.120
		Double word	Source	0.120	0.120	0.120	0.220
			Destination	0.120	0.120	0.120	0.220
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.240
			Destination	—	—	—	0.200
		Word	Source	—	—	—	0.240
			Destination	—	—	—	0.200
		Double word	Source	—	—	—	0.460
			Destination	—	—	—	0.400
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.180
			Destination	—	—	—	0.160
		Word	Source	—	—	—	0.180
			Destination	—	—	—	0.160
		Double word	Source	—	—	—	0.340
			Destination	—	—	—	0.320
Module access device (Multiple CPU high speed transmission area) (U3En\G10000)		Bit	Source	—	—	—	12.000
			Destination	—	—	—	17.300
		Word	Source	—	—	—	9.700
			Destination	—	—	—	33.000
		Double word	Source	—	—	—	24.200
			Destination	—	—	—	34.800

Tab. A-24: *Processing times to be added for instructions other than subset instructions for Universal model CPU (1)*

Device Name	Data	Device Specification Location	Processing time (μs)			
			Q00UJ	Q00U	Q01U	Q02U
Link direct device (Jn\□)	Bit	Source	—	—	—	32.900
		Destination	—	—	—	67.300
	Word	Source	—	—	—	37.200
		Destination	—	—	—	37.000
	Double word	Source	—	—	—	39.500
		Destination	—	—	—	41.900

Tab. A-24: *Processing times to be added for instructions other than subset instructions for Universal model CPU (1)*

- Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UDE(H)CPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU and Q100UDEHCPU

Device Name		Data	Device Specification Location	Processing time (μs)			
				Q03UD(E)	Q04/ Q06UD(E)H	Q10/Q13/ Q20/ Q26UD(E)H	Q50/ Q100UDEH
File register (R)	When standard RAM is used	Bit	Source	0.100	0.048	0.048	0.048
			Destination	0.100	0.038	0.038	0.038
		Word	Source	0.100	0.048	0.048	0.048
			Destination	0.100	0.038	0.038	0.038
		Double word	Source	0.200	0.095	0.095	0.095
			Destination	0.200	0.086	0.086	0.086
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.220	0.200	0.200	0.200
			Destination	0.180	0.162	0.162	0.162
		Word	Source	0.220	0.200	0.200	0.200
			Destination	0.180	0.162	0.162	0.162
		Double word	Source	0.440	0.399	0.399	0.399
			Destination	0.380	0.361	0.361	0.361
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.160	0.152	0.152	0.152
			Destination	0.140	0.133	0.133	0.133
		Word	Source	0.160	0.152	0.152	0.152
			Destination	0.140	0.133	0.133	0.133
		Double word	Source	0.320	0.304	0.304	0.304
			Destination	0.300	0.295	0.295	0.295
File register (ZR), Extended data register (D), Extended link register (W)	When standard RAM is used	Bit	Source	0.120	0.057	0.057	0.057
			Destination	0.120	0.048	0.048	0.048
		Word	Source	0.120	0.057	0.057	0.057
			Destination	0.120	0.048	0.048	0.048
		Double word	Source	0.220	0.105	0.105	0.105
			Destination	0.220	0.095	0.095	0.095
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.240	0.209	0.209	0.209
			Destination	0.200	0.171	0.171	0.171
		Word	Source	0.240	0.209	0.209	0.209
			Destination	0.200	0.171	0.171	0.171
		Double word	Source	0.460	0.409	0.409	0.409
			Destination	0.400	0.371	0.371	0.371
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.180	0.162	0.162	0.162
			Destination	0.160	0.143	0.143	0.143
		Word	Source	0.180	0.162	0.162	0.162
			Destination	0.160	0.143	0.143	0.143
		Double word	Source	0.340	0.314	0.314	0.314
			Destination	0.320	0.304	0.304	0.304
Module access device (Un\G□, U3En\G0 to G4095)		Bit	Source	11.700	11.200	11.200	11.200
			Destination	15.400	15.300	15.300	15.300
		Word	Source	9.460	9.410	9.410	9.410
			Destination	19.000	19.000	19.000	19.000
		Double word	Source	11.000	10.900	10.900	10.900
			Destination	18.800	18.700	18.700	18.700

Tab. A-25: *Processing times to be added for instructions other than subset instructions for Universal model CPU (2)*

Device Name	Data	Device Specification Location	Processing time (μs)			
			Q03UD(E)	Q04/ Q06UD(E)H	Q10/Q13/ Q20/ Q26UD(E)H	Q50/ Q100UDEH
Link direct device (Jn\□)	Bit	Source	32.700	31.300	31.300	31.300
		Destination	52.300	29.900	29.900	29.900
	Word	Source	28.500	17.300	17.300	17.300
		Destination	27.500	14.700	14.700	14.700
	Double word	Source	30.300	18.100	18.100	18.100
		Destination	30.600	15.700	15.700	15.700

Tab. A-25: *Processing times to be added for instructions other than subset instructions for Universal model CPU (2)*

A.4 Operation Processing Time of LCPU

NOTE

- *The processing time shown in section A.4.1 applies when the device used in an instruction meets the device condition for subset processing (for device condition triggering subset processing, refer to section 3.8.1).*
- *When using a file register (R, ZR), extended data register (D), and extended link register (W), add the processing time shown in table A-27 to that of the instruction.*
- *When using an F, T(ST) or C device with an OUT/SET/RST instruction, add the processing time for each instruction, with reference to the adding time in table A-28.*
- *Since the processing time of an instruction varies depending on that of the cache function, both the minimum and maximum values are described in the table.*

A.4.1 Subset instruction processing time

<table>
<tr><th rowspan="3">Instruction</th><th colspan="2" rowspan="3">Condition (Device)</th><th colspan="4">Processing time (μs)</th></tr>
<tr><th colspan="2">L02CPU</th><th colspan="2">L26CPU-BT</th></tr>
<tr><th>Min.</th><th>Max.</th><th>Min.</th><th>Max.</th></tr>
<tr><td>LD</td><td colspan="2" rowspan="12">executed</td><td colspan="2" rowspan="12">0.040</td><td colspan="2" rowspan="12">0.0095</td></tr>
<tr><td>LDI</td></tr>
<tr><td>AND</td></tr>
<tr><td>ANI</td></tr>
<tr><td>OR</td></tr>
<tr><td>ORI</td></tr>
<tr><td>LDP</td></tr>
<tr><td>LDF</td></tr>
<tr><td>ANDP</td></tr>
<tr><td>ANDF</td></tr>
<tr><td>ORP</td></tr>
<tr><td>ORF</td></tr>
<tr><td>LDPI
LDFI</td><td colspan="2">executed</td><td colspan="2">0.120</td><td colspan="2">0.0285</td></tr>
<tr><td>ANDPI</td><td colspan="2" rowspan="4">executed</td><td colspan="2" rowspan="4">0.160</td><td colspan="2" rowspan="4">0.038</td></tr>
<tr><td>ANDFI</td></tr>
<tr><td>ORPI</td></tr>
<tr><td>ORFI</td></tr>
<tr><td rowspan="2">OUT</td><td colspan="2">not changed</td><td colspan="2" rowspan="2">0.040</td><td colspan="2" rowspan="2">0.0095</td></tr>
<tr><td colspan="2">changed</td></tr>
<tr><td rowspan="2">OUT H</td><td colspan="2">not changed</td><td colspan="2" rowspan="2">0.040</td><td colspan="2" rowspan="2">0.0095</td></tr>
<tr><td colspan="2">changed</td></tr>
<tr><td rowspan="4">SET
RST</td><td colspan="2">not executed</td><td colspan="2" rowspan="4">0.040</td><td colspan="2" rowspan="4">0.0095</td></tr>
<tr><td rowspan="3">executed</td><td>not changed</td></tr>
<tr><td>changed</td></tr>
<tr><td rowspan="2">LD=</td><td colspan="2">continuity</td><td colspan="2" rowspan="2">0.120</td><td colspan="2" rowspan="2">0.0285</td></tr>
<tr><td colspan="2">no continuity</td></tr>
<tr><td rowspan="4">AND=</td><td colspan="2">not executed</td><td colspan="2" rowspan="4">0.120</td><td colspan="2" rowspan="4">0.0285</td></tr>
<tr><td rowspan="3">executed</td><td>continuity</td></tr>
<tr><td>no continuity</td></tr>
<tr><td rowspan="4">OR=</td><td colspan="2">not executed</td><td colspan="2" rowspan="4">0.120</td><td colspan="2" rowspan="4">0.0285</td></tr>
<tr><td rowspan="3">executed</td><td>continuity</td></tr>
<tr><td>no continuity</td></tr>
</table>

Tab. A-26: *Subset instruction processing time for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
LD<>	continuity		0.120		0.0285	
	no continuity					
AND<>	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
OR<>	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LD>	continuity		0.120		0.0285	
	no continuity					
AND>	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
OR>	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LD<=	continuity		0.120		0.0285	
	no continuity					
AND<=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
OR<=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LD<	continuity		0.120		0.0285	
	no continuity					
AND<	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
OR<	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LD>=	continuity		0.120		0.0285	
	no continuity					
AND>=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
OR>=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LDD=	continuity		0.120		0.0285	
	no continuity					
ANDD=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
ORD=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LDD<>	continuity		0.120		0.0285	
	no continuity					

Tab. A-26: *Subset instruction processing time for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
ANDD<>	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
ORD<>	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LDD>	continuity		0.120		0.0285	
	no continuity					
ANDD>	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
ORD>	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LDD<=	continuity		0.120		0.0285	
	no continuity					
ANDD<=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
ORD<=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LDD<	continuity		0.120		0.0285	
	no continuity					
ANDD<	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
ORD<	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
LDD>=	continuity		0.120		0.0285	
	no continuity					
ANDD>=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
ORD>=	not executed		0.120		0.0285	
	executed	continuity				
		no continuity				
+ (s, d)	executed		0.120		0.0285	
+ (s1, s2, d)	executed		0.160		0.038	
- (s, d)	executed		0.120		0.0285	
- (s1, s2, d)	executed		0.160		0.038	
d + (s, d)	executed		0.120		0.0285	
d + (s1, s2, d)	executed		0.160		0.038	
d - (s, d)	executed		0.120		0.0285	
d - (s1, s2, d)	executed		0.160		0.038	
* (s1, s2, d)	executed		0.180		0.057	
/ (s1, s2, d)	executed		0.280		0.105	
d * (s1, s2, d)	executed		0.260		0.095	
d/ (s1, s2, d)	executed		0.400		0.162	

Tab. A-26: *Subset instruction processing time for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
B + (s, d)	executed		3.100	6.800	2.900	4.100
B + (s1, s2, d)	executed		4.800	8.900	4.200	5.900
B - (s, d)	executed		3.100	6.800	2.900	4.100
B - (s1, s2, d)	executed		4.800	8.900	4.200	4.600
B * (s1, s2, d)	executed		3.900	7.400	3.400	4.800
B/ (s1, s2, d)	executed		3.900	8.500	3.700	5.200
E + (s, d)	single precision	s = 0, d = 0	0.180		0.057	
		$s = 2^{127}$, $d = 2^{127}$	0.180		0.057	
E + (s1, s2, d)	single precision	s1= 0, s2 = 0	0.220		0.0665	
		$s1 = 2^{127}$, $s2 = 2^{127}$	0.220		0.0665	
E - (s, d)	single precision	s = 0, d = 0	0.180		0.057	
		$s = 2^{127}$, $d = 2^{127}$	0.180		0.057	
E - (s1, s2, d)	single precision	s1= 0, s2 = 0	0.220		0.0665	
		$s1 = 2^{127}$, $s2 = 2^{127}$	0.220		0.0665	
E * (s1, s2, d)	single precision	s1= 0, s2 = 0	0.180		0.057	
		$s1 = 2^{127}$, $s2 = 2^{127}$	0.180		0.057	
E/ (s1, s2, d)	single precision	$s1 = 2^{127}$, $s2 = 2^{127}$	3.900	8.500	0.285	
INC	executed		0.080		0.019	
DINC	executed		0.080		0.019	
DEC	executed		0.080		0.019	
DDEC	executed		0.080		0.019	
BCD	executed		0.160		0.057	
DBCD	executed		0.240		0.095	
BIN	executed		0.100		0.0285	
DBIN	executed		0.100		0.0285	
FLT	single precision	s = 0	0.100		0.0475	
		$s = 7FFF_H$	0.140		0.0475	
DFLT	single precision	s = 0	0.140		0.0475	
		$s = 7FFFFFFF_H$	0.140		0.0475	
INT	single precision	s = 0	0.140		0.0475	
		s = 32766.5	0.140		0.0475	
DINT	single precision	s = 0	0.140		0.0475	
		s = 1234567890.3	0.140		0.0475	
MOV	–		0.080		0.019	
DMOV	–		0.080		0.019	
EMOV	–		0.080		0.019	
CML	–		0.080		0.019	
DCML	–		0.080		0.019	
BMOV	SM237=ON	n=1	3.600	4.100	2.900	3.200
		n=96	4.500	4.700	3.400	3.700
	SM237= OFF	n=1	5.000	7.400	4.200	5.500
		n=96	6.000	7.900	4.700	6.000
FMOV	SM237=ON	n=1	5.900	6.800	2.800	3.200
		n=96	6.300	11.000	3.000	5.200
	SM237=OFF	n=1	7.000	8.000	3.400	3.800
		n=96	5.200	6.900	3.600	5.800
XCH		–	2.100	4.100	1.800	2.300
DXCH		–	2.200	4.200	2.100	2.900

Tab. A-26: *Subset instruction processing time for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
DFMOV	SM237=ON	n=1	2.000	3.200	1.750	1.750
		n=96	5.600	6.100	3.650	4.150
	SM237= OFF	n=1	2.900	4.600	2.250	3.150
		n=96	6.100	8.200	4.200	5.500
CJ	–		2.100	2.900	1.100	2.400
SCJ	–		2.100	2.900	1.100	2.400
JMP	–		2.100	2.900	1.100	2.400
WAND (s, d)	executed		0.120		0.0285	
WAND (s1, s2, d)	executed		0.160		0.038	
DAND (s, d)	executed		0.120		0.0285	
DAND (s1, s2, d)	executed		0.160		0.038	
WOR (s, d)	executed		0.120		0.0285	
WOR (s1, s2, d)	executed		0.160		0.038	
DOR (s, d)	executed		0.120		0.0285	
DOR (s1, s2, d)	executed		0.160		0.038	
WXOR (s, d)	executed		0.120		0.0285	
WXOR (s1, s2, d)	executed		0.160		0.038	
DXOR (s, d)	executed		0.120		0.0285	
DXOR (s1, s2, d)	executed		0.160		0.038	
WXNR (s, d)	executed		0.120		0.0285	
WXNR (s1, s2, d)	executed		0.160		0.038	
DXNR (s, d)	executed		0.120		0.0285	
DXNR (s1, s2, d)	executed		0.160		0.038	
ROR (d, n)	n = 1		2.200	4.900	1.700	2.500
	n = 15		2.200	4.900	1.700	2.500
RCR (d, n)	n = 1		2.100	4.800	1.700	3.200
	n = 15		2.100	4.800	1.700	3.200
ROL (d, n)	n = 1		2.100	4.800	1.800	3.200
	n = 15		2.100	4.800	1.800	3.200
RCL (d, n)	n = 1		2.100	5.200	1.800	2.200
	n = 15		2.100	5.200	1.800	2.200
DROR (d, n)	n = 1		2.200	5.200	1.900	2.700
	n = 31		2.200	5.200	1.900	2.700
DRCR (d, n)	n = 1		2.200	5.900	1.900	4.200
	n = 31		2.200	5.900	1.900	4.200
DROL (d, n)	n = 1		2.200	4.900	1.800	3.300
	n = 31		2.200	4.900	1.800	3.300
DRCL (d, n)	n = 1		2.200	5.900	1.900	3.800
	n = 31		2.200	5.900	1.900	3.800
SFR (d, n)	n = 1		2.200	4.600	1.700	2.600
	n = 15		2.200	4.600	1.700	2.600
SFL (d, n)	n = 1		2.200	4.600	1.800	2.700
	n = 15		2.200	4.600	1.800	2.700
DSFR (d, n)	n = 1		2.200	6.100	2.200	4.300
	n = 96		33.400	38.100	23.900	26.100
DSFL (d, n)	n = 1		2.200	6.100	2.100	4.000
	n = 96		33.500	38.000	23.700	25.800
SUM	s = 0		3.000	4.800	2.900	3.600
	s = FFFFH		3.000	4.900	2.900	3.600
SEG	executed		1.700	3.600	1.500	2.100

Tab. A-26: *Subset instruction processing time for LCPU*

Instruction	Condition (Device)	Processing time (μs)			
		L02CPU		L26CPU-BT	
		Min.	Max.	Min.	Max.
FOR	–	1.300	3.200	0.870	2.100
CALL pn	Internal file pointer	2.600	4.000	2.300	3.600
	Common pointer	4.600	13.500	3.200	4.900
CALL pn s1 to s5	–	31.200	36.000	26.100	29.300

Tab. A-26: *Subset instruction processing time for LCPU*

NOTE *For the instructions for which a leading edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.*

Example: MOVP instruction, WANDP instruction etc.

Table of the time to be added when file register, extended data register, and extended link register are used

Device Name		Data	Device Specification Location	Processing time (μs)	
				L02CPU	L26CPU-BT
File register (R)	When standard RAM is used	Bit	Source	0.100	0.048
			Destination	0.220	0.038
		Word	Source	0.100	0.048
			Destination	0.100	0.038
		Double word	Source	0.200	0.095
			Destination	0.200	0.086
File register (ZR), Extended data register (D), Extended link register (W)	When standard RAM is used	Bit	Source	0.140	0.057
			Destination	0.280	0.048
		Word	Source	0.140	0.057
			Destination	0.140	0.048
		Double word	Source	0.240	0.105
			Destination	0.240	0.095

Tab. A-27: *Processing times to be added for instructions other than subset instruction for LCPU*

Table of the time to be added when F/T(ST)/C device is used in OUT/SET/RST instruction

Instruction Name	Device Name	Condition		Processing time (μs)	
				L02CPU	L26CPU-BT
OUT	F	not executed		2.000	1.570
		executed	when displayed	53.100	38.090
			display completed	53.000	37.980
	T(ST), C	not executed		0.120	0.030
		executed	when displayed	0.120	0.030
			display completed	0.120	0.030
SET	F	not executed		0.040	0.010
		executed	when displayed	52.000	40.600
			display completed	43.600	37.900
RST	F	not executed		0.040	0.010
		executed	when displayed	45.700	36.600
			display completed	19.000	16.190
	T(ST), C	not executed		0.120	0.030
		executed		0.120	0.030

Tab. A-28: *Processing times to be added for LCPU and OUT/SET/RST instructions*

A.4.2 Processing time of instructions other than subset instruction

NOTE

- *The processing time shown in table A-29 applies when the device used in an instruction does not meet the device condition for subset processing (for device condition that does not trigger subset processing, refer to section 3.8.1).*
- *For instructions not shown in the following table, refer to table A-26 in section A.4.1.*
- *When using a file register (R, ZR), extended data register (D), extended link register (W), and module access device (Un/G□), add the processing time shown in table A-30 to that of the instruction.*
- *Since the processing time of an instruction varies depending on that of the cache function, both the minimum and maximum values are described in the table.*

Instruction	Condition (Device)			Processing time (μs)			
				L02CPU		L26CPU-BT	
				Min.	Max.	Min.	Max.
ANB	—				0.040		0.0095
ORB							
MPS							
MRD							
MPP							
INV	not executed				0.040		0.0095
	executed						
MEP	not executed				0.040		0.0095
MEF	executed						
EGP	not executed				0.040		0.0095
EGF	executed						
PLS	–			1.600	1.700	0.890	1.200
PLF	–s			1.600	1.700	0.890	1.200
FF	not executed				0.080		0.0185
	executed			1.500	1.500	0.790	0.910
DELTA	not executed				0.080		0.0185
	executed			2.700	6.800	2.400	3.200
SFT	not executed				0.080		0.0185
	executed			1.700	4.300	1.100	2.700
MC	–				0.080		0.0185
MCR	–				0.040		0.0185
FEND	eror check performed			170.000	210.000	130.000	170.000
END	no error check performed			170.000	210.000	130.000	170.000
STOP	–			—		—	
NOP	–				0.040		0.0095
NOPLF							
PAGE							
LDE=	single precision	continuity		3.900	10.000		0.0285
		no continuity		3.900	10.000		0.0285
ANDE=	single precision	not executed			0.120		
		executed	continuity	3.400	9.300		0.0285
			no continuity	3.400	9.300		0.0285
ORE=	single precision	not executed			0.120		0.0285
		executed	continuity	3.500	8.500		0.0285
			no continuity	3.500	8.500		0.0285

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)			Processing time (μs)			
				L02CPU		L26CPU-BT	
				Min.	Max.	Min.	Max.
LDE< >	single precision	continuity		3.900	10.000		0.0285
		no continuity		3.900	10.000		0.0285
ANDE< >	single precision	not executed			0.120		0.0285
		executed	continuity	3.400	9.300		0.0285
			no continuity	3.400	9.300		0.0285
ORE< >	single precision	not executed			0.120		0.0285
		executed	continuity	3.400	9.300		0.0285
			no continuity	3.400	9.300		0.0285
LDE>	single precision	continuity		3.900	10.000		0.0285
		no continuity		3.900	10.000		0.0285
ANDE>	single precision	not executed			0.120		0.0285
		executed	continuity	3.400	9.300		0.0285
			no continuity	3.400	9.300		0.0285
ORE>	single precision	not executed			0.120		0.0285
		executed	continuity	3.500	8.500		0.0285
			no continuity	3.500	8.500		0.0285
LDE<=	single precision	continuity		3.900	10.000		0.0285
		no continuity		3.900	10.000		0.0285
ANDE<=	single precision	not executed			0.120		0.0285
		executed	continuity	3.400	9.300		0.0285
			no continuity	3.400	9.300		0.0285
ORE<=	single precision	not executed			0.120		0.0285
		executed	continuity	3.500	8.500		0.0285
			no continuity	3.500	8.500		0.0285
LDE<	single precision	continuity		3.900	10.000		0.0285
		no continuity		3.900	10.000		0.0285
ANDE<	single precision	not executed			0.120		0.0285
		executed	continuity	3.400	9.300		0.0285
			no continuity	3.400	9.300		0.0285
ORE<	single precision	not executed			0.120		0.0285
		executed	continuity	3.500	8.500		0.0285
			no continuity	3.500	8.500		0.0285
LDE>=	single precision	continuity		3.900	10.000		0.0285
		no continuity		3.900	10.000		0.0285
ANDE>=	single precision	not executed			0.120		0.0285
		executed	continuity	3.400	9.300		0.0285
			no continuity	3.400	9.300		0.0285
ORE>=	single precision	not executed			0.120		0.0285
		executed	continuity	3.500	8.500		0.0285
			no continuity	3.500	8.500		0.0285
LDED=	double precision	continuity		4.800	16.000	3.500	9.000
		no continuity		4.800	16.000	3.500	9.000
ANDED=	double precision	not executed			0.120		0.0285
		executed	continuity	4.400	15.100	3.200	7.500
			no continuity	4.400	15.100	3.200	7.500
ORED=	double precision	not executed			0.120		0.0285
		executed	continuity	4.500	14.900	3.400	9.200
			no continuity	4.500	14.900	3.400	9.200
LDED<>	double precision	continuity		4.800	16.000	3.500	9.000
		no continuity		4.800	16.000	3.500	9.000

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)			Processing time (μs)			
				L02CPU		L26CPU-BT	
				Min.	Max.	Min.	Max.
ANDED<>	double precision	not executed			0.120		0.0285
		executed	continuity	4.400	15.100	3.200	7.500
			no continuity	4.400	15.100	3.200	7.500
ORED<>	double precision	not executed			0.120		0.0285
		executed	continuity	4.500	14.900	3.400	9.200
			no continuity	4.500	14.900	3.400	9.200
LDED>	double precision	continuity		4.800	16.000	3.500	9.000
		no continuity		4.800	16.000	3.500	9.000
ANDED>	double precision	not executed			0.120		0.0285
		executed	continuity	4.400	15.100	3.200	7.500
			no continuity	4.400	15.100	3.200	7.500
ORED>	double precision	not executed			0.120		0.0285
		executed	continuity	4.500	14.900	3.400	9.200
			no continuity	4.500	14.900	3.400	9.200
LDED<=	double precision	continuity		4.800	16.000	3.500	9.000
		no continuity		4.800	16.000	3.500	9.000
ANDED<=	double precision	not executed			0.120		0.0285
		executed	continuity	4.400	15.100	3.200	7.500
			no continuity	4.400	15.100	3.200	7.500
ORED<=	double precision	not executed			0.120		0.0285
		executed	continuity	4.500	14.900	3.400	9.200
			no continuity	4.500	14.900	3.400	9.200
LDED<	double precision	continuity		4.800	16.000	3.500	9.000
		no continuity		4.800	16.000	3.500	9.000
ANDED<	double precision	not executed			0.120		0.0285
		executed	continuity	4.400	15.100	3.200	7.500
			no continuity	4.400	15.100	3.200	7.500
ORED<	double precision	not executed			0.120		0.0285
		executed	continuity	4.500	14.900	3.400	9.200
			no continuity	4.500	14.900	3.400	9.200
LDED>=	double precision	continuity		4.800	16.000	3.500	9.000
		no continuity		4.800	16.000	3.500	9.000
ANDED>=	double precision	not executed			0.120		0.0285
		executed	continuity	4.400	15.100	3.200	7.500
			no continuity	4.400	15.100	3.200	7.500
ORED>=	double precision	not executed			0.120		0.0285
		executed	continuity	4.500	14.900	3.400	9.200
			no continuity	4.500	14.900	3.400	9.200
LD$=	continuity			5.600	17.100	4.200	8.200
	no continuity			5.600	17.100	4.200	8.200
AND$=	not executed				0.120		0.0285
	executed	continuity		5.300	16.400	3.900	7.300
		no continuity		5.300	16.400	3.900	7.300
OR$=	not executed				0.120		0.0285
	executed	continuity		5.200	15.700	4.000	7.600
		no continuity		5.200	15.700	4.000	7.600
LD$< >	continuity			5.600	17.100	4.200	8.200
	no continuity			5.600	17.100	4.200	8.200

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
AND$< >	not executed			0.120		0.0285
	executed	continuity	5.300	16.400	3.900	7.300
		no continuity	5.300	16.400	3.900	7.300
OR$< >	not executed			0.120		0.0285
	executed	continuity	5.200	15.700	4.000	7.600
		no continuity	5.200	15.700	4.000	7.600
LD$>	continuity		5.600	17.100	4.200	8.200
	no continuity		5.600	17.100	4.200	8.200
AND$>	not executed			0.120		0.0285
	executed	continuity	5.300	16.400	3.900	7.300
		no continuity	5.300	16.400	3.900	7.300
OR$>	not executed			0.120		0.0285
	executed	continuity	5.200	15.700	4.000	7.600
		no continuity	5.200	15.700	4.000	7.600
LD$<=	continuity		5.600	17.100	4.200	8.200
	no continuity		5.600	17.100	4.200	8.200
AND$<=	not executed			0.120		0.0285
	executed	continuity	5.300	16.400	3.900	7.300
		no continuity	5.300	16.400	3.900	7.300
OR$<=	not executed			0.120		0.0285
	executed	continuity	5.200	15.700	4.000	7.600
		no continuity	5.200	15.700	4.000	7.600
LD$<	continuity		5.600	17.100	4.200	8.200
	no continuity		5.600	17.100	4.200	8.200
AND$<	not executed			0.120		0.0285
	executed	continuity	5.300	16.400	3.900	7.300
		no continuity	5.300	16.400	3.900	7.300
OR$<	not executed			0.120		0.0285
	executed	continuity	5.200	15.700	4.000	7.600
		no continuity	5.200	15.700	4.000	7.600
LD$>=	continuity		5.600	17.100	4.200	8.200
	no continuity		5.600	17.100	4.200	8.200
AND$>=	not executed			0.120		0.0285
	executed	continuity	5.300	16.400	3.900	7.300
		no continuity	5.300	16.400	3.900	7.300
OR$>=	not executed			0.120		0.0285
	executed	continuity	5.200	15.700	4.000	7.600
		no continuity	5.200	15.700	4.000	7.600
BKCMP = (s1, s2, d, n)	n = 1		9.200	15.600	7.500	10.100
	n = 96		60.700	69.100	45.600	50.500
BKCMP<> (s1, s2, d, n)	n = 1		9.200	15.600	7.500	10.100
	n = 96		60.700	69.100	45.600	50.500
BKCMP> (s1, s2, d, n)	n = 1		9.200	15.600	7.500	10.100
	n = 96		60.700	69.100	45.600	50.500
BKCMP<= (s1, s2, d, n)	n = 1		9.200	15.600	7.500	10.100
	n = 96		60.700	69.100	45.600	50.500
BKCMP< (s1, s2, d, n)	n = 1		9.200	15.600	7.500	10.100
	n = 96		60.700	69.100	45.600	50.500
BKCMP>= (s1, s2, d, n)	n = 1		9.200	15.600	7.500	10.100
	n = 96		60.700	69.100	45.600	50.500

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
DBKCMP = (s1, s2, d, n)	n = 1		9.700	16.400	8.600	13.000
	n = 96		61.200	69.900	47.900	52.800
DBKCMP<> (s1, s2, d, n)	n = 1		9.700	16.400	8.600	13.000
	n = 96		61.200	69.900	47.900	52.800
DBKCMP> (s1, s2, d, n)	n = 1		9.700	16.400	8.600	13.000
	n = 96		61.200	69.900	47.900	52.800
DBKCMP<= (s1, s2, d, n)	n = 1		9.700	16.400	8.600	13.000
	n = 96		61.200	69.900	47.900	52.800
DBKCMP< (s1, s2, d, n)	n = 1		9.700	16.400	8.600	13.000
	n = 96		61.200	69.900	47.900	52.800
DBKCMP>= (s1, s2, d, n)	n = 1		9.700	16.400	8.600	13.000
	n = 96		61.200	69.900	47.900	52.800
DB + (s, d)	executed		4.800	8.400	4.600	6.400
DB + (s1, s2, d)	executed		5.100	8.700	4.800	6.700
DB - (s, d)	executed		4.800	8.400	4.600	6.400
DB - (s1, s2, d)	executed		5.100	8.700	4.800	6.700
DB * (s1, s2, d)	executed		8.700	18.900	8.100	11.600
DB/ (s1, s2, d)	executed		6.100	9.100	5.800	8.800
ED + (s, d)	double precision	s = 0, d = 0	4.800	8.000	4.300	7.200
		$s = 2^{1023}$, $d = 2^{1023}$	5.400	14.900	4.300	7.200
ED + (s1, s2, d)	double precision	s1 = 0, s2 = 0	5.500	9.800	4.800	9.200
		$s = 2^{1023}$, $d = 2^{1023}$	6.100	17.800	4.800	9.200
ED - (s, d)	double precision	s = 0, d = 0	4.400	10.800	4.400	7.500
		$s = 2^{1023}$, $d = 2^{1023}$	5.400	15.500	4.400	7.500
ED - (s1, s2, d)	double precision	s1 = 0, s2 = 0	4.700	13.900	3.800	7.500
		$s = 2^{1023}$, $d = 2^{1023}$	5.700	17.200	3.800	7.500
ED * (s1, s2, d)	double precision	s1 = 0, s2 = 0	5.800	9.500	5.100	8.800
		$s = 2^{1023}$, $d = 2^{1023}$	5.900	17.600	5.100	8.800
ED / (s1, s2, d)	double precision	$s = 2^{1023}$, $d = 2^{1023}$	7.300	18.700	5.900	10.000
BK + (s1, s2, d, n)	n = 1		9.100	11.200	8.500	10.600
	n = 96		60.500	66.200	44.600	47.900
BK - (s1, s2, d, n)	n = 1		9.700	12.000	8.900	11.300
	n = 96		60.500	66.200	44.600	47.900
DBK + (s1, s2, d, n)	n = 1		7.500	12.400	6.450	9.950
	n = 96		59.900	65.200	43.700	47.500
DBK - (s1, s2, d, n)	n = 1		7.500	12.400	6.450	9.950
	n = 96		59.900	65.200	43.700	47.500
$ + s d	–		11.200	24.700	8.100	13.900
$ + (s1, s2, d)	–		7.900	16.600	6.500	10.300
FLTD	double precision	s = 0	2.800	9.400	1.800	4.700
		$s = 7FFF_H$	3.300	9.600	2.200	4.800
DFLTD	double precision	s = 0	2.900	9.100	2.000	4.900
		$s = 7FFFFFFF_H$	3.400	9.300	2.300	5.100
INTD	double precision	s = 0	3.500	8.700	2.200	4.100
		s = 32766.5	4.100	12.900	3.200	5.600
DINTD	double precision	s = 0	3.200	9.500	2.200	3.400
		s = 1234567890.3	4.100	13.400	3.000	5.100
DBL	executed		2.500	4.400	2.300	2.700
WORD	executed		2.800	3.900	2.600	3.600
GRY	executed		2.700	4.300	2.300	3.000

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)	Processing time (μs)			
		L02CPU		L26CPU-BT	
		Min.	Max.	Min.	Max.
DGRY	executed	2.700	4.300	2.300	3.000
GBIN	executed	4.000	6.400	3.800	4.300
DGBIN	executed	5.000	6.900	5.000	5.900
NEG	executed	2.100	4.400	2.000	3.300
DNEG	executed	2.500	3.700	2.500	3.300
ENEG	Floating point = 0	2.500	3.300	2.300	2.800
	Floating point = -1.0	2.800	5.600	2.500	3.900
EDNEG	Floating point = 0	3.000	8.800	1.800	3.100
	Floating point = -1.0	2.700	9.400	1.900	3.000
BKBCD (s, d, n)	n = 1	6.000	13.400	5.900	8.200
	n = 96	83.300	91.400	61.000	63.400
BKBIN (s, d, n)	n = 1	6.500	9.800	5.600	9.300
	n = 96	55.400	62.900	49.200	52.500
ECON	–	3.000	9.800	2.100	4.500
EDCON	–	3.300	10.300	2.500	5.400
EDMOV	–	2.700	8.500	1.700	5.000
$MOV	Character string to be transferred = 0	4.400	12.300	3.400	5.600
	Character string to be transferred = 32	14.000	21.900	11.400	13.300
BXCH	n = 1	6.200	7.900	5.500	7.300
	n = 96	67.300	71.400	47.300	49.300
SWAP	–	2.400	2.700	1.900	2.200
GOEND	–		0.700		0.500
DI	–	2.100	4.000	1.500	1.800
EI	–	3.600	6.300	3.000	3.300
IMASK	–	11.800	20.500	7.200	10.500
IRET	–		1.400		1.000
RSF X n	n = 1	5.900	12.500	3.700	5.600
	n = 96	12.900	19.300	10.700	12.400
RSF Y n	n = 1	5.100	11.500	3.400	4.800
	n = 96	8.600	15.300	8.100	8.900
UDCNT1	–	6.200	16.400	5.100	12.300
UDCNT2	–	6.300	16.800	5.400	12.500
TTMR	–	4.500	9.500	3.400	5.400
STMR	–	7.800	21.400	5.800	12.500
ROTC	–	20.900	21.500	8.000	9.400
RAMP	–	6.700	14.600	5.200	8.400
SPD	–	5.400	14.800	4.900	11.200
PLSY	–	10.500	10.500	7.900	7.900
PWM	–	10.100	10.100	7.500	7.500
MTR	–	14.700	25.100	9.400	10.000
BKAND (s1, s2, d, n)	n = 1	9.000	11.700	8.300	11.000
	n = 96	60.600	66.400	43.800	47.300
BKOR (s1, s2, d, n)	n = 1	7.900	14.000	7.700	9.500
	n = 96	60.700	66.500	44.300	45.800
BKXOR (s1, s2, d, n)	n = 1	8.800	13.800	7.300	9.200
	n = 96	61.300	66.300	43.800	45.800
BKXNR (s1, s2, d, n)	n = 1	8.400	13.900	7.600	8.900
	n = 96	60.900	66.700	43.900	45.300
BSFR (d, n)	n = 1	3.600	9.500	3.200	4.800
	n = 96	6.500	15.900	5.800	7.700

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
BSFL (d, n)	n = 1		3.600	9.300	3.400	5.100
	n = 96		6.300	15.800	6.000	7.900
SFTBR (n1, n2, d)	n1 = 16 / n2 = 1		8.100	21.000	7.500	17.400
	n1 = 16 / n2 = 15		8.100	22.100	7.500	17.300
SFTBL (n1, n2, d)	n1 = 16 / n2 = 1		8.100	21.000	7.500	17.400
	n1 = 16 / n2 = 15		8.100	22.100	7.500	17.300
SFTWR (n1, n2, d)	n1 = 16 / n2 = 1		6.200	13.100	4.500	8.700
	n1 = 16 / n2 = 15		6.100	13.100	4.600	8.800
SFTWL (n1, n2, d)	n1 = 16 / n2 = 1		6.200	13.100	4.500	8.700
	n1 = 16 / n2 = 15		6.100	13.100	4.600	8.800
BSET (d, n)	n = 1		2.800	3.100	2.500	2.800
	n = 15		2.800	3.100	2.500	2.800
BRST (d, n)	n = 1		2.800	3.100	2.500	2.800
	n = 15		2.800	3.100	2.500	2.800
TEST	executed		4.700	6.100	3.700	4.800
DTEST	executed		4.700	6.100	3.700	4.800
BKRST (s, n)	n = 1		4.300	5.700	3.700	4.100
	n = 96		6.200	10.000	5.100	6.000
SER (s1, s2, d, n)	n = 1	all match	4.800	5.300	4.200	4.600
		none match	4.700	5.300	4.200	4.600
	n = 96	all match	33.200	35.900	25.900	26.300
		none match	33.200	35.900	25.900	26.300
DSER (s1, s2, d, n)	n = 1	all match	6.500	9.000	5.400	5.700
		none match	6.500	9.000	5.500	5.900
	n = 96	all match	54.800	57.500	41.200	41.800
		none match	54.700	57.500	41.200	41.800
DSUMS (s, d)	s = 0		3.400	3.700	3.200	3.700
	s = FFFFFFFFH		3.400	3.700	3.200	3.700
DECO (s, d, n)	n = 2		6.000	10.700	5.300	6.900
	n = 8		9.500	16.700	6.800	7.800
ENCO (s, d, n)	n = 2	M1 = ON	5.400	6.900	4.700	5.100
		M4 = ON	5.300	6.600	4.600	5.000
	n = 8	M1 = ON	10.700	14.000	9.000	10.000
		M256 = ON	7.000	11.100	5.100	6.100
DIS (s, d, n)	n = 1		4.600	7.000	3.800	4.600
	n = 4		4.900	7.300	4.000	5.000
UNI (s, d, n)	n = 1		5.000	7.300	3.500	4.800
	n = 4		5.700	8.300	4.000	5.100
NDIS	executed		11.200	15.200	11.000	13.200
NUNI	executed		10.600	12.700	7.300	13.200
WTOB (s, d, n)	n = 1		5.400	8.100	4.400	5.800
	n = 96		38.400	40.900	28.200	29.300
BTOW (s, d, n)	n = 1		5.300	8.200	4.600	5.500
	n = 96		31.700	34.200	22.800	23.800
MAX (s, d, n)	n = 1		5.400	11.900	4.000	6.100
	n = 96		34.200	41.100	24.700	27.000
MIN (s, d, n)	n = 1		6.100	12.000	4.000	6.000
	n = 96		32.900	39.300	26.500	28.300
DMAX (s, d, n)	n = 1		6.000	14.800	4.800	8.100
	n = 96		61.100	69.500	47.100	49.600

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)	Processing time (μs)			
		L02CPU		L26CPU-BT	
		Min.	Max.	Min.	Max.
DMIN (s, d, n)	n = 1	6.000	14.800	4.300	5.900
	n = 96	57.000	67.000	45.400	47.400
SORT (s1, n, s2, d1, d2)	n = 1	6.800	13.700	5.600	8.800
	n = 96	34.400	48.600	27.200	31.800
DSORT (s1, n, s2, d1, d2)	n = 1	6.800	14.300	5.600	8.200
	n = 96	41.800	57.500	33.200	39.000
WSUM (s, d, n)	n = 1	5.000	7.300	4.200	5.500
	n = 96	28.100	30.700	21.300	22.300
DWSUM (s, d, n)	n = 1	6.100	11.300	4.800	6.100
	n = 96	56.200	62.100	42.700	44.000
MEAN (s, d, n)	n = 1	4.400	10.400	3.900	7.800
	n = 96	16.100	24.500	12.900	18.000
DMEAN (s, d, n)	n = 1	6.000	12.500	5.300	9.950
	n = 96	34.000	42.000	23.000	28.800
NEXT	–	0.940	1.400	0.770	1.200
BREAK	–	3.500	10.200	3.100	7.600
RET	Return to original program	2.900	8.800	1.600	2.600
	Return to other program	3.200	10.500	2.000	3.100
FCALL Pn	Internal file pointer	3.600	3.800	2.700	3.600
	Common pointer	5.300	13.500	3.600	5.100
FCALL pn s1 to s5	–	20.900	30.300	16.500	18.600
ECALL * pn *: Program name	–	72.700	109.000	65.900	77.600
ECALL * pn s1 to s5 *: Program name	–	101.400	141.400	91.800	105.000
EFCALL * pn *: Program name	–	72.800	109.600	66.200	78.100
EFCALL * pn s1 to s5 *: Program name	–	101.900	141.500	78.800	91.600
XCALL	–	5.200	14.600	3.700	5.200
COM CCOM	When selecting I/O refresh only	8.400	14.600	12.600	17.200
	When selecting CC-Link refresh only (Master station side)	10.500	29.400	10.100	22.000
	When selecting CC-Link refresh only (Local station side)	10.500	29.400	10.100	22.000
	When selecting intelli auto refresh only	7.900	14.400	7.400	11.900
	When selecting communications with display unit	29.700	79.900	26.800	60.700
	When selecting communication with peripheral device	9.500	32.800	9.200	25.200
FIFW	Number of data points = 0	4.200	6.700	3.200	4.600
	Number of data points = 96	4.400	6.800	3.300	3.800
FIFR	Number of data points = 0	5.100	7.400	3.800	4.400
	Number of data points = 96	36.100	38.800	24.800	25.700
FPOP	Number of data points = 0	4.900	7.500	3.800	5.300
	Number of data points = 96	5.000	7.500	3.700	5.400
FINS	Number of data points = 0	5.400	7.500	3.700	5.300
	Number of data points = 96	5.000	7.400	3.700	5.300
FDEL	Number of data points = 0	5.700	8.300	4.200	5.800
	Number of data points = 96	36.900	39.300	25.400	25.900
FROM (n1, n2, d, n3)	n3 = 1	11.600	31.000	10.700	23.600
	n3 = 1000	403.900	432.900	390.900	410.200
DFRO (n1, n2, d, n3)	n3 = 1	13.300	35.400	12.600	26.700
	n3 = 500	405.000	434.600	390.900	410.200

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)	Processing time (μs)			
		L02CPU		L26CPU-BT	
		Min.	Max.	Min.	Max.
TO (n1, n2, s, n3)	n3 = 1	11.200	28.400	9.600	21.300
	n3 = 1000	381.500	410.900	372.500	390.800
DTO (n1, n2, s, n3)	n3 = 1	12.500	33.900	12.000	25.700
	n3 = 500	379.800	410.400	372.500	390.800
LEDR	No display --> no display	2.400	2.600	1.900	2.000
	LED instruction execution --> no display	32.700	50.600	24.400	35.800
BINDA (s, d)	s = 1	5.000	7.300	4.300	5.600
	s = -32768	7.400	9.800	6.500	8.000
DBINDA (s, d)	s = 1	5.600	8.300	4.900	6.300
	s = -2147483648	10.500	12.900	9.600	11.000
BINHA (s, d)	s = 1	4.500	6.900	3.700	5.200
	s = FFFFH	4.500	6.900	3.700	5.200
DBINHA (s, d)	s = 1	5.000	7.600	4.600	6.000
	s = FFFFFFFFH	5.000	7.600	4.600	6.000
BCDDA (s, d)	s = 1	4.300	6.700	3.600	5.000
	s = 9999	4.800	7.100	4.100	5.400
DBCDDA (s, d)	s = 1	4.900	7.200	4.000	5.500
	s = 99999999	5.700	8.300	4.900	6.300
DABIN (s, d)	s = 1	5.800	10.100	5.600	7.800
	s = -32768	5.800	10.100	5.600	7.800
DDABIN (s, d)	s = 1	8.300	12.600	8.100	10.500
	s = -2147483648	8.300	12.600	8.100	10.500
HABIN (s, d)	s = 1	4.500	8.800	4.400	6.500
	s = FFFFH	4.500	8.800	4.400	6.500
DHABIN (s, d)	s = 1	5.500	10.000	5.300	7.700
	s = FFFFFFFFH	5.500	10.000	5.300	7.700
DABCD (s, d)	s = 1	4.500	8.700	4.300	6.300
	s = 9999	4.500	8.700	4.300	6.300
DDABCD (s, d)	s = 1	5.500	9.800	5.500	7.500
	s = 99999999	5.500	9.800	5.500	7.500
COMRD	–	65.700	65.700	50.900	51.200
LEN	1 character	3.900	7.800	3.600	5.500
	96 characters	19.700	23.900	16.800	18.700
STR	–	7.500	16.700	6.600	10.400
DSTR	–	10.200	19.700	9.600	11.500
VAL	–	9.800	19.900	8.900	13.000
DVAL	–	12.700	23.900	12.700	16.800
ESTR	–	21.200	43.400	17.900	23.100
EVAL	Decimal point format all 2-digit specification	28.300	41.000	22.500	29.00
	Exponent format all 6-digit specification	28.300	41.000	22.500	29.00
ASC (s, d, n)	n = 1	6.200	17.100	5.400	8.300
	n = 96	30.300	42.100	25.200	28.400
HEX (s, d, n)	n = 1	5.400	16.000	5.400	9.000
	n = 96	42.400	54.900	31.300	35.000
RIGHT (s, d, n)	n = 1	7.400	13.900	7.300	6.600
	n = 96	39.300	45.800	29.200	31.600
LEFT (s, d, n)	n = 1	6.900	13.400	5.900	8.200
	n = 96	39.300	45.800	29.200	31.500
MIDR	–	10.200	16.500	8.100	10.300
MIDW	–	10.700	14.900	8.800	10.200

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
INSTR	No match		20.000	25.600	16.600	18.400
	Match	Head	11.000	16.500	9.100	10.900
		End	53.900	60.000	42.700	44.900
EMOD	–		11.200	15.100	9.600	11.000
EREXP	–		20.400	22.900	18.800	20.100
STRINS (s, d, n)	s = 128 / d = 40 / n = 1		45.300	63.400	35.300	47.600
	s = 128 / d = 40 / n = 48		63.200	81.900	48.600	61.700
STRDEL (s, d, n)	s = 128 / d = 40 / n = 1		39.000	53.500	34.800	44.600
	s = 128 / d = 40 / n = 48		40.800	50.400	29.200	38.100
SIN	single precision		5.000	8.400	4.100	5.700
COS	single precision		5.200	8.000	4.000	5.600
TAN	single precision		6.100	9.200	5.100	6.700
ASIN	single precision		6.900	10.900	5.900	8.500
ACOS	single precision		7.800	11.000	6.700	8.900
ATAN	single precision		4.700	7.300	3.900	6.000
SIND	double precision		9.400	22.300	8.500	13.800
COSD	double precision		10.000	22.300	8.800	14.600
TAND	double precision		12.200	24.900	10.800	16.500
ASIND	double precision		12.800	25.900	11.600	16.600
ACOSD	double precision		12.600	25.900	11.200	16.200
ATAND	double precision		10.500	22.900	9.100	13.800
RAD	single precision		3.000	6.400	2.100	4.300
RADD	double precision		5.200	16.900	3.600	9.200
DEG	single precision		2.900	6.600	2.200	4.400
DEGD	double precision		5.200	16.800	3.800	9.000
SQR	single precision		3.600	7.200	2.600	4.300
SQRD	double precision		6.200	19.100	5.200	11.000
EXP (s, d)	single precision	s = -10	4.700	7.500	3.800	5.600
		s = 1	4.700	7.500	3.800	5.600
EXPD (s, d)	double precision	s = -10	9.300	22.100	8.000	13.500
		s = 1	9.300	22.100	8.000	13.500
LOG (s, d)	single precision	s = 1	4.700	8.800	3.800	6.400
		s = 10	6.300	10.400	5.200	7.700
LOGD (s, d)	double precision	s = 1	8.600	21.100	7.700	12.500
		s = 10	10.200	23.000	9.200	14.300
RND	–		1.500	2.500	0.800	1.800
SRND	–		1.800	2.900	1.100	2.000
BSQR (s, d)	s = 0		2.700	4.400	1.500	3.000
	s = 9999		6.100	12.500	5.100	8.000
BDSQR (s, d)	s = 0		2.700	4.400	1.500	3.000
	s = 99999999		8.500	15.200	7.500	9.900
BSIN	–		9.500	21.500	8.100	14.500
BCOS	–		9.500	21.400	7.800	13.700
BTAN	–		10.400	22.600	9.000	13.300
BASIN	–		11.800	23.600	10.100	12.800
BACOS	–		13.100	23.700	11.100	14.100
BATAN	–		11.100	21.500	9.100	10.900
POW (s1, s2, d)	single precision	s1 = 12.3E+5; s2 = 3.45E+0	9.600	13.300	8.400	10.900
POWD (s1, s2, d)	double precision	s1 = 12.3E+5; s2 = 3.45E+0	18.900	30.600	18.200	26.500

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
LOG10	single precision		6.000	9.600	5.700	8.050
LOG10D	double precision		11.900	22.900	11.100	18.600
LIMIT	–		4.000	4.000	2.400	2.700
DLIMIT	–		4.400	4.400	2.800	3.000
BAND	–		4.500	6.600	2.700	3.800
DBAND	–		4.800	6.900	3.300	4.600
ZONE	–		4.200	6.100	2.600	4.300
DZONE	–		4.700	6.900	3.000	4.600
LDDT =	Comparison of specified date	continuity	7.700	14.200	6.800	10.900
		no continuity	7.700	14.200	6.800	10.900
	Comparison of current date	continuity	6.400	12.800	5.500	9.700
		no continuity	6.400	12.800	5.500	9.700
ANDDT=	not executed			0.160		0.038
	Comparison of specified date	continuity	7.300	14.000	6.500	10.700
		no continuity	7.300	14.000	6.500	10.700
	Comparison of current date	continuity	6.100	12.700	5.300	9.300
		no continuity	6.100	12.700	5.300	9.300
ORDT=	not executed			0.160		0.038
	Comparison of specified date	continuity	7.400	14.400	6.700	10.800
		no continuity	7.400	14.400	6.700	10.800
	Comparison of current date	continuity	6.000	12.800	5.400	9.600
		no continuity	6.000	12.800	5.400	9.600
LDDT <>	Comparison of specified date	continuity	7.700	14.200	6.800	10.900
		no continuity	7.700	14.200	6.800	10.900
	Comparison of current date	continuity	6.400	12.800	5.500	9.700
		no continuity	6.400	12.800	5.500	9.700
ANDDT<>	not executed			0.160		0.038
	Comparison of specified date	continuity	7.300	14.000	6.500	10.700
		no continuity	7.300	14.000	6.500	10.700
	Comparison of current date	continuity	6.100	12.700	5.300	9.300
		no continuity	6.100	12.700	5.300	9.300
ORDT<>	not executed			0.160		0.038
	Comparison of specified date	continuity	7.400	14.400	6.700	10.800
		no continuity	7.400	14.400	6.700	10.800
	Comparison of current date	continuity	6.000	12.800	5.400	9.600
		no continuity	6.000	12.800	5.400	9.600
LDDT>	Comparison of specified date	continuity	7.700	14.200	6.800	10.900
		no continuity	7.700	14.200	6.800	10.900
	Comparison of current date	continuity	6.400	12.800	5.500	9.700
		no continuity	6.400	12.800	5.500	9.700
ANDDT>	not executed			0.160		0.038
	Comparison of specified date	continuity	7.300	14.000	6.500	10.700
		no continuity	7.300	14.000	6.500	10.700
	Comparison of current date	continuity	6.100	12.700	5.300	9.300
		no continuity	6.100	12.700	5.300	9.300
ORDT>	not executed			0.160		0.038
	Comparison of specified date	continuity	7.400	14.400	6.700	10.800
		no continuity	7.400	14.400	6.700	10.800
	Comparison of current date	continuity	6.000	12.800	5.400	9.600
		no continuity	6.000	12.800	5.400	9.600

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
LDDT<=	Comparison of specified date	continuity	7.700	14.200	6.800	10.900
		no continuity	7.700	14.200	6.800	10.900
	Comparison of current date	continuity	6.400	12.800	5.500	9.700
		no continuity	6.400	12.800	5.500	9.700
ANDDT<=	not executed			0.160		0.038
	Comparison of specified date	continuity	7.300	14.000	6.500	10.700
		no continuity	7.300	14.000	6.500	10.700
	Comparison of current date	continuity	6.100	12.700	5.300	9.300
		no continuity	6.100	12.700	5.300	9.300
ORDT<=	not executed			0.160		0.038
	Comparison of specified date	continuity	7.400	14.400	6.700	10.800
		no continuity	7.400	14.400	6.700	10.800
	Comparison of current date	continuity	6.000	12.800	5.400	9.600
		no continuity	6.000	12.800	5.400	9.600
LDDT<	Comparison of specified date	continuity	7.700	14.200	6.800	10.900
		no continuity	7.700	14.200	6.800	10.900
	Comparison of current date	continuity	6.400	12.800	5.500	9.700
		no continuity	6.400	12.800	5.500	9.700
ANDDT<	not executed			0.160		0.038
	Comparison of specified date	continuity	7.300	14.000	6.500	10.700
		no continuity	7.300	14.000	6.500	10.700
	Comparison of current date	continuity	6.100	12.700	5.300	9.300
		no continuity	6.100	12.700	5.300	9.300
ORDT<	not executed			0.160		0.038
	Comparison of specified date	continuity	7.400	14.400	6.700	10.800
		no continuity	7.400	14.400	6.700	10.800
	Comparison of current date	continuity	6.000	12.800	5.400	9.600
		no continuity	6.000	12.800	5.400	9.600
LDDT>=	Comparison of specified date	continuity	7.700	14.200	6.800	10.900
		no continuity	7.700	14.200	6.800	10.900
	Comparison of current date	continuity	6.400	12.800	5.500	9.700
		no continuity	6.400	12.800	5.500	9.700
ANDDT>=	not executed			0.160		0.038
	Comparison of specified date	continuity	7.300	14.000	6.500	10.700
		no continuity	7.300	14.000	6.500	10.700
	Comparison of current date	continuity	6.100	12.700	5.300	9.300
		no continuity	6.100	12.700	5.300	9.300
ORDT>=	not executed			0.160		0.038
	Comparison of specified date	continuity	7.400	14.400	6.700	10.800
		no continuity	7.400	14.400	6.700	10.800
	Comparison of current date	continuity	6.000	12.800	5.400	9.600
		no continuity	6.000	12.800	5.400	9.600
LDTM=	Comparison of specified time	continuity	7.600	14.000	6.700	10.800
		no continuity	7.600	14.000	6.700	10.800
	Comparison of current time	continuity	6.200	12.700	5.400	9.500
		no continuity	6.200	12.700	5.400	9.500

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
ANDTM=	not executed			0.160		0.038
	Comparison of specified time	continuity	7.200	13.900	6.300	10.800
		no continuity	7.200	13.900	6.300	10.800
	Comparison of current time	continuity	5.900	12.500	5.100	9.500
		no continuity	5.900	12.500	5.100	9.500
ORTM=	not executed			0.160		0.038
	Comparison of specified time	continuity	7.300	14.100	6.600	10.800
		no continuity	7.300	14.100	6.600	10.800
	Comparison of current time	continuity	6.000	12.700	5.300	9.500
		no continuity	6.000	12.700	5.300	9.500
LDTM<>	Comparison of specified time	continuity	7.600	14.000	6.700	10.800
		no continuity	7.600	14.000	6.700	10.800
	Comparison of current time	continuity	6.200	12.700	5.400	9.500
		no continuity	6.200	12.700	5.400	9.500
ANDTM<>	not executed			0.160		0.038
	Comparison of specified time	continuity	7.200	13.900	6.300	10.800
		no continuity	7.200	13.900	6.300	10.800
	Comparison of current time	continuity	5.900	12.500	5.100	9.500
		no continuity	5.900	12.500	5.100	9.500
ORTM<>	not executed			0.160		0.038
	Comparison of specified time	continuity	7.300	14.100	6.600	10.800
		no continuity	7.300	14.100	6.600	10.800
	Comparison of current time	continuity	6.000	12.700	5.300	9.500
		no continuity	6.000	12.700	5.300	9.500
LDTM>	Comparison of specified time	continuity	7.600	14.000	6.700	10.800
		no continuity	7.600	14.000	6.700	10.800
	Comparison of current time	continuity	6.200	12.700	5.400	9.500
		no continuity	6.200	12.700	5.400	9.500
ANDTM>	not executed			0.160		0.038
	Comparison of specified time	continuity	7.200	13.900	6.300	10.800
		no continuity	7.200	13.900	6.300	10.800
	Comparison of current time	continuity	5.900	12.500	5.100	9.500
		no continuity	5.900	12.500	5.100	9.500
ORTM>	not executed			0.160		0.038
	Comparison of specified time	continuity	7.300	14.100	6.600	10.800
		no continuity	7.300	14.100	6.600	10.800
	Comparison of current time	continuity	6.000	12.700	5.300	9.500
		no continuity	6.000	12.700	5.300	9.500
LDTM<=	Comparison of specified time	continuity	7.600	14.000	6.700	10.800
		no continuity	7.600	14.000	6.700	10.800
	Comparison of current time	continuity	6.200	12.700	5.400	9.500
		no continuity	6.200	12.700	5.400	9.500
ANDTM<=	not executed			0.160		0.038
	Comparison of specified time	continuity	7.200	13.900	6.300	10.800
		no continuity	7.200	13.900	6.300	10.800
	Comparison of current time	continuity	5.900	12.500	5.100	9.500
		no continuity	5.900	12.500	5.100	9.500

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)		Processing time (μs)			
			L02CPU		L26CPU-BT	
			Min.	Max.	Min.	Max.
ORTM<=	not executed			0.160		0.038
	Comparison of specified time	continuity	7.300	14.100	6.600	10.800
		no continuity	7.300	14.100	6.600	10.800
	Comparison of current time	continuity	6.000	12.700	5.300	9.500
		no continuity	6.000	12.700	5.300	9.500
LDTM<	Comparison of specified time	continuity	7.600	14.000	6.700	10.800
		no continuity	7.600	14.000	6.700	10.800
	Comparison of current time	continuity	6.200	12.700	5.400	9.500
		no continuity	6.200	12.700	5.400	9.500
ANDTM<	not executed			0.160		0.038
	Comparison of specified time	continuity	7.200	13.900	6.300	10.800
		no continuity	7.200	13.900	6.300	10.800
	Comparison of current time	continuity	5.900	12.500	5.100	9.500
		no continuity	5.900	12.500	5.100	9.500
ORTM<	not executed			0.160		0.038
	Comparison of specified time	continuity	7.300	14.100	6.600	10.800
		no continuity	7.300	14.100	6.600	10.800
	Comparison of current time	continuity	6.000	12.700	5.300	9.500
		no continuity	6.000	12.700	5.300	9.500
LDTM>=	Comparison of specified time	continuity	7.600	14.000	6.700	10.800
		no continuity	7.600	14.000	6.700	10.800
	Comparison of current time	continuity	6.200	12.700	5.400	9.500
		no continuity	6.200	12.700	5.400	9.500
ANDTM>=	not executed			0.160		0.038
	Comparison of specified time	continuity	7.200	13.900	6.300	10.800
		no continuity	7.200	13.900	6.300	10.800
	Comparison of current time	continuity	5.900	12.500	5.100	9.500
		no continuity	5.900	12.500	5.100	9.500
ORTM>=	not executed		0.160	0.038		
	Comparison of specified time	continuity	7.300	14.100	6.600	10.800
		no continuity	7.300	14.100	6.600	10.800
	Comparison of current time	continuity	6.000	12.700	5.300	9.500
		no continuity	6.000	12.700	5.300	9.500
SCL (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	12.500	29.200	11.900	23.000
		Point No. 9 < s1 < Point No. 10	13.200	29.100	12.100	23.000
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	12.100	28.900	10.900	22.200
		Point No. 9 < s1 < Point No. 10	13.900	30.900	12.700	23.900
DSCL (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	12.500	29.200	11.900	23.000
		Point No. 9 < s1 < Point No. 10	13.200	29.100	12.100	23.000
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	12.100	28.900	10.900	22.200
		Point No. 9 < s1 < Point No. 10	13.900	30.900	12.700	23.900
SCL2 (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	13.400	29.700	11.800	23.300
		Point No. 9 < s1 < Point No. 10	12.900	29.500	12.100	23.300
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	12.200	29.100	11.000	22.600
		Point No. 9 < s1 < Point No. 10	13.900	30.700	12.600	23.900
DSCL2 (s1, s2, d)	SM750 = ON	Point No. 1 < s1 < Point No. 2	13.400	29.700	11.800	23.300
		Point No. 9 < s1 < Point No. 10	12.900	29.500	12.100	23.300
	SM750 = OFF	Point No. 1 < s1 < Point No. 2	12.200	29.100	11.000	22.600
		Point No. 9 < s1 < Point No. 10	13.900	30.700	12.600	23.900
RSET	Standard RAM		3.500	11.100	2.700	5.900

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

Instruction	Condition (Device)	Processing time (μs)			
		L02CPU		L26CPU-BT	
		Min.	Max.	Min.	Max.
DATE -	No digit increase	9.000	17.900	4.600	7.000
	Digit increase	10.000	19.200	4.600	6.500
SECOND	–	4.600	9.800	2.200	3.400
HOUR	–	4.600	10.300	2.400	4.300
QCDSET	SD memory card to standard ROM	690.800	736.470	1.146.900	1.179.500
	Standard ROM to SD memory card	6.981.400	7.232.070	5.613.900	5.653.500
DATERD	–	4.600	11.200	2.500	4.200
DATEWR	–	6.500	19.300	4.100	8.900
DATE +	No digit increase	10.000	19.400	4.700	6.600
	Digit increase	9.900	19.700	4.600	6.500
S.DATERD	–	7.800	22.500	4.800	7.100
S.DATE +	No digit increase	15.100	34.100	7.400	10.000
	Digit increase	15.000	34.100	7.400	10.000
S.DATE -	No digit increase	13.700	33.600	7.400	10.300
	Digit increase	13.700	33.600	7.500	10.200
PSTOP	–	67.600	104.100	56.600	79.800
POFF	–	66.800	103.600	57.200	79.800
PSCAN	–	67.900	104.800	60.100	79.900
WDT	–	1.600	4.800	1.100	2.400
DUTY	–	4.900	10.100	4.800	9.600
TIMCHK	–	4.100	9.100	3.500	4.700
ZRRDB	File register of standard RAM	2.900	3.300	1.800	2.100
ZRWRB	File register of standard RAM	3.600	3.800	2.400	2.700
ADRSET	–	2.200	4.800	2.100	2.600
ZPUSH	–	8.000	12.000	5.800	7.500
ZPOP	–	8.200	10.900	5.800	6.400
S.ZCOM	When mounting CC-Link module (Master station side)	23.700	48.500	19.300	26.000
	When mounting CC-Link module (Local station side)	23.700	48.500	19.100	26.200
UNIRD (n1, d, n2)	n2 = 1	5.000	14.100	3.700	8.000
	n2 = 16	13.600	22.600	12.200	16.600
TYPERD	–	32.100	67.600	29.500	52.500
TRACE	Start	58.100	58.100	43.800	44.700
TRACER	–	6.100	6.100	4.500	4.500
UMSG	Number of displayed characters = 1	7.300	17.000	7.000	13.500
	Number of displayed characters = 32	16.500	26.300	14.300	21.300
SP.FWRITE	–	81.000	81.800	63.500	64.100
SP.FREAD	–	81.100	81.700	61.600	62.500
SP.DEVST	–	50.100	50.100	39.400	39.400
S.DEVLD	–	12.000	27.600	10.000	17.000

Tab. A-29: *Processing times for instructions other than subset instructions for LCPU*

NOTE *For the instructions for which a leading edge instruction (□P) is not described, the processing time is the same as an ON execution instruction.*

Example: MOVP instruction, WANDP instruction etc.

Table of the time to be added when file register, extended data register, extended link register, and module access device are used

Device Name		Data	Device Specification Location	Processing time (μs)	
				L02CPU	L26CPU-BT
File register (R)	When standard RAM is used	Bit	Source	0.100	0.048
			Destination	0.220	0.038
		Word	Source	0.100	0.048
			Destination	0.100	0.038
		Double word	Source	0.200	0.095
			Destination	0.200	0.086
File register (ZR), Extended data register (D), Extended link register (W)	When standard RAM is used	Bit	Source	0.140	0.057
			Destination	0.280	0.048
		Word	Source	0.140	0.057
			Destination	0.140	0.048
		Double word	Source	0.240	0.105
			Destination	0.240	0.095
Module access device (Un\G□)		Bit	Source	11.700	11.200
			Destination	15.400	15.300
		Word	Source	9.460	9.410
			Destination	19.000	19.000
		Double word	Source	11.000	10.900
			Destination	18.800	18.700

Tab. A-30: *Processing times to be added for instructions other than subset instructions for LCPU*

A.5 Comparison of the CPUs

The following table contains the characteristics, i.e. available devices, processing modes, special relays, etc. of the different CPUs (QCPU, LCPU, A-series CPUs).

A.5.1 Available devices

Device		QCPU			LCPU	A series		
		Q00J, Q00, Q01	**Q00UJ, Q00U, Q01U, Q02U**	**Qn, QnH, QnPH, QnPRH, QnUD(E), QnUD(E)H**	**L02CPU, L26CPU-BT**	**AnU**	**AnA**	**AnN**
Number of inputs/outputs [9]		Q00J: 256 Q00: 1024 Q01: 1024	Q00UJ: 256 Q00U: 1024 Q01U: 1024 Q02U: 2048	4096	L02CPU: 1024 L26CPU-BT: 4096	A2U : 512 A2U-S1: 1024 A3U: 2048 A4U : 4096	A2A : 512 A2A-S1:1024 A3A: 2048	A1N: 256 A2N: 512 A2N-S: 1024 A3N: 2048
Number of I/O device points [8]		2048 [1]	8192 [1]	8192 [1]	8192 [1]	8192	Same with I/O devices points of each CPU	
Internal relays		8192 [1]	8192 [1]	8192 [1]	8192 [1]	Total 8192	Total 8192	Total 2048
Latch relays		2048 [1]	8192 [1]	8192 [1]	8192 [1]			
Step relays	Sequence program	—			—			—
	SFC	2048 [6]	8192	8192	8192	—		
Annunciators		1024 [1]	2048 [1]	2048 [1]	2048 [1]	2048	2048	256
Edge triggered relays		1024 [1]	2048 [1]	2048 [1]	2048 [1]	—		
Link relays		2048 [1]	8192 [1]	8192 [1]	8192 1	8192	4096	1024
Special link relays		1024	2048 [1]	2048 [1]	2048 [1]	56	56	56
Timers		512 [1]	2048 [1]	2048 [1]	2048 [1]	Total 2048	Total 2048	Total 256
Retentive Timers		0 [1]	0 [1]	0 [1]	0 [1]			
Counters		512 [1]	1024 [1]	1024 [1]	1024 1	1024	1024	256
Data registers		11136 [1]	12288 [1]	12288 [1]	12288 1)	8192	6144	1024
Link registers		2048 [1]	8192 [1]	8192 [1]	8192 1)	8192	4096	1024
Special link Registers		1024 [1]	2048 [1]	2048 [1]	2048 [1]	56	56	56
Function inputs		16 (FX0 to FXF) [7]			16 (FX0 to FXF) [7]	—		
Function output		16 (FX0 to FXF) [7]			16 (FY0 to FYF) [7]	—		
Special relays		1000	2048	2048	2048	256	256	256
Function registers		5 (FD0 to FD4)			5 (FD0 to FD15)	—		
Special registers		1000	2048	2048	2048	256	256	256
Direct access link devices		Designated by **J□\□**			—	—		
Direct access special devices		Designated by **U□\G□**			Designated by **U□\G□**	—		

Tab. A-31: *Device comparison*

Device		QCPU			LCPU	A series		
		Q00J, Q00, Q01	Q00UJ, Q00U, Q01U, Q02U	Qn, QnH, QnPH, QnPRH, QnUD(E), QnUD(E)H	L02CPU, L26CPU-BT	AnU	AnA	AnN
Index registers	Z	10 (Z0 to Z15)	Other than Universal model QCPU: 16 (Z0 to Z15) Universal model QCPU: 20 (Z0 to Z19)		20 (Z0 to Z19)	7 (Z, Z1 to Z6)	7 (Z, Z1 to Z6)	1 (Z)
	V [2]	—			—	7 (V, V1 to V6)	7 (V, V1 to V6)	1 (V)
File registers		32768 per block (R0 to R32767) [5]	32768 per block (R0 to R32767) [10]		32768 per block (R0 to R32767)	8192 per block (R0 to R8191)	8192 per block (R0 to R8191)	8192 per block (R0 to R8191)
Accumulators [3]		—			—	2	2	2
Nesting		15	15	15	15	8	8	8
Pointer		300	512	4096	4096	256	256	256
Interrupt pointers		128	256	48	256	32	32	32
SFC blocks		126 [6]	320	320	320	—		
SFC transition devices		—	512	512	512	—		
Decimal constants		K-2147483648 to K2147483647						
Hexadecimal constants		H0 to HFFFFFFFF						
Real number constants [6]		E±1.17549-38 to E±3.40282+38				—		
Character strings		„QnA CPU“, „ABCD“ [4]				—		

Tab. A-31: *Device comparison*

[1] The number of device points can be changed via parameters.
[2] CPU uses V for the edge relay.
[3] Instructions using accumulators with the AnN, AnA, and AnU CPUs have different formats than those with the QCPUs.
[4] Can only be used by the $MOV instruction with the Q00JCPU, Q00CPU and Q01CPU.
[5] The Q00JCPU does not have file registers.
[6] Applicable to products with the first 5 digits of the serial number 04122 or higher (Q00JCPU, Q00CPU, and Q01CPU).
[7] Each 5 points of FX0 to FX4 and FY0 to FY4 can be used on the programs.
[8] The number of points that can be used on the programs
[9] The number of accessible points to actual I/O modules
[10] The Q00UJCPU does not have file registers.

A.5.2 I/O control modes

I/O control mode			Type of CPU				
			QCPU	LCPU	AnUCPU	AnACPU	AnNCPU
Refresh mode			●	●	●	●	●2)
	Direct input/output mode	Partial refresh instructions	●	●	●	●	●
		Dedicated instructions1)	—	—	●	●	—
		Direct access inputs	●	●	—	—	—
		Direct access outputs	●	●	—	—	—
Direct mode			—	—	—	—	●2)

Tab. A-32: *I/O Control mode*

1 The DOUT, DSET, and SRST instructions are dedicated instructions for direct access outputs. There are no dedicated instructions for direct access inputs.
2 With the AnN CPU refresh mode and direct mode are switched over via DIP switch.

A.5.3 Data types

Set Data		QCPU	LCPU	AnUCPU	AnACPU	AnNCPU
Bit Data	Bit device	●	●	●	●	●
	Word device	● (Bit designation required)	● (Bit designation required)	—	—	—
16-bit word data	Bit device	● (Digit designation required)	● (Digit designation required)	● (Digit designation required)	● (Digit designation required)	● (Digit designation required)
	Word device	●	●	●	●	●
32-bit word data	Bit device	● (Digit designation required)	● (Digit designation required)	● (Digit designation required)	● (Digit designation required)	● (Digit designation required)
	Word device	●	●	●	●	●
Real number data		●1)	●1)	●	●	—
Character string data		●2)	●2)	—	—	—

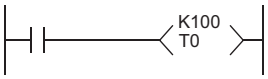
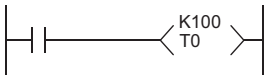
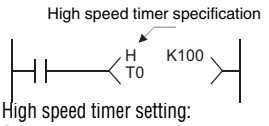
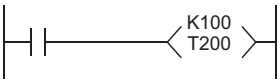
Tab. A-33: *Data that can be used by instructions*

1 Applicable to products with the first 5 digits of the serial number 04122 or higher (Q00JCPU, Q00CPU, and Q01CPU).
2 Character string data can be used in the Q00JCPU, Q00CPU and Q01CPU in combination with the $MOV instruction only.

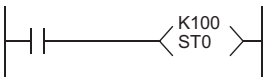
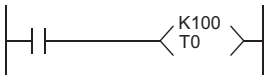
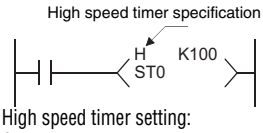
NOTE *Refer to section 3.5 for detailed information on data types.*

A.5.4 Timer comparison

Timer functions

Name	Function			QCPU/LCPU	AnUCPU [1)]	AnACPU [1)]	AnNCPU [1)]
Low-speed timer	Measurement unit			100 ms (default) Change of measurement unit at the parameter is enabled. QCPU/LCPU 1 to 1000 ms (1 ms unit)	Fixed at 100 ms		
	Designation method						
	Programming (GX IEC Developer)	TIMER_M (regular/dedicated timers)	Setting value designation and timer start	● [2)]	●	●	●
		TIMER_VALUE_M (dedicated timers only)	Setting value designation	● [2)]	●	●	●
		TIMER_START_M (dedicated timers only)	Timer start	● [2)]	●	●	●
	Programming (GX Works2)	Setting value designation and timer start		OUT Tn Setting value	OUT Tn Setting value		
High-speed timer	Measurement unit			10 ms (default) Change of measurement unit at the parameter is enabled. QnUCPU/LCPU: 0.01 to 100ms (0.01 ms unit) QCPU(Other than QnUCPU): 0.1 to 100 ms (0.1 ms unit)	Fixed at 100 ms		
	Designation method			 High speed timer setting: Conducted by sequence program	 High speed timer setting: Conducted at parameters		
High-speed timer	Programming (GX IEC Developer)	TIMER_M (regular/dedicated timers)	Setting value designation and timer start	● [2)]	●	●	●
		TIMER_VALUE_M (dedicated timers only)	Setting value designation	● [2)]	●	●	●
		TIMER_START_M (dedicated timers only)	Timer start	● [2)]	●	●	●
	Programming (GX Works2)	Setting value designation and timer start		OUT Tn Setting value	OUT Tn Setting value		

Tab. A-34: *Timer comparison*

Name	Function			QCPU/LCPU	AnUCPU [1]	AnACPU [1]	AnNCPU [1]
Retentive low-speed timer	Measurement unit			Same measurement unit as low speed timer	Fixed at 100 ms		
	Designation method						
	Programming (GX IEC Developer)	TIMER_H_M (regular/dedicated timers)	Setting value designation and timer start	● [2]	●	●	●
		TIMER_VALUE_M (dedicated timers only)	Setting value designation	● [2]	●	●	●
		TIMER_START_M (dedicated timers only)	Timer start	● [2]	●	●	●
	Programming (GX Works2)	Setting value designation and timer start		OUT Tn Setting value	OUT Tn Setting value		
Retentive High-speed timer	Measurement unit			Same measurement unit as high speed timer	—		
	Designation method			 High speed timer setting: Conducted by sequence program	—		
	Programming (GX IEC Developer)	TIMER_H_M (regular/dedicated timers)	Setting value designation and timer start	● [2]	—	—	—
		TIMER_VALUE_M (dedicated timers only)	Setting value designation	● [2]	—	—	—
		TIMER_START_M (dedicated timers only)	Timer start	● [2]	—	—	—
	Programming (GX Works2	Setting value designation and timer start		OUTH STn Set value	—	—	—
Setting range for setting value				1 to 32767	1 to 32767		
Processing of setting value 0				ON momentarily	No maximum (does not time out)		
Index qualification	Contact			Enabled (Z0 and Z1 usable only)	Capable		Not capable
	Coil			Enabled (Z0 and Z1 usable only)	Not capable		Not capable
	Setting value			Enabled (Z0 to Z15 are usable) [1]	Not capable		Not capable
	Current value			Enabled (Z0 to Z15 are usable) [1]	Capable		Capable
Update processing for current value				At OUT Tn instruction execution	After END processing		
Contact ON/OFF processing							

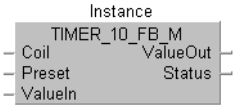
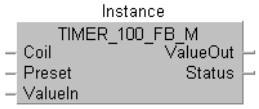
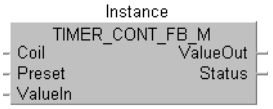
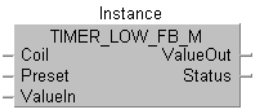
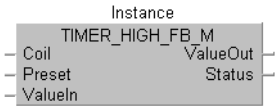
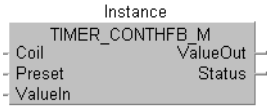
Tab. A-34: *Timer comparison*

[1] The initial number for the different timers must be specified in the GX IEC Developer in the dialogbox “PLC Parameter - T/C Range“

[2] GX IEC Developer does not support LCPU

[3] The Q00J/Q00/Q01CPU can use Z0 to Z9.
The Universal model QCPU/LCPU can use Z0 to Z19.

Timer function blocks in the GX IEC Developer

Name	Function block	Type of CPU				
		QCPU	LCPU	AnU	AnA	AnN
10 ms timer		●	—	●	●	●
100 ms timer		●	—	●	●	●
retentive timer		●	—	●	●	●
Low-speed timer		●	—	—	—	—
High-speed timer		●	—	—	—	—
retentive High-speed timer		●	—	—	—	—

Timer function blocks (legend)

Term in function block	Meaning		Indication of regular timers	Indication of retentive timers
Coil	Coil	Execution condition for timer process	TC	STC
Preset	Setting value	—	TValue	TValue
ValueIn	Initial value	Default: 0	—	—
ValueOut	Actual value	—	TN	STN
Status	Contact	Output contact is switched after time	TS	STS

Assign the function block to the instance label specified in the header and assign the input and output variables.

NOTE

Cautions on using timers

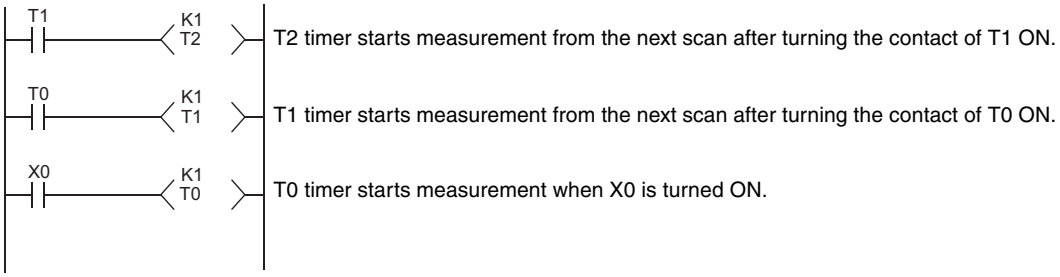
During the execution of the OUT T □ instruction, the present value of the timers is updated and the contact is switched ON or OFF. If the present value of the timer is larger than or equal to the set value when the timer coil is turned ON, the contact of that timer is turned ON.

In a program, in which the operation of a timer is started by another timer, the instruction for the timer which is started later must be processed first. For example, if the contact of T1 activates the coil of T2, the instruction for T2 must placed in the program before the instruction for T1.

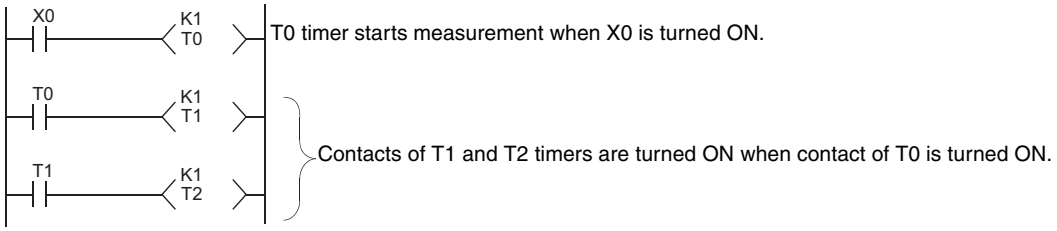
By doing so, it is prevented that all timer contact are turned ON at the same scan. This can happen if the instruction for a timer, which starts another timer is processed first and the setting value for high speed timers is smaller than the scan time or the setting value for slow speed timers is „1“.

Example

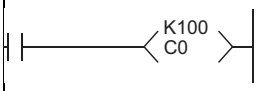
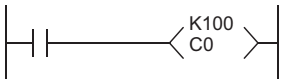
For timers T0 to T2, the program is created in the order the timer operates later.



For timers T0 to T2, the program is created in the order of timer operation.



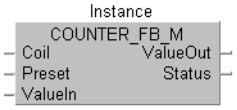
A.5.5 Comparison of counters

Function			Type of CPU			
			QCPU/LCPU	AnU	AnA	AnN
Programming (GX IEC Developer)	Counter_M	Setting value designation and counter start	● [2)]	●	●	●
	Counter_Start_M	Setting value designation	● [2)]	●	●	●
	Counter_Value_M	Counter start	● [2)]	●	●	●
Programming (GX Works2)	OUT Cn Set value	Setting value designation and counter start	● [2)]	●	●	●
Designation method						
Index qualification	Contact		Enabled (Z0 and Z1 usable only)	Capable		Not capable
	Coil		Enabled (Z0 and Z1 usable only)	Not capable		Not capable
	Setting value		Not capable	Not capable		Not capable
	Current value		Enabled (Z0 to Z15 are usable) [1)]	Capable		Capable
Update processing for current value			At OUT Cn instruction execution	After END processing		
Contact ON/OFF processing						

Tab. A-35: *Counter functions*

[1] GX IEC Developer does not support LCPU
[2] The Q00J/Q00/Q01CPU can use Z0 to Z9.
The Universal model QCPU/LCPU can use Z0 to Z19.

Counter function blocks

Name	Function blocks	Type of CPU				
		QCPU	LCPU	AnU	AnA	AnN
Counter		—	—	●	●	●

Tab. A-36: *Counter function blocks*

Term in function block	Meaning		Indication of counter
Coil	Coil	Execution condition for counter	CC
Preset	Setting value		CValue
ValueIn	Initial value	Default: 0	—
ValueOut	Current value		CN
Status	Contact	Output contact is switched after the function block is processed.	CS

Tab. A-37: *Counter function blocks (legend)*

A.5.6 Comparison of display instructions

Instruction	QCPU/LCPU	AnUCPU	AnACPU	AnNCPU
PR [1)]	When SM701is OFF: Output continued until 00H encountered When SM701 is ON: 16 characters output	When M9049 is OFF: Output continued until 00H encountered When M9049 is ON: 16 characters output		
PRC [1)]	When SM701 is OFF: 32 character comment output When SM701 is ON: Upper 16 characters output	16-character comment output		

Tab. A-38: *Comparison of display instructions*

1 These instructions are not available for a Q00JCPU, Q00CPU or Q01CPU.

A.5.7 QCPU, LCPU instructions whose designation format has been changed

Since QCPU, LCPU do not use accumulators (A0, A1), the format of the AnU, AnA, and AnN CPU instructions that use accumulators has changed.

Function	QCPU/LCPU		AnU CPU / AnA CPU / AnN CPU	
	Instruction format	Remark	Instruction format	Remark
16-bit rotation to right	ROR (d, n)	D: Rotation data	ROR (n)	Rotation data is set at A0
	RCR (d, n)	D: Rotation data The carry flag uses SM700	RCR (n)	Rotation data is set at A0 Carry flag uses M9012
16-bit rotation to left	ROL (d, n)	D: Rotation data	ROL (n)	Rotation data is set at A0
	RCL (d, n)	D: Rotation data The carry flag uses SM700	RCL (n)	Rotation data is set at A0 Carry flag uses M9012
32-bit rotation to right	DROR (d, n)	D: Rotation data	DROR (n)	Rotation data is set at A0 and A1
	DRCR (d, n)	D: Rotation data The carry flag uses SM700	DRCR (n)	Rotation data is set at A0 and A1 Carry flag uses M9012
32-bit rotation to left	DROL (d, n)	D: Rotation data	DROL (n)	Rotation data is set at A0 and A1
	DRCL (d, n)	D: Rotation data The carry flag uses SM700	DRCL (n)	Rotation data is set at A0 and A1 Carry flag uses M9012
16-bit data search	SER (s1, s2, d, n)	Search results are stored at the d and d+1 devices	SER (s1, s2, n)	Search results stored at A0 and A1
32-bit data search	DSER (s1, s2, d, n)	Search results are stored at the d and d+1 devices	DSER (s1, s2, n)	Search results stored at A0 and A1 Carry flag uses M9012
16-bit data bit check	SUM (s, d)	Check results are stored at the d device	SUM (s)	Check results stored at A0
32-bit data bit check	DSUM (s, d)	Check results are stored at the d device	DSUM (s)	Check results stored at A0
Partial refresh	RFS (s, n)	Added dedicated instruction	SEG (d, n)	Only when M9052 is ON
8 character ASCII conversion	$MOV ((character string), d)		ASC ((character string), d)	
Carry flag set	SET (SM700)	No dedicated instruction	STC	
Carry flag reset	RST (SM700)	No dedicated instruction	CLC	
Jump to END instruction	GOEND	Added dedicated instruction	CJ (P255)	P255: END instruction designation
CHK instruction [1)]	CHKST CHK	Added CHKST instruction	CJ (Pn) CHK (P254)	

Tab. A-39: *Instructions whose expression has changed*

[1] These instructions are not available for a Q00JCPU, Q00CPU or Q01CPU.

A.5.8 AnACPU and AnUCPU dedicated instructions

Method of expression of dedicated instructions

Dedicated instructions based on the LEDA, LEDB, LEDC, SUB, and LEDR instructions, that are used with the AnACPU or AnUCPU have been changed for the same format as the basic instructions and the application instructions for the QCPU, LCPU.

The instructions that cannot be converted due to the absence of the corresponding instructions in the QCPU, LCPU are converted into OUT SM1255/OUT SM999 (for the Q00J/Q00/Q01CPU).

The instructions that have been converted into OUT SM1255/OUT SM999 should be replaced by other instructions or deleted.

QCPU	AnUCPU/AnACPU
Instruction name (s, d, n)	LEDA(B) (instruction name) LEDC/SUB (s) LEDC/SUB (d) LEDC/SUB (n) LEDR

Tab. A-40: *Method of expression of dedicated instruction*

Dedicated instructions whose names have been changed

Dedicated instructions for the AnUCPU or AnACPU which have the same instruction name as is used for basic instructions and application instructions have undergone name changes in the QCPU, LCPU.

Function	QCPU/LCPU	AnUCPU/AnACPU
Floating point addition	E+	ADD
Floating point subtraction	E–	SUB
Floating point multiplication	Ex	MUL
Floating point division	E/	DIV
Data dissociation	NDIS	DIS
Data association	NUNI	UNI
Updating check patterns	CHKCIR [1)], CHKEND [1)]	CHK, CHKEND

Tab. A-41: *Dedicated instructions with changed names*

[1] Not available on Q00J/Q00/Q01CPU/Universal model QCPU/LCPU.

A.6 Table of special relays

Special relays (SM) are internal relays the application of which is fixed in the PLC. Therefore, they cannot be used like other internal relays in a sequence program. However, some of them can be set ON or OFF in order to control the CPU.

The table below describes the meanings of the headings in the following tables:

Item	Meaning
Number	Indicates the number of the special relay.
Name	Indicates the name of the special relay.
Meaning	Contains the function of the special relay in brief.
Description	Contains a detailed description of the special relay.
Set by (if set)	Indicates whether the special relay is set by the system or the user. <Set by> S : Set by the system U : Set by the user (using a program, programming tool, GOT, or test operation from other external devices) S/U : Set by the system or user Is indicated only if the setting is done by the system. <if set> END processing : Set during END processing Initial : Set during initial processing (Power ON, STOP->RUN) Status change : Set after status change Error : Set after error Instruction execution : Set during instruction execution Request : Set for user request (through SM, etc.) When system is switched : Set when the system is switched (between the control system and the standby system)
ACPU M9□□□	Indicates special relay M9□□□ corresponding to the ACPU (Change and notation when contents changed. Incompatible with the Q00J/Q00/Q01 and QnPRH.) Items indicated as "New" were newly added to the QCPU or LCPU.
Valid for:	Indicates the corresponding CPU: • QCPU: All the System Q CPU modules • Q00J/Q00/Q01: Basic model QCPU • Qn(H): High Performance model QCPU • QnPH: Process CPU • QnPRH: Redundant CPU • QnU: Universal model QCPU • Q00UJ/Q00U/Q01U: Q00UJCPU, Q00UCPU, and Q01UCPU • LCPU: All the L series CPU modules • CPU module model: Only the specified model (Example: Q02UCPU, L26CPU-BT)

For detailed information on the following topic refer to the manuals:

- Networks → Manuals for each network module
- SFC → Programming Manual (SFC)

NOTE *Do not change the values of special relay set by system using a program or by test operation. Doing so may result in system down or communication failure.*

A.6.1 Diagnostic information

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM0	Diagnostic errors	OFF: No error ON: Error	• ON if diagnosis results show error occurrence (Also turns on if an error is detected by an annunciator or the CHK instruction.) • Stays ON subsequently even if normal operations restored.	S (Error)	New	Qn(H) QnPH QnPRH
			• This relay turns on if an error is detected by diagnostics. (Also turns on if an error is detected by an annunciator.) • This relay remains on even after the system returns to normal.			Q00J/ Q00/ Q01 QnU LCPU
SM1	Self-diagnostic error	OFF: No self-diagnosis errors ON: Self-diagnosis	• Comes ON when an error occurs as a result of self-diagnosis. (Remains OFF if an error is detected by an annunciator or the CHK instruction.) • Stays ON subsequently even if normal operations restored.	S (Error)	M9008	Qn(H) QnPH QnPRH
			• This relay turns on if an error is detected by self-diagnostics. (Remains off if an error is detected by an annunciator.) • This relay remains on even after the system returns to normal.			Q00J/ Q00/ Q01 QnU LCPU
SM5	Error common information	OFF: No error common information ON: Error common information	When SM0 is ON, ON if there is error common information.	S (Error)	New	QCPU LCPU
SM16	Error individual information	OFF: No error individual information ON: Error individual information	When SM0 is ON, ON if there is error individual information.	S (Error)	New	
SM50	Error reset	OFF → ON: Error reset	Conducts error reset operation.	U	New	
SM51	Battery low latch	OFF: Normal ON: Battery low	ON if battery voltage at CPU or memory card drops below rated value. Stays ON subsequently even after normal operation is restored. Synchronous with BAT. ALARM LED.	S (Error)	M9007	Qn(H) QnPH QnPRH QnU LCPU
			ON if battery voltage at CPU drops below rated value. Stays ON subsequently even after normal operation is restored. Synchronous with ERR. LED.		New	Q00J/ Q00/ Q01
SM52	Battery low	OFF: Normal ON: Battery low	Same as SM51, but goes OFF subsequently when battery voltage returns to normal.	S (Error)	M9006	QCPU LCPU
SM53	AC/DC DOWN detection	OFF: AC/DC DOWN not detected ON: AC/DC DOWN detected	Comes ON when a AC power supply module is used and a momentary power interruption not exceeding 20 ms has occured; reset by turning the power OFF then ON again.	S (Error)	M9005	QCPU
			Comes ON when a AC power supply module is used and a momentary power interruption not exceeding 10 ms has occured; reset by turning the power OFF then ON again.			LCPU
			Comes ON when a DC power supply module is used and a momentary power interruption not exceeding 10 ms has occured; reset by turning the power OFF then ON again.			Q CPU LCPU
SM56	Operation errors	OFF: Normal ON: Operation error	ON when operation error is generated. Stays ON subsequently even if normal operation is restored.	S (Error)	M9011	Q CPU LCPU

Tab. A-42: *Special relays (1): Diagnostic information*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM60	Blown fuse detection	OFF: Normal ON: Module with blown fuse	Comes ON even if there is only one output module with a blown fuse and remains ON even after return to normal. Blown fuse state is checked even for remote I/O station output modules.	S (Error)	M9000	QCPU
SM61	I/O module Verification error	OFF: Normal ON: Error	Comes ON if there is a discrepancy between the actual I/O modules and the registered information when the power is turned on. Remains ON even after return to normal. I/O module verification is also conducted for remote I/O station modules.	S (Error)	M9002	Q CPU LCPU
SM62	Annunciator detection	OFF: Not detected ON: Detected	Goes ON if at least one annunciator F goes ON.	S (Instruction execution)	M9009	
SM80	CHK detection	OFF: Not detected ON: Detected	Goes ON if error is detected by CHK instruction. Stays ON subsequently even after normal operation is restored.	S (Instruction execution)	New	Qn(H) QnPH QnPRH
SM84	Error clear	OFF → ON: Error clear	Turns on to clear an error set to SD84 and SD85.	U		LCPU
SM90	Startup of watchdog timer for step transition (Enabled only when SFC program exists)	OFF: Not startet (watchdog timer reset) ON: Started (watchdog timer started)	Corresponds to SD90	Goes ON when measurement of step transition watchdog timer is commenced. Resets watchdog timer when it goes OFF.	M9108	Qn(H) QnPH QnPRH
SM91			Corresponds to SD91		M9109	
SM92			Corresponds to SD92		M9110	
SM93			Corresponds to SD93		M9111	
SM94			Corresponds to SD94		M9112	
SM95			Corresponds to SD95		M9113	
SM96			Corresponds to SD96		M9114	
SM97			Corresponds to SD97		New	
SM98			Corresponds to SD98			
SM99			Corresponds to SD99			
SM100	Serial communication function in use	OFF: Serial communication is not in use ON: Serial communication is used	Indicates whether the serial communication function in the serial communication setting parameter is selected or not.	S (power on or reset)	New	Q00/Q01 Q00UJ/ Q00U/Q01U Q02U [2]
SM101	Communication protocol status flag	OFF: Protocol for programming devices ON: MC protocol	Indicates whether the device that is communicating via the RS232 interface is using the protocol for progamming devices or the MC protocol.	S (RS232 communication)	New	Q00/Q01 Q00UJ/ Q00U/Q01U Q02U [2]
		Communication with programming tool	Always off (communication with a programming tool)			LCPU
SM110	Protocol error	OFF: No error ON: Error	Turns ON when an abnormal protocol was used to make communication in the serial communication function. Remains ON if the protocol is restored to normal thereafter.	S (Error)	New	Q00/Q01 Q00UJ/ Q00U/Q01U Q02U [2]
SM111	Communication status	OFF: No error ON: Error	Turns ON when the mode used to make communication was different from the setting in the serial communication function. Remains ON if the mode is restored to normal thereafter.	S (Error)		
SM112	Clear error information	ON: Clear special relays and registers	When turned ON, the special relays SM110 and SM111 are reset and the contents of the diagnostic special registers SD110 and SD111 is cleared.	U		
SM113	Overrun error	OFF: No error ON: Error	Turns ON when an overrun error (to much data) occured during the serial communication.	S (Error)		
SM114	Parity error	OFF: No error ON: Error	Turns ON when a parity error occured during the serial communication.	S (Error)		
SM115	Framing error	OFF: No error ON: Error	Turns ON when a framing error occured during the serial communication.	S (Error)		

Tab. A-42: *Special relays (1): Diagnostic information*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM165	Program memory batch transfer execution status	OFF: Completed ON: Not being executed or Not completed	• Turns on when data are written to the program cache memory. • Turns off when program memory batch transfer is completed. • Remains on when data written to the program cache memory are not batchtransferred to the program memory.	S (Status change)	New	QnU [1)] LCPU

Tab. A-42: *Special relays (1): Diagnostic information*

[1] The relevant modules are as follows:
- The Universal model QCPU whose serial number (first five digits) is "10012" or higher.
- Q13UDHCPU, Q26UDHCPU

[2] The module whose first 5 digits of serial No. is "10102" or higher.

A.6.2 System information

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM202	LED off command	OFF → ON: LED off	At change from OFF to ON, the LEDs corresponding to the individual bits at SD202 go off.	U	New	Qn(H) QnPH QnPRH QnU LCPU
SM203	STOP contact	STOP state	Goes ON at STOP state.	S (Status change)	M9042	Q CPU LCPU
SM204	PAUSE contact	PAUSE state	Goes ON at PAUSE state.	S (Status change)	M9041	
SM206	PAUSE enable coil	OFF: PAUSE disabled ON: PAUSE enabled	PAUSE state is entered if this relay is ON when the remote PAUSE contact goes ON.	U	M9040	Q CPU LCPU
SM210	Clock data set request	OFF: Ignored ON: Set request	When this relay goes from OFF to ON, clock data being stored from SD210 through SD213 after execution of END instruction for changed scan is written to the clock device.	U	M9025	Q CPU LCPU
SM211	Clock data error	OFF: No error ON: Error	ON when error is generated in clock data (SD210 through SD213) value and OFF if no error is detected.	S (Request)	M9026	
SM213	Clock data read request	OFF: Ignored ON: Read request	When this relay is ON, clock data is read to SD210 through SD213 as BCD values.	U	M9028	Q CPU LCPU
SM220	CPU No.1 preparation completed	OFF: CPU No.1 preparation uncompleted ON: CPU No.1 preparation completed	Turns on when an access to CPU No.1 from another CPU becomes possible after power-on or reset operation. This relay is used as an interlock for accessing CPU No.1 when the multiple CPU synchronous setting is set to asynchronous.	S (When status changed)	New	QCPU
SM221	CPU No.2 preparation completed	OFF: CPU No.2 preparation uncompleted ON: CPU No.2 preparation completed	Turns on when an access to CPU No.2 from another CPU becomes possible after power-on or reset operation. This relay is used as an interlock for accessing CPU No.2 when the multiple CPU synchronous setting is set to asynchronous.			QnU [7]
SM222	CPU No.3 preparation completed	OFF: CPU No.3 preparation uncompleted ON: CPU No.3 preparation completed	Turns on when an access to CPU No.3 from another CPU becomes possible after power-on or reset operation. This relay is used as an interlock for accessing CPU No.3 when the multiple CPU synchronous setting is set to asynchronous.			
SM223	CPU No.4 preparation completed	OFF: CPU No.4 preparation uncompleted ON: CPU No.4 preparation completed	Turns on when an access to CPU No.4 from another CPU becomes possible after power-on or reset operation. This relay is used as an interlock for accessing CPU No.4 when the multiple CPU synchronous setting is set to asynchronous.			QnU [5]
SM235	Online module change flag	OFF: Online module change is not in progress ON: Online module change in progress	This relay is on during online module change. (for host CPU)	S (During online module change)		QnPH
SM236	Online module change complete flag	OFF: Online module change incomplete ON: Online module change complete	• This relay is on only for one scan after completion of online module change. • This relay can be used only in the scan execution type program. (for host CPU)	S (When online module change is complete)		
SM237	Device range check inhibit flag	OFF: Device range checked ON: Device range not checked	Selects whether to check a device range during execution of the BMOV, FMOV or DFMOV instruction (only when the conditions for subset processing are established).	U	New	QnU [6] LCPU

Tab. A-43: *Special relays (2): System information*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM240	CPU No. 1 reset flag	OFF: No reset ON: CPU 1 has been reset	This flag comes ON when the CPU no. 1 has been reset or has been removed from the base. The other CPUs of the multi-CPU system are also put in reset status.	S (Status change)	New	Q00/Q01 [1] Qn(H) [1] QnPH QnU [7]
		Reset status	Aways off (reset status)			LCPU
SM241	CPU No. 2 reset flag	OFF: No reset ON: CPU 2 has been reset	This flag comes ON when the CPU no. 2 has been reset or has been removed from the base. In the other CPUs of the multi-CPU system the error code 7000 ("MULTI CPU DOWN") will occur.	S (Status change)	New	Q00/Q01 [1] Qn(H) [1] QnPH QnU [7]
SM242	CPU No. 3 reset flag	OFF: No reset ON: CPU 3 has been reset	This flag comes ON when the CPU no. 3 has been reset or has been removed from the base. In the other CPUs of the multi-CPU system the error code 7000 ("MULTI CPU DOWN") will occur.	S (Status change)	New	
SM243	CPU No. 4 reset flag	OFF: No reset ON: CPU 4 has been reset	This flag comes ON when the CPU no. 4 has been reset or has been removed from the base. In the other CPUs of the multi-CPU system the error code 7000 ("MULTI CPU DOWN") will occur.	S (Status change)	New	Qn(H) [1] QnPH QnU [5]
SM244	CPU No. 1 error flag	OFF: No error ON: CPU no. 1 is stopped due to an error	The set flag indicates that an error has occured which has stopped the CPU. The flag goes OFF when the CPU is normal or when an error occurs which will not stop the CPU.	S (Status change)	New	Q00/Q01 [1] Qn(H) [1] QnPH QnU [7] LCPU
SM245	CPU No. 2 error flag	OFF: No error ON: CPU no. 2 is stopped due to an error		S (Status change)	New	Q00/Q01 [1] Qn(H) [1] QnPH QnU [7]
SM246	CPU No. 3 error flag	OFF: No error ON: CPU no. 3 is stopped due to an error		S (Status change)	New	
SM247	CPU No. 4 error flag	OFF: No error ON: CPU no. 4 is stopped due to an error		S (Status change)	New	Qn(H) [1] QnPH QnU [5]
SM250	Max. loaded I/O read	OFF: Ignored ON: Read	When this relay goes from OFF to ON, maximum loaded I/O number is read to SD250.	U	New	Qn(H) QnPH QnPRH
SM254	All stations refresh command	OFF: Refresh the head station only ON: Refresh all stations	• Effective for the batch refresh and the low-speed cycle. • Designate whether to receive arrival stations only or to receive all slave stations in the MELSECNET/H.	U	New	Qn(H) QnPH QnPRH
			• Effective for the batch refresh and the low-speed cycle. • Designate whether to receive arrival stations only or to receive all slave stations in the CC-Link IE controller network.			
			• Effective for the batch refresh and the low-speed cycle. • Specify whether to receive only arrival station or all stations in the MELSECNET/H or CC-Link IE controller network.			QnU
SM255	MELSECNET/10, MELSECNET/H module 1 information	OFF: Operative network ON: Standby network	Goes ON for standby network. (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	Qn(H) QnPH QnPRH
SM256		OFF: Reads ON: Does not read	For refresh from link to CPU (B, W, etc.) indicate whether to read from the link module.	U	New	
SM257		OFF: Writes ON: Does not write	For refresh from CPU to link (B, W etc.) designate whether to write to the link module.	U	New	

Tab. A-43: *Special relays (2): System information*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM260	MELSECNET/10, MELSECNET/H module 2 information	OFF: Operative network ON: Standby network	Goes ON for standby network. (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	Qn(H) QnPH QnPRH
SM261		OFF: Reads ON: Does not read	For refresh from link to CPU (B, W, etc.) indicate whether to read from the link module.	U	New	
SM262		OFF: Writes ON: Does not write	For refresh from CPU to link (B, W etc.) designate whether to write to the link module.	U	New	
SM265	MELSECNET/10, MELSECNET/H module 3 information	OFF: Operative network ON: Standby network	Goes ON for standby network. (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	Qn(H) QnPH QnPRH
SM266		OFF: Reads ON: Does not read	For refresh from link to CPU (B, W, etc.) indicate whether to read from the link module.	U	New	
SM267		OFF: Writes ON: Does not write	For refresh from CPU to link (B, W etc.) designate whether to write to the link module.	U	New	
SM270	MELSECNET/10 module 4 information	OFF: Operative network ON: Standby network	Goes ON for standby network. (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	Qn(H) QnPH QnPRH
SM271		OFF: Reads ON: Does not read	For refresh from link to CPU (B, W, etc.) indicate whether to read from the link module.	U	New	
SM272		OFF: Writes ON: Does not write	For refresh from CPU to link (B, W etc.) designate whether to write to the link module.	U	New	
SM280	CC-Link error	OFF: Normal ON: Error	Goes ON when a CC-Link error is detected in any of the CC-Link modules installed. Goes OFF when normal operation is restored.	S (Status change)	New	Qn(H) QnPH QnPRH
SM310	RS-232 adapter	OFF: Not connected ON: Connected	This relay stores information on whether an RS-232 adapter is connected or not. Connection of an RS-232 adapter is checked at the time of initialization, and if it is connected, this relay turns on. The on/off status set during initialization is held until the power is turned off and on again or the system is reset.	S (Initial)	New	LCPU
SM315	Communication reserved time delay enable flag	OFF: Witout delay ON: With delay	• The usage of this flag is enabled when the time reserved for communication has been set in SD315. • When this flag is turned ON, the END processing is delayed by the time set in SD315 if no communication is performed. The scan time increases by the time set in SD315. • When this flag is turned OFF, the END processing is performed without delay if there is no communication processing.	U	New	Q00J/Q00/ Q01
SM319	Automatic CC-Link start	OFF: Not activated ON: Activated	• This relay indicates whether the CC-Link module is started and all the data are refreshed by the automatic CC-Link start function. • This relay is on when all the data are refreshed by the automatic CC-Link start function. • Then the automatic CC-Link start function is not activated, or when the refresh device range is insufficient, this relay is turned off. (If the refresh device range set for the automatic CC-Link start function is insufficient, all of the refresh is stopped.)	S (Initial processing and status change)	New	LCPU

Tab. A-43: *Special relays (2): System information*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM320	Presence/absence of SFC program	OFF: SFC program absent ON: SFC program present	ON if SFC program is correctly registered, and OFF if not registered. Goes OFF if SFC dedicated instruction is not correct.	S (Initial)	M9100	Q00J/Q00/Q01 [1)] Qn(H) QnPH QnPRH QnU LCPU
SM321	Start/stop SFC program	OFF: SFC program stop ON: SFC program start	Initial value is set at the same value as SM900. (Goes ON automatically if SFC program is present.) SFC program will not execute if this goes OFF prior to SFC program processing. Subsequently, starts SFC program when this goes from OFF to ON. Subsequently, stops SFC program when this goes from ON to OFF.	S (Initial)/U	M9101 format change	
SM322	SFC program start state	OFF: Initial start ON: Restart	Initial value is set at ON or OFF depending on parameters. When OFF, all execution states are cleared from time SFC program was stopped; starts from the initial step of block where the start request was made. When ON, starts from execution block and execution step active at time SFC program was stopped. (ON is enabled only when resumptive start has been designated at parameters.) SM902 is not automatically designated for latch.	S/U (Initial)	M9102 format change	
SM323	Presence/absence of continuous transition for entire block	OFF: Continuous transition not effective ON: Continuous transition effective	When OFF, transition occurs at one scan/one step, for all blocks. When ON, transition occurs continuously for all blocks in one scan. In designation of individual blocks, priority is given to the continuous transition bit of the block. (Designation is checked when block starts.)	U	M9103	Q00J/Q00/Q01 [1)] Qn(H) QnPH QnPRH QnU LCPU
SM324	Continuous transition prevention flag	OFF: When transition is executed ON: When no transition	When continuous transition is effective, goes ON when continuous transition is not being executed; goes OFF when continuous transition is being executed. Normally ON when continuous transition is not effective.	S (Instruction execution)	M9104	
				S (Status change)	New	Q00J/Q00/Q01 [1)] Qn(H) QnPH QnPRH QnU
SM325	Output mode at block stop	OFF: OFF ON: Preserves	When block stops, selects active step operation output. All coil outputs go OFF when OFF. Coil outputs are preserved when ON.	S (Status change)	M9196	Q00J/Q00/Q01 [1)] Qn(H) QnPH QnPRH QnU LCPU
SM326	SFC device clear mode	OFF: Clear device ON: Preserves device	Selects the device status when the stopped CPU is run after the sequence profram or SFC program has been modified when the SFC program exists.	U	New	
SM327	Output during end step execution	OFF: Hold step output turned OFF (cleared) ON: Hold step output held	If this relay is off, the coil output turns off when the step held after transition (SC, SE, or ST) reaches the end step.	S (Initial) U	New	Qn(H) QnPH QnPRH QnU LCPU
				U	New	Q00J/Q00/Q01 [1)]
SM328	Clear processing mode when end step is reached	OFF: Clear processing is performed. ON: Clear processing is not performed.	Select whether clear processing will be performed or not if active steps other than the ones being held exist in the block when the end step is reached. • When this relay turns OFF, all active steps are forcibly terminated to terminate the block. • When this relay is ON, the execution of the block is continued as-is. • If active steps other than the ones being held do not exist when the end step is reached, the steps being held are terminated to terminate the block.		New	Q00J/Q00/Q01 [1)] QnU LCPU

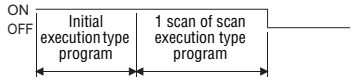
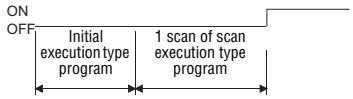
Tab. A-43: *Special relays (2): System information*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM329	Online change (inactive block) status flag	OFF: Not executed ON: Being executed	This relay is on while online change (inactive block) is executed.	S (Status change)	New	QnU [8]
SM330	Operation mode for low-speed execution type programs	OFF: Asynchronous mode ON: Synchronous mode	• Asynchronous mode: Mode where the operations for the low-speed execution type program are continued during excess time. • Synchronous mode: Mode where the operations for the low-speed execution type program are started from the next scan even when there is excess time.	U	New	Qn(H) QnPH
SM331	Normal SFC program execution status	OFF: Not executed ON: Being executed	• This relay stores the information on whether the normal SFC program is in execution or not. • Used as an interlock for execution of the SFC control instruction.	S (Status change)	New	Qn(H) [3] QnPH [4] QnPRH
SM332	Program execution management SFC program execution status	OFF: Not executed ON: Being executed	• This relay stores the information on whether the SFC program for program execution management is in execution or not. • Used as an interlock for execution of the SFC control instruction.			
SM390	Access execution flag	ON indicates completion of intelligent function module access	• This relay stores the status information on the intelligent function module access instruction that was just executed. (This data is overwritten if the intelligent function module access instruction is executed again.) • Used by the user in a program as a completion bit.	S (Status change)	New	Qn(H) QnPH QnPRH
SM391	GINT instruction execution completion flag	OFF: Not executed ON: Execution completed	Stores the execution status of the S(P).GINT instruction. • Turns off before execution of the instruction. • Turns on after completion of the instruction.	S (Instruction execution)	New	QnU

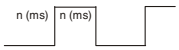
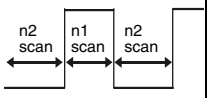
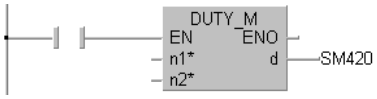
Tab. A-43: *Special relays (2): System information*

[1] This applies to the CPU of function version B or later.
[2] The module whose first 5 digits of serial No. is "09012" or higher.
[3] The module whose first 5 digits of serial No. is "04122" or higher.
[4] The module whose first 5 digits of serial No. is "07032" or higher.
[5] The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.
[6] The relevant modules are as follows:
• The Universal model QCPU whose serial number (first five digits) is "10012" or higher.
• Q13UDHCPU, Q26UDHCPU
[7] The Universal model QCPU except the Q00UJCPU.
[8] This applies when the first five digits of the serial number is "12052" or higher.

A.6.3 System clocks

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM400	Always ON	ON OFF	This flag is normally ON	S (Every END processing)	M9036	QCPU LCPU
SM401	Always OFF	ON OFF	This flag is normally OFF	S (Every END processing)	M9037	
SM402	ON for 1 scan only after RUN	ON OFF 1 scan	• After RUN, ON for 1 scan only. • This connection can be used for scan execution type programs only. • When an initial execution type program is used, this relay turns off at the END processing of the scan execution type program in the first scan after the CPU module enters the RUN status. 	S (Every END processing)	M9038	Qn(H) QnPH QnPRH QnU LCPU
			After RUN, ON for 1 scan only.		New	Q00J/Q00 /Q01
SM403	After RUN, OFF for 1 scan only	ON OFF 1 scan	• After RUN, OFF for 1 scan only. • This connection can be used for scan execution type programs only. • When an initial execution type program is used, this relay turns on at the END processing of the scan execution type program in the first scan after the CPU module enters the RUN status. 	S (Every END processing)	M9039	Qn(H) QnPH QnPRH QnU LCPU
			After RUN, OFF for 1 scan only.		New	Q00J/Q00 /Q01
SM404	ON for 1 scan only after RUN	ON OFF 1 scan	After RUN, ON for 1 scan only. This connection can be used for scan execution type programs only.	S (Every END processing)	New	Qn(H) QnPH
SM405	After RUN, OFF for 1 scan only	ON OFF 1 scan	After RUN, OFF for 1 scan only. This connection can be used for scan execution type programs only.	S (Every END processing)	New	
SM409	0.01 second clock	0.005 s 0.005 s	Repeatedly changes between ON and OFF at 5-ms interval. When power supply is turned ON, or reset is performed, starts with OFF.	S (Status change)	New	Qn(H) QnPH QnPRH QnU LCPU
SM410	0.1 second clock	0.05 s 0.05 s	Repeatedly changes between ON and OFF at each designated time interval. When power supply is turned ON, or reset is performed, starts with OFF.	S (Status change)	M9030	QCPU LCPU
SM411	0.2 second clock	0.1 s 0.1 s			M9031	
SM412	1 second clock	0.5 s 0.5 s			M9032	
SM413	2 second clock	1 s 1 s			M9033	
SM414	2x n second clock	n (s) n (s)	Goes between ON and OFF in accordance with the number of seconds designated by SD414.		M9034 format change	

Tab. A-44: *Special relays (3): System clocks*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM415	2 x n ms clock		Goes between ON and OFF in accordance with the number of milliseconds designated by SD415.	S (Status change)	New	Qn(H) QnPH QnPRH QnU LCPU
SM420	User timing clock No. 0		Relay repeats ON/OFF switching at fixed scan intervals. When power supply is turned ON, or reset is performed, goes from OFF to start. The ON/OFF intervals are set with the DUTY instruction. 	S (Every END processing)	M9020	QCPU LCPU
SM421	User timing clock No.1				M9021	
SM422	User timing clock No. 2				M9022	
SM423	User timing clock No. 3				M9023	
SM424	User timing clock No. 4					
SM430	User timing clock No. 5				M9024	
SM431	User timing clock No. 6					
SM432	User timing clock No. 7		For use with SM420 through SM424 low speed programs.	S (Every END processing)	New	Qn(H) QnPH
SM433	User timing clock No. 8					
SM434	User timing clock No. 9					

Tab. A-44: *Special relays (3): System clocks*

A.6.4 Scan information

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM510	Low speed program execution flag	OFF: Completed or not executed ON: Execution under way	Goes ON when low-speed execution type program is executed.	S (Every END processing)	New	Qn(H) QnPH
SM551	Reads module service interval	OFF: Ignored ON: Read	When this goes from OFF to ON, the module service interval designated by SD550 is read to SD551 through SD552.	U	New	Qn(H) QnPH QnPRH

Tab. A-45: *Special relays (4): Scan information*

A.6.5 I/O refresh

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM580	Program to program I/O refresh	OFF: Not refreshed ON: Refreshed	When this relay is turned on, I/O refresh is performed after execution of the first program, and then the next program is executed. When a sequence program and a SFC program are to be executed, the sequence program is executed, I/O refresh is performed, and then the SFC program is executed.	U	New	Q00J/Q00/Q01 [1)]

Tab. A-46: *Special relays (5): I/O refresh*

[1] This applies to the CPU of function version B or later.

A.6.6 Drive information

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM600	Memory card usable flags	OFF: Unusable ON: Use enabled	ON when memory card is ready for use by user.	S (Status change)	New	Qn(H) QnPH QnPRH QnU [1]
			Turns ON when the SD memory card becomes ready for use. (This relay turns on when a compatible SD memory card is inserted and set to be enabled with the SD memory card lock switch.)			LCPU
SM601	Memory card protect flag	OFF: No protect ON: Protect	Goes ON when memory card protect switch is ON.	S (Status change)	New	Qn(H) QnPH QnPRH QnU [1] LCPU
SM602	Drive 1 flag	OFF: No drive 1 ON: Drive 1 present	Goes ON when drive 1 (card 1 RAM area) is present.	S (Status change)	New	Qn(H) QnPH QnPRH QnU [1]
SM603	Drive 2 flag	OFF: No drive 2 ON: Drive 2 present	Goes ON when drive 2 (card 1 ROM area) is present.	S (Status change)	New	Qn(H) QnPH QnPRH QnU [1]
			Is ON while a SD memory card is being inserted. (This relay is ON while a SD memory card is being inserted, regardless of the availability and the type of the card.)	S (Status change)		LCPU
SM604	Memory card in-use flag	OFF: Not in use ON: In use	Goes ON when memory card is in use.	S (Status change)	New	Qn(H) QnPH QnPRH QnU [1] LCPU
SM605	Memory card remove/insert prohibit flag	OFF: Remove/insert enabled ON: Remove/insert prohibited	Goes ON when memory card cannot be inserted or removed.	U	New	Qn(H) QnPH QnPRH QnU [1]
			Turns ON to disable the insertion and removal of a memory card. (Turns ON when a compatible SD memory card is inserted and set to be enabled with the SD memory card lock switch. This relay does not turn ON while "ICM.OPE.ERROR" occurs.)	S (Status change)	New	LCPU
SM606	SD memory card forced disable instruction	OFF: SD memory card forced disable cancel instruction ON: SD memory card forced disable instruction	• This relay is turned on to execute the SD memory card forced disable instruction. • When there are any functions accessing to an SD memory card, the process of disablement is held until it is completed. • This relay is turned off to cancel the SD memory card forced disable instruction.	U	New	LCPU
SM607	SD memory card forced disable status flag	OFF: Not being disabled by SD memory card forced disable instruction ON: Being disabled by SD memory card forced disable instruction	• This relay turns on when an SD memory card is disabled by turning on SM606 (SD memory card forced disable instruction). • This relay turns off when the forced disable status of SD memory card is canceled by turning off SM606 (SD memory card forced disable instruction).	S (Status change)	New	LCPU

Tab. A-47: *Special relays (6): Drive information*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM609	Memory card remove/insert enable flag	OFF: Remove/insert prohibited ON: Remove/insert enabled	• This relay is turned ON to enable the insertion and removal of a memory card. • Turned OFF by the system after the memory card is removed. • This relay can be used while both SM604 and SM605 are off.	S/U	New	Qn(H) QnPH QnPRH QnU 1)
SM620	Drive 3/4 usable flags	OFF: Unusable ON: Use enabled	Always ON	S (Initial)	New	QCPU LCPU
SM621	Drive 3/4 protect flag	OFF: No protect ON: Protect	Always OFF	S (Initial)	New	QCPU LCPU
SM622	Drive 3 flag	OFF: No drive 3 ON: Drive 3 present	Always ON	S (Initial)	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU 2) LCPU
SM623	Drive 4 flag	OFF: No drive 4 ON: Drive 4 present	Always ON	S (Initial)	New	QCPU LCPU
SM624	Drive 3/4 in-use flag	OFF: Not in use ON: In Use	This relay is ON while a file stored in the drive 3 (standard RAM) or the drive 4 (standard ROM) is being used.	S (Status change)	New	Qn(H) QnPH QnPRH QnU LCPU
SM640	File register use	OFF: File register not in use ON: File register in use	Goes ON when file register is in use.	S (Status change)	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU 2) LCPU
SM650	Comment use	OFF: Comment not used ON: Comment in use	Goes ON when comment ile is in use.	S (Status change)	New	Qn(H) QnPH QnPRH QnU LCPU
SM660	Boot operation	OFF: Internal memory execution ON: Boot operation in progress	Goes ON while boot operation is in process Goes OFF if boot designation switch is OFF.	S (Status change)	New	Qn(H) QnPH QnPRH
		OFF: Program memory execution ON: Boot operation in progress	Goes ON while boot operation is in process			Q00J/Q00/Q01 QnU 3) LCPU
SM671	Latch data backup to standard ROM completion flag	OFF: Not completed ON: Completed	• This relay turns on when latch data backup to the standard ROM is completed. • Time when the backup is completed is stored in SD672 or later.	S (Status change)	New	QnU LCPU
SM672	Memory card A file register access range flag	OFF: Within access range ON: Outside access range	Goes ON when access is made to area outside the range of file register R of memory card A (set within END processing). Reset at user program.	S/U	New	Qn(H) QnPH QnPRH
SM675	Error completion of latch data backup to standard ROM	OFF: No Error ON: Error	• This relay turns on if latch data backup to the standard ROM is not completed. • This relay turns off when the backup is completed.	S	New	QnU LCPU

Tab. A-47: *Special relays (6): Drive information*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM676	Specification of restration repeated execution	OFF: Not specified ON: Specified	• When latch data are backed up while this relay is on, the backup data will be restored at every power-on of the CPU module. • The backup data will be restored at every power-on until the latch data are deleted or the latch data are backed up again.	U	New	QnU LCPU
SM680	Program memory write error	OFF: Write not executed/normal ON: Write error	This relay turns on if a write error is detected during writing to the program memory (flash ROM). This relay turns off when a write command is given.	S (At write)	New	QnU LCPU
SM681	Program memory writing flag	OFF: Write not executed ON: During writing	This relay is on during writing to the program memory (flash ROM) and turns off when the writing is completed.	S (At write)	New	QnU LCPU
SM682	Program memory overwrite count error flag	OFF: Overwrite count is less than 100,000 ON: Overwrite count is 100,000 or more	This relay turns on when overwrite count of the program memory (flash ROM) reaches to 100,000. (It is necessary to change CPU module.)	S (At write)	New	QnU LCPU
SM685	Standard ROM write error	OFF: Write not executed/normal ON: Write error	This relay turns on if a write error is detected during writing to the standard ROM (flash ROM). This relay turns off when a write command is given.	S (At write)	New	QnU LCPU
SM686	Standard ROM writing flag	OFF: Overwrite not executed ON: During overwriting	This relay is on during writing to the standard ROM (flash ROM) and turns off when the writing is completed.	S (At write)	New	QnU LCPU
SM687	Standard ROM overwrite count error flag	OFF: Overwrite count is less than 100,000 ON: Overwrite count is 100,000 or more	This relay turns on when overwrite count of the standard ROM (flash ROM) reaches to 100,000. (It is necessary to change CPU module.)	S (At write)	New	QnU LCPU
SM691	Backup start preparation status flag	OFF: Backup start preparation not completed ON: Backup start preparation completed	Turns on when the backup preparation is completed.	S (Status change)	New	QnU [1)] LCPU
SM692	Restoration complete flag	OFF: Restoration not completed ON: Restoration completed	This relay turns on when backup data in a memory card has been restored.			

Tab. A-47: *Special relays (6): Drive information*

[1] The modules whose serial number (first five digits) is "10102" or higher are the relevant models. (Except the Q00UJCPU, Q00UCPU, and Q01UCPU)

[2] The Universal model QCPU except the Q00UJCPU.

[3] The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

A.6.7 Instruction related special relays

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM700	Carry flag	OFF: Carry OFF ON: Carry ON	Carry flag used in application instruction.	S (Instruction execution)	M9012	QCPU LCPU
SM701	Number of output characters selection	OFF: Outputs until NUL ON: 16 characters output	Used for the PR, PRC, BINDA, DBINDA, BINHA, DBINHA, BCDDA, DBCDDA, or COMRD instruction	U	M9049	Qn(H) QnPH QnPRH QnU LCPU
SM702	Search method	OFF: Search next ON: 2-part search	Designates method to be used by search instruction. Data must be arranged for 2-part search.	U	New	QCPU LCPU
SM703	Sort order	OFF: Ascending order ON: Descending order	The sort instruction is used to designate whether data should be sorted in ascending order or in descending order.	U	New	
SM704	Block comparison	OFF: Non-match found ON: All match	Goes ON when all data conditions have been met for the BKCMP instruction.	S (Instruction execution)	New	
			This relay turns on when all data conditions are met for the DBKCMP instruction.			QnU [2] LCPU
SM709	DT/TM instruction improper data detection flag	OFF: Improper data not detected ON: Improper data detected	This relay turns on when the data to be compared by the DT or TM instruction cannot be recognized as date or time data, when the device (three words) to be compared is exceeding the specified device range.	S (Instruction execution)/U	New	QnU [2] LCPU
SM710	CHK instruction priority ranking flag	OFF: Conditions priority ON: Pattern priority	Remains as originally set when OFF. CHK priorities updated when ON.	S (Instruction execution)	New	Qn(H) QnPH QnPRH
SM715	EI flag	OFF: During DI ON: During EI	ON when EI instruction is being executed.	S (Instruction execution)	New	QCPU LCPU
SM716	Block comparison (Except an interrupt program)	OFF: Mismatch found ON: No mismatch	This relay turns on when all data conditions are met for the DBKCMP instruction. (Initial execution type program and scan execution type program or standby type program executed from initial execution type program or scan execution type program)			QnU [2] LCPU
SM717	Block comparison (Interrupt program)	OFF: Mismatch found ON: No mismatch	This relay turns on when all data conditions are met for the DBKCMP instruction. (Interrupt program, fixed scan execution type program, or standby type program executed from interrupt program or fixed scan execution type program)			
SM718	Block comparison (Interrupt program (I45))	OFF: Mismatch found ON: No mismatch	This relay turns on when all data conditions are met for the DBKCMP instruction. (Interrupt program (I45) or standby type program that was executed from interrupt program (I45))			QnU [3]
SM720	Comment read completion flag	OFF: Comment read not completed ON: Comment read completed	SM720 is set for one scan after the execution of the COMRD or PRC instruction	S (Status change)	New	Qn(H) QnPH
			This relay turns on only during first scan after the processing of the COMRD instruction is completed.			QnPRH QnU LCPU

Tab. A-48: *Special relays (7): Instruction related special relays*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM721	File being accessed	OFF: File is not accessed ON: File is accessed	This flag is ON while a file is being accessed by the SP.FWRITE, SP.FREAD, COMRD, PRC, or LEDC instruction	S (Status change)	New	Qn(H) QnPH
			This relay is on while a file is being accessed by the SP. FWRITE, SP. FREAD, COMRD, or LEDC instruction.			Qn(H) QnPH QnPRH
			This relay is on while a file is being accessed by the SP. FWRITE, SP. FREAD, COMRD, or SP.DEVST instruction.			QnU
			• This relay is on while a file is being accessed by the SP. FWRITE, SP. FREAD, COMRD, or SP.DEVST instruction. • This relay is on while a SD memory card or the standard ROM is being accessed.			LCPU
			This relay is on while an ATA card or the standard ROM is being accessed.			QnU [4)]
			This relay is on while the S(P).SFCSCOMR or the S(P).SFCTCOMR instruction is executed.			QnU [11)]
SM722	BIN/DBIN instruction error disabling flag	OFF: Error enabled ON: Error disabled	When this flag is set, an "OPERATION ERROR" is suppressed for both the BIN and the DBIN instruction.	U	New	QCPU LCPU
SM734	XCALL instruction execution condition designation	OFF: Not executed by execution condition risen ON: Executed by execution condition risen	• During OFF, XCALL instructions will not be executed even if execution condition is risen. • During ON, XCALL instructions will be executed when execution condition is risen.	U	New	Qn(H) [4)]
SM735	SFC comment readout instruction in execution flag	OFF: SFC comment readout instruction is inactivated. ON: SFC comment readout instruction is activating.	This relay turns on while a SFC step comment readout instruction (S(P).SFCSCOMR) or SFC transmission condition comment readout instruction (S(P). SFCTCOMR) is being executed.	S (Status change)	New	Qn(H) [5)] QnPH [6)] QnPRH [6)] QnU [11)]
SM738	MSG instruction reception flag	OFF: Instruction not executed ON: Instruction execution	Goes ON when MSG instruction is executed.	S (Instruction execution)	New	Qn(H) QnPRH
SM740	Display unit availability flag	OFF: Not usable ON: Usable	This relay is on while the display unit can be used.	S (Initial/Status change)	New	LCPU
SM750	Scaling instruction search method setting	OFF: Search next ON: 2-part search	Determines a search method when the scaling instruction is executed.	U	New	QnU [2)] LCPU
SM774	PID bumpless processing	OFF: Forces match ON: Does not force match	In manual mode, designates whether or not to force the SV value to match the PV value.	U	New	Q00J/Q00/Q01 [1)] Qn(H) QnPRH QnU LCPU
SM775	Selection of link refresh processing during COM/CCOM instruction execution	OFF: Performs link refresh ON: No link refresh performed	Select whether or not to perform link refresh processing in cases where only general data processing will be conducted during the execution of the COM instruction.	U	New	Q00J/Q00/Q01 Qn(H) QnPH
		OFF: Performs refresh processes other than an I/O refresh ON: Performs refresh set by SD778	Select whether to perform refresh processes other than an I/O refresh set by SD778 when the COM or CCOM instruction is executed.			Q00J/Q00/Q01 [1)] Qn(H) [7)] QnPH [4)] QnPRH QnU LCPU

Tab. A-48: *Special relays (7): Instruction related special relays*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM776	Enable local device at CALL	OFF: Local device disabled ON: Local device enabled	This flag specifies whether to enable or disable the local device in the program called at the CALL instruction.	U	New	Qn(H) QnPH QnPRH QnU [10)] LCPU
SM777	Enable local device in interrupt program	OFF: Local device disabled ON: Local device enabled	This flag specifies whether to enable or disable the local device at the execution of an interrupt program.	U	New	
SM794	PID bumpless processing(for incomplete derivative)	OFF: Matched ON: Not matched	Specifies whether to match the set value (SV) with the process value (PV) or not in the manual mode.	S (When instruction/END processing executed)	New	Q00J/Q00/Q01 [1)] Qn(H) [8)] QnPRH QnU LCPU
SM796	Block information using multiple CPU high-speed transmission dedicated instruction (for CPU No.1)	OFF: Block is secured ON: Block set by SD796 cannot be secured	This relay turns on when the number of the remaining blocks in the dedicated instruction transmission area used for the multiple CPU high-speed transmission dedicated instruction (target CPU= CPU No.1) is less than the number of blocks specified in SD796. This relay is on when an instruction is executed, and is off while an END processing is being executed or when free space is available in the area.	S (When instruction/END processing executed)	New	QnU [9)]
SM797	Block information using multiple CPU high-speed transmission dedicated instruction (for CPU No.2)	OFF: Block is secured ON: Block set by SD797 cannot be secured	This relay turns on when the number of the remaining blocks in the dedicated instruction transmission area used for the multiple CPU high-speed transmission dedicated instruction (target CPU= CPU No.2) is less than the number of blocks specified in SD797. This relay is on when an instruction is executed, and is off while an END processing is being executed or when free space is available in the area.	S (When instruction/END processing executed)	New	QnU [9)]
SM798	Block information using multiple CPU high-speed transmission dedicated instruction (for CPU No.3)	OFF: Block is secured ON: Block set by SD798 cannot be secured	This relay turns on when the number of the remaining blocks in the dedicated instruction transmission area used for the multiple CPU high-speed transmission dedicated instruction (target CPU= CPU No.3) is less than the number of blocks specified in SD798. This relay is on when an instruction is executed, and is off while an END processing is being executed or when free space is available in the area.	S (When instruction/END processing executed)	New	QnU [9)]
SM799	Block information using multiple CPU high-speed transmission dedicated instruction (for CPU No.4)	OFF: Block is secured ON: Block set by SD799 cannot be secured	This relay turns on when the number of the remaining blocks in the dedicated instruction transmission area used for the multiple CPU high-speed transmission dedicated instruction (target CPU= CPU No.) is less than the number of blocks specified in SD799. This relay is on when an instruction is executed, and is off while an END processing is being executed or when free space is available in the area.	S (When instruction/END processing executed)	New	QnU [9)]

Tab. A-48: *Special relays (7): Instruction related special relays*

[1] This applies to the CPU module of function version B or later.

[2] The relevant modules are as follows:
- The Universal model QCPU whose serial number (first five digits) is "10102" or higher.
- Q00UJCPU, Q00UCPU, Q01UCPU

[3] The relevant modules are as follows:
- The Universal model QCPU whose serial number (first five digits) is "10102" or higher.
- Q00UCPU, Q01UCPU

[4] The module whose first 5 digits of serial No. is "07032" or higher.

[5] The module whose first 5 digits of serial No. is "06082" or higher.

[6] The module whose first 5 digits of serial No. is "07012" or higher.

[7] The module whose first 5 digits of serial No. is "04012" or higher.

[8] The module whose first 5 digits of serial No. is "05032" or higher.

[9] The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

[10] The Universal model QCPU except the Q00UJCPU.

[11] This applies when the first five digits of the serial number is "12052" or higher.

A.6.8 Debugging

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM800	Trace preparation	OFF: Not prepared ON: Ready	Goes ON when the trace preparation is completed.	S (Status change)	New	Qn(H) QnPH QnPRH QnU [1)] LCPU
SM801	Trace start	OFF: Suspend ON: Start	Trace is started when this goes ON. Suspended when OFF (Related special M all OFF).	U	M9047	Qn(H) QnPH QnPRH QnU [1)] LCPU
SM802	Trace execution in progress	OFF: Suspend ON: Start	Goes ON during execution of trace.	S (Status change)	M9046	Qn(H) QnPH QnPRH QnU [1)] LCPU
SM803	Trace trigger	OFF → ON: Start	• This relay turns on when the specified trigger condition is met. • This relay is turned on to meet the trigger condition.	S (Status change)/U	M9044	Qn(H) QnPH QnPRH QnU [1)] LCPU
SM804	After Trace trigger	OFF: Not after trigger ON: After trigger	Goes ON after trace trigger is triggered.	S (Status change)	New	Qn(H) QnPH QnPRH QnU [1)] LCPU
SM805	Trace completed	OFF: Not completed ON: End	Goes ON at completion of trace.	S (Status change)	M9043	Qn(H) QnPH QnPRH QnU [1)] LCPU
SM826	Trace error	OFF: Normal ON: Error	Goes ON if error occurs during execution of trace/sampling trace.	S (Status change)	New	Qn(H) QnPH QnPRH QnU [1)] LCPU
SM829	Forced registration specification of trace setting	OFF: Forced registration disabled ON: Forced registration enabled	When this relay is turned on and a sampling trace setting is registered using a programming tool, the sampling trace setting can be registered with the CPU module even when the trigger condition has been met.	U	New	QnU [1)] LCPU
SM841	Auto logging	OFF: No auto logging ON: Auto logging	This relay is on while auto logging is being executed. This relay turns off when auto logging is completed and the SD memory card lock switch is slid toward the module top to stop access to the SD memory card.	S (Status change)		LCPU

Tab. A-49: *Special relays (8): Debugging*

[1] The Universal model QCPU except the Q00UJCPU.

A.6.9 Conversion from A series to System Q or L series

For a conversion from the MELSEC A series to the MELSEC System Q or L series the special relays M9000 through M9255 (A series) correspond to the special relays SM1000 through SM1255 for QCPU or LCPU after the A to Q/L conversion. (Note that the Basic model QCPU and Redundant CPU do not support the A to Q/L conversion.)

These special relays are all set by the system and cannot be changed by a user-program. Users intending to set or reset these relays should alter their programs so that only real QCPU or LCPU special relays are applied. An exception are the special relays M9084 and M9200 through M9255. If a user can set or reset some of these special relays before conversion, the user can also set and reset the corresponding relays among SM1084 and SM1200 through SM1255 after the conversion.
Refer to the manuals of the CPUs and the networks MELSECNET and MELSECNET/B for detailed information on the special relays of the A series.

NOTE *To use the converted special relay in the High Performance model QCPU, Process CPU, Universal model QCPU, or LCPU, check "Use special relay/special register from SM/SD1000" under "A-PLC Compatibility Setting".*
Project window ⇒ [Parameter] ⇒ [PLC Parameter] ⇒ [PLC System]
Note that the processing time will increase when the converted special relay is used.

How to read the Special Relay for Modification column:

• If the special relay number for QCPU or LCPU is provided, correct the program using it.

• If no special relay is specified (–), the converted special relay can be used.

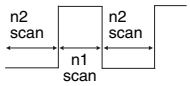
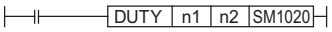
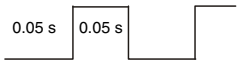
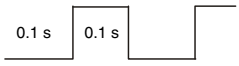
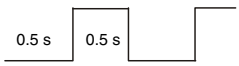
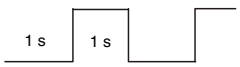
• If the special relay cannot be used in QCPU or LCPU, this is indicated as "No function for QCPU/LCPU".

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Valid for:
M9000	SM1000	—	Fuse blown	OFF: Normal ON: Fuse blown module with blown fuse present	• Turns on if there is at least one output module whose fuse has blown. • This relay remains on even after the condition returns to normal. • Output modules on remote I/O stations are also checked for blown fuse.	Qn(H) QnPH QnU 1)
M9002	SM1002	—	I/O module verification error	OFF: Normal ON: Error	• This relay turns on if the status of the I/O module differs from that registered at power-on. • This relay remains on even after the system returns to normal. • I/O modules on remote I/O stations are also checked. • This relay is reset only when SD1116 to SD1123 are reset.	Qn(H) QnPH QnU 1) LCPU

Tab. A-50: *Special relays (9): Conversion from A series to System Q or L series*

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Valid for:
M9005	SM1005	—	AC DOWN detection	OFF: AC DOWN not detected ON: AC DOWN detected	• This relay turns on if a momentary power failure within 20ms occurs during use of an AC power supply module. • This relay is reset when the CPU module is powered off and then on.	Qn(H) QnPH QnU 1)
					• This relay turns on if a momentary power failure within 10ms occurs when using an AC power supply module. • This relay is reset when the CPU module is powered off and then on.	LCPU
					• This relay turns on if a momentary power failure within 10ms occurs during use of a DC power supply module. • This relay is reset when the CPU module is powered off and then on.	Qn(H) QnPH QnU 1) LCPU
M9006	SM1006	—	Battery low	OFF: Normal ON: Battery low	• This relay turns on when the battery voltage drops to or below the specified. • It turns off when the battery voltage returns to normal.	Qn(H) QnPH QnU 1) LCPU
M9007	SM1007	—	Battery low (latched)	OFF: Normal ON: Battery low	• This relay turns on when the battery voltage drops to or below the specified. • This relay remains on even after the battery voltage returns to normal.	
M9008	SM1008	SM1	Self-diagnostic error	OFF: No error ON: Error	This relay turns on if an error is detected by selfdiagnostics.	
M9009	SM1009	SM62	Annunciator detection	OFF: No F number detected ON: F number detected	• This relay turns on when the OUT F or SET F instruction is executed. • It turns off when the SD1124 value is cleared to zero.	
M9011	SM1011	SM56	Operation error flag	OFF: No error ON: Error	• This relay turns on when an operation error occurs during execution of an application instruction. • This relay remains on even after the system returns to normal.	Qn(H) QnPH QnU 1)
M9012	SM1012	SM700	Carry Flag	OFF: Carry OFF ON: Carry ON	Carry flag used in application instruction.	Qn(H) QnPH
M9016	SM1016	No function for QCPU/LCPU	Data memory clear flag	OFF: Ignored ON: Output cleared	When SM1016 turns on and remote RUN mode is activated from a computer, all the data memory including the latch range (except for the special relay and special register) is cleared.	Qn(H) QnPH
M9017	SM1017	No function for QCPU/LCPU	Data memory clear flag	OFF: Ignored ON: Output cleared	When SM1017 turns on and remote RUN mode is activated from a computer, all the data memory that is not latched (except for the special relay and special register) is cleared.	Qn(H) QnPH

Tab. A-50: *Special relays (9): Conversion from A series to System Q or L series*

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Valid for:
M9020	SM1020	—	User timing clock No. 0		• This relay repeatedly turns on and off at the specified scan intervals. • When the CPU module is powered on or reset, this relay is set to on from off to start the clock. Set the intervals of on/off by DUTY instruction.  n1: ON scan interval n2: OFF scan interval When SM1020 to SM1024 are specified for the DUTY instruction in programs, if the CPU type is changed from the High Performance model QCPU or Process CPU to the Universal model QCPU or LCPU, they are replaced with SM420 to SM424. (For the Universal model QCPU and LCPU, SM1020 to SM1024 cannot be specified.)	Qn(H) QnPH QnU [1] LCPU
M9021	SM1021	—	User timing clock No. 1			
M9022	SM1022	—	User timing clock No. 2			
M9023	SM1023	—	User timing clock No. 3			
M9024	SM1024	—	User timing clock No. 4			
M9025	SM1025	—	Clock data set request	OFF: Ignored ON: Set request present used	Clock data stored in SD1025 to SD1028 are written to the CPU module after the END instruction execution in the scan where SM1025 is turned on.	
M9026	SM1026	—	Clock data error	OFF: No error ON: Error	This relay turns on if an error occurs in the clock data (SD1025 to SD1028), and is off while there is no error.	
M9028	SM1028	—	Clock data read request	OFF: Ignored ON: Read request	This relay is turned on to read clock data and store them as BCD values into SD1025 to SD1028.	
M9029	SM1029	No function for QCPU/LCPU	Batch processing of data communications request	OFF: Batch processing not conducted ON: Batch processing conducted	• When this relay is turned on in the program, all the data communication requests accepted during one scan are processed in the END processing of that scan. • The batch processing of data communication requests can be turned on or off during running. • The default is OFF (processed one at a time for each END processing in the order in which data communication requests are accepted).	Qn(H) QnPH
M9030	SM1030	—	0.1 second clock		• 0.1-, 0.2-, 1-, and 2-second clocks are generated. • The relay turns on or off not for each scan, but also during a scan if the time has elapsed. • When the CPU module is powered on or reset, this relay is set to on from off to start the clock.	Qn(H) QnPH QnU [1] LCPU
M9031	SM1031	—	0.2 second clock			
M9032	SM1032	—	1 second clock			
M9033	SM1033	—	2 second clock			
M9034	SM1034	—	2n minute clock (1 minute clock) [2]		• This relay repeatedly turns on and off according to the number of seconds specified in SD414. (Default: n = 30) • The relay turns on or off not for each scan, but also during a scan if the time has elapsed. • When the CPU module is powered on or reset, this relay is set to on from off to start the clock.	Qn(H) QnPH QnU [1] LCPU

Tab. A-50: *Special relays (9): Conversion from A series to System Q or L series*

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Valid for:
M9036	SM1036	—	Always ON	ON OFF	• This relay is used for initialization or as a dummy contact of application instructions in the program. • SM1036 and SM1037 are turned on or off regardless of the key switch setting on the front face of the CPU module. The states of SM1038 and SM1039 change depending on the key switch setting. When it is set to STOP, the relay is off.When it is set to other than STOP, SM1038 is on for one scan only and SM1039 is off for one scan only.	Qn(H) QnPH QnU [1)] LCPU
M9037	SM1037	—	Always OFF	ON OFF		
M9038	SM1038	—	ON for 1 scan only after RUN	ON OFF 1 scan		
M9039	SM1039	—	RUN flag (After RUN, OFF for 1 scan only)	ON 1 scan OFF		
M9040	SM1040	SM206	PAUSE enable coil	OFF: PAUSE disabled ON: PAUSE enabled	This relay is on when the CPU module is in PAUSE status or when the PAUSE contact is on.	Qn(H) QnPH
M9041	SM1041	SM204	PAUSE status contact	OFF: PAUSE not in effect ON: PAUSE in effect		Qn(H) QnPH QnU [1)] LCPU
M9042	SM1042	SM203	STOP status contact	OFF: STOP not in effect ON: STOP in effect	This relay turns on when the RUN key switch or RUN/STOP switch is set to STOP.	
M9043	SM1043	SM805	Sampling trace completed	OFF: Sampling trace in progress ON: Sampling trace completed	This relay turns on after execution of the TRACE instruction and upon completion of sampling trace performed the number of times preset by the parameter. Reset when TRACER instruction is executed.	
M9045	SM1045	No function for QCPU/LCPU	Watchdog timer (WDT) reset	OFF: Does not reset WDT ON: Resets WDT	If SM1045 is turned on, the watchdog timer is reset when the ZCOM instruction and batch processing of data communication requests are executed. (Use this when scan time exceeds 200 ms.)	Qn(H) QnPH
M9046	SM1046	SM802	Sampling trace	OFF: Trace not in progress ON: Trace in progress	This relay is on during execution of sampling trace.	Qn(H) QnPH QnU [1)] LCPU
M9047	SM1047	SM801	Sampling trace preparations	OFF: Sampling Trace suspended ON: Sampling Trace started	Sampling trace is not executed unless SM1047 is turned ON. Sampling trace is cancelled when SM1047 turns off.	Qn(H) QnPH
M9049	SM1049	SM701	Selection of number of characters output	OFF: Output until NUL ON: 16 characters output	• When SM1049 is off, characters up to NULL (00H) code are output. • When SM1049 is ON, ASCII codes of 16 characters are output.	Qn(H) QnPH
M9051	SM1051	No function for QCPU/LCPU	CHG instruction execution disable	OFF: Enabled ON: Disable	• Switched ON to disable the CHG instruction. • Turn this on when requesting program transfer. It is automatically turned off upon completion of the transfer.	Qn(H) QnPH
M9052	SM1052	No function for QCPU/LCPU	SEG instruction switch	OFF: 7 segment display ON: I/O partial refresh	When SM1052 is on, the SEG instruction is used as an I/O part refresh instruction. When SM1052 is off, the SEG instruction is used as a 7-SEG display instruction.	Qn(H) QnPH

Tab. A-50: *Special relays (9): Conversion from A series to System Q or L series*

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Valid for:
M9056	SM1056	No function for QCPU/LCPU	Main side P, I set request	OFF: Other than when P, I set being requested ON: P, I set being requested	While a program is running, upon completion of transfer of another program (for example, a subprogram when the main program is running), a P and I set request is turned on. This relay automatically turns off upon completion of P and I setting.	Qn(H) QnPH
M9057	SM1057		Sub side P, I set request			
M9058	SM1058		Main program P, I set completion	Momentarily ON at P, I set completion	This relay turns on for a moment upon completion of P and I setting, and immediately turns off.	
M9059	SM1059		Sub program P, I set completion	Momentarily ON at P, I set completion		
M9060	SM1060		Sub program 2 P, I set request	OFF: Other than when P, I set being requested ON: P, I set being requested	While a program is running, upon completion of transfer of another program (for example, a subprogram when the main program is running), a P and I set request is turned on. This relay automatically turns off upon completion of P and I setting.	
M9061	SM1061		Sub program 3 P, I set request			
M9070	SM1070	No function for QCPU/LCPU	A8UPU/A8PUJ required search time [3)]	OFF: Read time not shortened ON: Read time shortened	When this is turned on, the search time in the A8UPU/A8PUJ can be shortened. (In this case, the scan time is extended by 10%.)	Qn(H) QnPH
M9084	SM1084	No function for QCPU/LCPU	Error check	OFF: Error check executed ON: No error check	This relay sets whether or not to check the following errors at the time of the END instruction processing (for setting of the END instruction processing time). • Check for fuse blown. • Check of battery • Collation check of I/O module	Qn(H) QnPH
M9091	SM1091	No function for QCPU/LCPU	Instruction error flag	OFF: No error ON: Error	• Turns on when the detail factor of the operation error is stored into SD1091. • Remains on even after the condition returns to normal.	

Tab. A-50: *Special relays (9): Conversion from A series to System Q or L series*

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Valid for:
M9100	SM1100	SM320	Presence/absence of SFC program	OFF: SFC programs not used ON: SFC programs used	This relay is on when an SFC program has been registered, and is off when no program is registered.	Qn(H) QnPH
M9101	SM1101	SM321	Start/stop SFC program	OFF: SFC programs stop ON: SFC programs start	• The same value as in SM1100 is set as the initial value. (This relay turns on when an SFC program is registered.) • This relay is turned off to stop SFC program execution. • This relay is turned on to resume the SFC program execution.	
M9102	SM1102	SM322	SFC program start state	OFF: Initial Start ON: Continue	In the SFC setting of the PLC parameter dialog box, Initial start is set for the SFC program start mode. • At initial start: OFF • At continue start: ON	
M9103	SM1103	SM323	Presence/absence of continuous transition	OFF: Continuous transition not effective ON: Continuous transition effective	Set whether to enable or disable continuous transition for the blocks where "continuous transition bit" of the SFC information device is not set.	
M9104	SM1104	SM324	Continuous transition suspension flag	OFF: When transition is completed ON: When no transition	• This relay is off during operation in the continuous transition mode or during continuous transition, and is on while continuous transition is not performed. • This relay is always on while the CPU module is operating not in the continuous transition mode.	
M9108	SM1108	SM90	Step transition watchdog timer start (equivalent of SD90)	OFF: Watchdog timer reset ON: Watchdog timer reset start	The relay turns on when measurement by the step transition monitoring timer is started. The step transition monitoring timer is reset when the relay turns off.	
M9109	SM1109	SM91	Step transition watchdog timer start (equivalent of SD91)			
M9110	SM1110	SM92	Step transition watchdog timer start (equivalent of SD92)			
M9111	SM1111	SM93	Step transition watchdog timer start (equivalent of SD93)			
M9112	SM1112	SM94	Step transition watchdog timer start (equivalent of SD94)			
M9113	SM1113	SM95	Step transition watchdog timer start (equivalent of SD95)			
M9114	SM1114	SM96	Step transition watchdog timer start (equivalent of SD96)			

Tab. A-50: *Special relays (9): Conversion from A series to System Q or L series*

<table>
<tr><th>ACPU Special Relay</th><th>Special Relay after Conversion</th><th>Special Relay for Modification</th><th>Name</th><th>Meaning</th><th>Details</th><th>Valid for:</th></tr>
<tr><td>M9196</td><td>SM1196</td><td>SM325</td><td>Operation output at block stop</td><td>OFF: Coil output OFF
ON: Coil output ON</td><td>• Selects the operation output when block stop is executed.
• On: Retains the on or off status of the coil used in the operation output of the step, which was being executed at the time of block stop.
• Off:Turns off all the coil outputs. (Operation output by the SET instruction is retained regardless of the on/off status of SM1196.)</td><td rowspan="4">Qn(H)
QnPH</td></tr>
<tr><td>M9197</td><td>SM1197</td><td>No function for QCPU/LCPU</td><td rowspan="2">Switch between blown fuse and I/O verification error display</td><td rowspan="2">Display is changed depending on combination of M9197 ON/OFF state and M9198 ON/OFF state.
<table><tr><th>SM1197</th><th>SM1198</th><th>I/O numbers to be displayed</th></tr><tr><td>OFF</td><td>OFF</td><td>X/Y0 to 7F0</td></tr><tr><td>ON</td><td>OFF</td><td>X/Y800 to FF0</td></tr><tr><td>OFF</td><td>ON</td><td>X/Y1000 to 17F0</td></tr><tr><td>ON</td><td>ON</td><td>X/Y1800 to 1FF0</td></tr></table></td><td rowspan="2">Switches I/O numbers between the fuse-blown module registers (SD1100 to SD1107) and I/O module verify error registers (SD1116 to SD1123) according to the on/off combination of SM1197 and SM1198.</td></tr>
<tr><td>M9198</td><td>SM1198</td><td>No function for QCPU/LCPU</td></tr>
<tr><td>M9199</td><td>SM1199</td><td>No function for QCPU/LCPU</td><td>Online recovery of sampling trace status latch data</td><td>OFF: Does not perform data recovery
ON: Performs data recovery</td><td>• Recovers the setting data stored in the CPU module at restart when sampling trace/status latch is executed.
• Turn this on to re-execute the sampling trace or status latch. (Rewriting data using the programming tool is not required.)</td></tr>
</table>

Tab. A-50: *Special relays (9): Conversion from A series to System Q or L series*

[1] The relevant modules are as follows:
- The Universal model QCPU whose serial number (first five digits) is "10102" or higher.
- Q00UJCPU, Q00UCPU, Q01UCPU

[2] 1 minute clock indicates the name of the special relay (M9034) of the ACPU.

[3] The A8UPU/A8PUJ is not available for the QCPU/LCPU.

A.6.10 Built-in Ethernet port and built-in Ethernet function

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1270	Time setting function (SNTP client) execution	OFF: No time setting function (SNTP client) execution ON: Time setting function (SNTP client) execution	This relay is turned on to perform the time setting function (SNTP client). (Turns on only when "Use" has been set for the time setting function in the time setting parameter.)	U	New	QnU [1] LCPU
SM1273	Remote password mismatch count clear	OFF: Normal ON: Clear	This relay is turned on to clear the accumulated number of mismatched remote password entries (SD979 to SD999).			
SM1292	IP address storage area write request	OFF: Ignored ON: Write request	The IP address setting stored in SD1292 to SD1297 are written to the IP address storage area (flash ROM) of the CPU module when the END instruction is executed in the scan where this relay is turned on.			QnU [2]
SM1293	IP address storage area write completion	OFF: Not completed ON: Completed	• This relay turns on when writing to the IP address storage area (flash ROM) is completed. • This relay turns off when the END instruction is executed in the scan where SM1292 is turned off.	S (Status change)		
SM1294	IP address storage area write error	OFF: Normal ON: Error	• This relay turns on when writing to the IP address storage area (flash ROM) fails. • This relay turns off when the END instruction is executed in the scan where SM1292 is turned off.			
SM1295	IP address storage area clear request	OFF: Ignored ON: Clear request	The IP address storage area (flash ROM) is cleared when the END instruction is executed in the scan where this relay is turned on.	U		
SM1296	IP address storage area clear completion	OFF: Not completed ON: Completed	• This relay turns on when clearing the IP address storage area (flash ROM) is completed. • This relay turns off when the END instruction is executed in the scan where SM1295 is turned off.	S (Status change)		
SM1297	IP address storage area clear error	OFF: Normal ON: Error	• This relay turns on when clearing the IP address storage area (flash ROM) fails. • This relay turns off when the END instruction is executed in the scan where SM1295 is turned off.			

Tab. A-51: *Special relays (10): Built-in Ethernet port and built-in Ethernet function*

[1] This applies to the Built-in Ethernet port QCPU.

[2] This applies to the built-in Ethernet port QCPU whose first five digits of serial No. is "11082" or higher.

A.6.11 Process control instruction

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1500	Hold mode	OFF: No-hold ON: Hold	Specifies whether or not to hold the output value when a range over occurs for the S.IN instruction range check.	U	New	QnPH QnPR
SM1501			Specifies whether or not the output value is held when a range over occurs for the S.OUT instruction range check.			

Tab. A-52: *Special relays (11): Process control instruction*

A.6.12 Redundant system (host system CPU information)

The special relays SM1510 to SM1599 store information of the host CPU module.

These special relays are valid only for redundant systems. SM1510 to SM1599 are set to off for stand-alone systems.

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU M9□□□</th><th>Valid for:</th></tr>
<tr><td>SM1510</td><td>Operation mode</td><td>OFF: Redundant system backup mode, stand-alone system
ON: Redundant system separate mode</td><td>This relay is on while the system is operating in the separate mode.</td><td>S (Every END processing)</td><td>New</td><td>QnPRH</td></tr>
<tr><td>SM1511</td><td>System A identification flag</td><td colspan="2" rowspan="2">• Distinguishes between system A and system B.
• The flag status does not change even if the tracking cable is disconnected.
<table><tr><th></th><th>System A</th><th>System B</th><th>If TRK. CABLE ERR. (error code: 6210) occurred (Unknown)</th></tr><tr><td>SM1511</td><td>ON</td><td>OFF</td><td>OFF</td></tr><tr><td>SM1512</td><td>OFF</td><td>ON</td><td>OFF</td></tr></table></td><td rowspan="3">S (Initial)</td><td>New</td><td>QnPRH</td></tr>
<tr><td>SM1512</td><td>System B identification flag</td><td>New</td><td>QnPRH</td></tr>
<tr><td>SM1513</td><td>Debug mode status flag</td><td>OFF: Not in debug mode
ON: Debug mode</td><td>This relay is on while the system is operating in the debug mode.</td><td>New</td><td>QnPRH</td></tr>
<tr><td>SM1515</td><td>Control system judgment flag</td><td colspan="2" rowspan="2">• Indicates operation system status.
• The flag status does not change even if the tracking cable is disconnected.
<table><tr><th></th><th>Control system</th><th>Standby system</th><th>If TRK. CABLE ERR. (error code: 6210) occurred (Unknown)</th></tr><tr><td>SM1515</td><td>ON</td><td>OFF</td><td>OFF</td></tr><tr><td>SM1516</td><td>OFF</td><td>ON</td><td>OFF</td></tr></table></td><td rowspan="2">S (Status change)</td><td>New</td><td>QnPRH</td></tr>
<tr><td>SM1516</td><td>Standby system judgment flag</td><td>New</td><td>QnPRH</td></tr>
<tr><td>SM1517</td><td>CPU module startup status</td><td>OFF: Power supply on startup
ON: Operation system switch start up</td><td>Turns on when the CPU module is started up by the system switching (switching from the standby system to the control system). Remains OFF when the standby system is switched to the control system by a power-ON startup.</td><td>S (Status change)</td><td>New</td><td>QnPRH</td></tr>
</table>

Tab. A-53: *Special relays (12): Redundant system (host system CPU information)*

Number	Name	Meaning	Description			Set by (if set)	ACPU M9□□□	Valid for:
SM1518	Standby system to control system switching status flag	ON 1 scan OFF	• This relay turns on during one scan after the standby system was switched to the control system. • This relay can be used only in a scan execution type program.			S (Every END processing)	New	QnPRH
SM1519	Previous Control System Identification Flag	ON 1 scan OFF	When the previous control system is System B, this relay turns on during one scan in System A, following the RUN state after both Systems A and B were simultaneously turned on or were reset.				New	QnPRH
SM1520	Data tracking transfer trigger specification	OFF: No trigger ON: Trigger	SM1520	Block 1	• When data is transferred based on the tracking setting of the Redundant parameter dialog box, the target block is specified as trigger. • When "Do auto forward Tracking block No.1" is selected for the tracking setting, SM1520 is turned on by the system at power-on or when the system is switched from STOP to RUN.In other cases, SM1520 to SM1583 are turned on by the user.	S (initial)/U	New	QnPRH
SM1521			SM1521	Block 2				
SM1522			SM1522	Block 3				
SM1523			SM1523	Block 4				
SM1524			SM1524	Block 5				
SM1525			SM1525	Block 6				
SM1526			SM1526	Block 7				
SM1527			SM1527	Block 8				
SM1528			SM1528	Block 9				
SM1529			SM1529	Block 10				
SM1530			SM1530	Block 11				
SM1531			SM1531	Block 12				
SM1532			SM1532	Block 13				
SM1533			SM1533	Block 14				
SM1534			SM1534	Block 15				
SM1535			SM1535	Block 16				
SM1536			SM1536	Block 17				
SM1537			SM1537	Block 18				
SM1538			SM1538	Block 19				
SM1539			SM1539	Block 20				
SM1540			SM1540	Block 21				
SM1541			SM1541	Block 22				
SM1542			SM1542	Block 23				
SM1543			SM1543	Block 24				
SM1544			SM1544	Block 25				
SM1545			SM1545	Block 26				
SM1546			SM1546	Block 27				
SM1547			SM1547	Block 28				
SM1548			SM1548	Block 29				
SM1549			SM1549	Block 30				
SM1550			SM1550	Block 31				
SM1551			SM1551	Block 32				

Tab. A-53: *Special relays (12): Redundant system (host system CPU information)*

Number	Name	Meaning	Description			Set by (if set)	ACPU M9□□□	Valid for:
SM1552	Data tracking transfer trigger specification	OFF: No trigger ON: Trigger	SM1552	Block 33	• When data is transferred based on the tracking setting of the Redundant parameter dialog box, the target block is specified as trigger. • When "Do auto forward Tracking block No.1" is selected for the tracking setting, SM1520 is turned on by the system at power-on or when the system is switched from STOP to RUN.In other cases, SM1520 to SM1583 are turned on by the user.	S (initial)/U	New	QnPRH
SM1553			SM1553	Block 34				
SM1554			SM1554	Block 35				
SM1555			SM1555	Block 36				
SM1556			SM1556	Block 37				
SM1557			SM1557	Block 38				
SM1558			SM1558	Block 39				
SM1559			SM1559	Block 40				
SM1560			SM1560	Block 41				
SM1561			SM1561	Block 42				
SM1562			SM1562	Block 43				
SM1563			SM1563	Block 44				
SM1564			SM1564	Block 45				
SM1565			SM1565	Block 46				
SM1566			SM1566	Block 47				
SM1567			SM1567	Block 48				
SM1568			SM1568	Block 49				
SM1569			SM1569	Block 50				
SM1570			SM1570	Block 51				
SM1571			SM1571	Block 52				
SM1572			SM1572	Block 53				
SM1573			SM1573	Block 54				
SM1574			SM1574	Block 55				
SM1575			SM1575	Block 56				
SM1576			SM1576	Block 57				
SM1577			SM1577	Block 58				
SM1578			SM1578	Block 59				
SM1579			SM1579	Block 60				
SM1581			SM1581	Block 61				
SM1582			SM1582	Block 62				
SM1583			SM1583	Block 63				
SM1590	System switching enable/disable flag from network module	OFF: System switching request issuing module absent ON: System switching request issuing module present	Turns on when a system switching request is issued from the network module. The module No. that issued system switching can be checked by SD1590. Turns off when all bits of SD1590 are off.			S (Every END processing)	New	QnPRH
SM1591	Standby system error detection disable flag at system switching	OFF: Error is detected by new standby system at system switching ON: Error is not detected by new standby system at system switching	This flag is used when switching the system in any of the following sources to determine whether to detect "STANDBY" (error code 6210) in the new standby system: [Reason(s) for system switching] • System switching with a programming tool • System switching using dedicated instruction • System switching by the intelligent function module			U	New	QnPRH

Tab. A-53: *Special relays (12): Redundant system (host system CPU information)*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1592	Enable/disable user system switching	OFF: Disable user system switching ON: Enable user system switching	This relay stores whether to enable manual switching using a programming tool or the system switching instruction (SP.CONTSW).	U	New	QnPRH
SM1593	Setting to access extension base unit of standby system CPU	OFF: Error ON: Ignored	This relay sets the behavior of the system after the standby CPU in the separate mode accessed the buffer memory of an intelligent function module mounted on an extension base unit. • OFF: "OPERATION ERROR" (error code: 4112) is returned. • ON: No processing	U	New	QnPRH
SM1595	Memory copy to other system start flag	OFF: No memory copy initiated ON: Start memory copy	When SM1595 is turned ON from OFF, memory copying from the control system to the standby system starts. Note that memory copy does not start even after SM1595 was turned on from off if the I/O No. of the copy destination (standby system CPU module: 3D1H) is not stored in SD1595.	U	New	QnPRH
SM1596	Memory copy to other system status flag	OFF: Memory copy not executed ON: Memory copy executed	• This relay is on during memory copy from the control system to the standby system. • This relay turns off when memory copy is complete.	S (Starting to copy/finish)	New	QnPRH
SM1597	Memory copy to other system completion flag	OFF: Memory copy not completed ON: Memory copy completed	This relay turns upon completion of memory copy from the control system to the standby system.	S (finish)/U	New	QnPRH
SM1598	Copy contents of standard ROM during memory copy	OFF: Copy standard ROM data ON: Standard ROM data is not copied	If set to on by user, the standard ROM data is not copied to the other system while memory copy is executing.	U	New	QnPRH

Tab. A-53: *Special relays (12): Redundant system (host system CPU information)*

A.6.13 Redundant system (other system CPU information)

The special relays SM1600 to SM1649 store diagnostic information and system information of other system CPU.

These special relays are valid only when the redundant system is in backup mode and is invalid in separate mode. SM1600 to SM1649 are set to OFF for stand-alone systems.

Number	Name	Meaning	Description	Set by (if set)	Corresponding host system special relay	Valid for:
SM1600	Other system error flag	OFF: No error ON: Error	• This relay turns on if an error is detected by error check for redundant system. (This relay turns on when any of the SD1600 bits turns on.) • This relay turns off when an error is cleared.	S (Every END processing)	—	QnPRH
SM1610	Other system diagnostics error	OFF: No error ON: Error	• This relay turns ON if a diagnostic error occurs in the CPU module in the other system. (Also turns ON when an annunciator turns ON and when an error is detected by the CHK instruction.) • The SM0 status for the CPU module in the other system is reflected.		SM0	QnPRH
SM1611	Other systems self diagnostics error.	OFF: No self diagnostics error occurred ON: Self diagnostics error occurred	• This relay turns on if a self-diagnostics error occurred in the CPU module in the other system. (Excluding error detections by an annunciator and the CHK instruction.) • The SM1 status for the CPU module in the other system is reflected.		SM1	QnPRH
SM1615	Other system common error information	OFF: No common error information present ON: Common error information present	• This relay turns on when there is error common information data for an error occurred in the CPU module in the other system. • The SM5 status for the CPU module in the other system is reflected.		SM5	QnPRH
SM1626	Error individual information for other systems	OFF: No individual error information present ON: Individual error information present	• This relay turns on when there is error individual information for an error occurred in the CPU module in the other system. • The SM16 status for the CPU module in the other system is reflected.		SM16	QnPRH
SM1649	Standby system cancel error flag	OFF to ON: Cancels error of standby system	This relay is turned on from off to clear a continuation error occurred in the standby system. Use SD1649 to specify the error code of the error to be canceled.	U	—	QnPRH

Tab. A-54: *Special relays (13): Redundant system (other system CPU information)*

A.6.14 Redundant system (tracking information)

The special relays SM1700 to SM1799 are valid when the redundant system is in backup mode or in separate mode. SM1700 to SM1799 are set to OFF for stand-alone systems.

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1700	Transfer trigger completion flag	OFF: Transfer not completed ON: Transfer completed	This relay remains on for one scan upon completion of a transfer for any of the blocks 1 to 64.	S (Status change)	New	QnPRH
SM1709	Manual system switching disable/ enable setting during online program change redundant tracking	OFF: Manual system switching disabled ON: Manual system switching enabled (Disable canceled)	• This relay is turned from off to on to enable the user to switch a system during online program change for redundancy. After the manual system switching disable status is canceled, the system automatically turns off SM1709. • A system can be switched even a online program change for redundancy is being performed and regardless of the status of this relay, if the reason for the switching is any of the following: - Power-off - Reset - Hardware failure - CPU stop error • The system switching disable status can also be canceled by this relay during the following states. - Multiple-block online program change redundant tracking execution status - File batch online program change redundant tracking execution status	S (Request)/ U	New	QnPRH
SM1710	Transfer tracking data during online program change enable flag	OFF: No device tracking ON: Transfer device memory	• This relay specifies whether to execute a tracking transfer for the following control data during online program change for redundancy. - Device memory (Including SMs and SDs that automatically execute a tracking transfer) - PIDINIT information, S.PIDINIT information, SFC information • SM1710 can be also used to specify whether to enable a tracking transfer while multiple-block online program change redundant tracking and while file batch online program change redundant tracking. • SM1710 is transferred from the control system to the standby system by tracking transfer.	U	New	QnPRH

Tab. A-55: *Special relays (14): Redundant system (tracking information)*

Number	Name	Meaning	Description			Set by (if set)	ACPU M9□□□	Valid for:
SM1712	Transfer trigger completion flag	OFF: Transfer uncompleted ON: Transfer completed	SM1712	Block 1	This relay turns on only during one scan upon completion of a transfer for the relevant block.	S (Status change)	New	QnPRH
SM1713			SM1713	Block 2				
SM1714			SM1714	Block 3				
SM1715			SM1715	Block 4				
SM1716			SM1716	Block 5				
SM1717			SM1717	Block 6				
SM1718			SM1718	Block 7				
SM1719			SM1719	Block 8				
SM1720			SM1720	Block 9				
SM1721			SM1721	Block 10				
SM1722			SM1722	Block 11				
SM1723			SM1723	Block 12				
SM1724			SM1724	Block 13				
SM1725			SM1725	Block 14				
SM1726			SM1726	Block 15				
SM1727			SM1727	Block 16				
SM1728			SM1728	Block 17				
SM1729			SM1729	Block 18				
SM1730			SM1730	Block 19				
SM1731			SM1731	Block 20				
SM1732			SM1732	Block 21				
SM1733			SM1733	Block 22				
SM1734			SM1734	Block 23				
SM1735			SM1735	Block 24				
SM1736			SM1736	Block 25				
SM1737			SM1737	Block 26				
SM1738			SM1738	Block 27				
SM1739			SM1739	Block 28				
SM1740			SM1740	Block 29				
SM1741			SM1741	Block 30				
SM1742			SM1742	Block 31				
SM1743			SM1743	Block 32				
SM1744			SM1744	Block 33				
SM1745			SM1745	Block 34				
SM1746			SM1746	Block 35				
SM1747			SM1747	Block 36				

Tab. A-55: *Special relays (14): Redundant system (tracking information)*

Number	Name	Meaning	Description			Set by (if set)	ACPU M9□□□	Valid for:
SM1748	Transfer trigger completion flag	OFF: Transfer uncompleted ON: Transfer completed	SM1748	Block 37	This relay turns on only for one scan upon completion of a transfer for the relevant block.	S (Status change)	New	QnPRH
SM1749			SM1749	Block 38				
SM1750			SM1750	Block 39				
SM1751			SM1751	Block 40				
SM1752			SM1752	Block 41				
SM1753			SM1753	Block 42				
SM1754			SM1754	Block 43				
SM1755			SM1755	Block 44				
SM1756			SM1756	Block 45				
SM1757			SM1757	Block 46				
SM1758			SM1758	Block 47				
SM1759			SM1759	Block 48				
SM1760			SM1760	Block 49				
SM1761			SM1761	Block 50				
SM1762			SM1762	Block 51				
SM1763			SM1763	Block 52				
SM1764			SM1764	Block 53				
SM1765			SM1765	Block 54				
SM1766			SM1766	Block 55				
SM1767			SM1767	Block 56				
SM1768			SM1768	Block 57				
SM1769			SM1769	Block 58				
SM1770			SM1770	Block 59				
SM1771			SM1771	Block 60				
SM1772			SM1772	Block 61				
SM1773			SM1773	Block 62				
SM1774			SM1774	Block 63				
SM1775			SM1775	Block 64				

Tab. A-55: *Special relays (14): Redundant system (tracking information)*

A.6.15 Redundant power supply module information

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1780	Power supply off detection flag	OFF: No redundant power supply module with input power OFF detected ON: Redundant power supply module with input power OFF detected	• Turns ON when one or more redundant power supply modules with input power OFF are detected. • Turns on if any of SD1780 bits is on. • Turns off if all bits of SD1780 are off. • This relay turns off when the main base unit is not the redundant main base unit (Q38RB). • When the multiple CPU system is configured, the flags are stored only to the CPU No.1.	S (Every END processing)	New	Qn(H) [3] QnPH [3] QnPRH QnU [4]
SM1781	Power supply failure detection flag	OFF: No faulty redundant power supply module detected ON: Faulty redundant power supply module detected	• Turns ON when one or more faulty redundant power supply modules are detected. • Turns on if any of SD1781 bits is on. • Turns off if all bits of SD1781 are off. • This relay turns off when the main base unit is not the redundant main base unit (Q38RB). • When the multiple CPU system is configured, the flags are stored only to the CPU No.1.	S (Every END processing)	New	Qn(H) [3] QnPH [3] QnPRH QnU [4]
SM1782	Momentary power failure detection flag for power supply 1 [1]	OFF: No momentary power failure detected ON: Momentary power failure detected	• Turns ON when a momentary power failure of the input power supply to the power supply 1 or 2 is detected one or more times. After turning on, this relay remains on even if the power supply recovers from the momentary power failure. • Turns off the flags (SM1782 and SM1783) of the power supply 1 and 2 when the CPU module starts. • When the input power to one of the redundant power supply modules turns OFF the corresponding flag turns OFF. • This relay turns off when the main base unit is not the redundant main base unit (Q38RB). • When the multiple CPU system is configured, the flags are stored only to the CPU No.1.	S (Every END processing)	New	Qn(H) [3] QnPH [3] QnPRH QnU [4]
SM1783	Momentary power failure detection flag for power supply 2 [2]			S (Every END processing)	New	Qn(H) [3] QnPH [3] QnPRH QnU [4]

Tab. A-56: *Special relays (15): Redundant power supply module information*

[1] The "power supply 1" indicates the redundant power supply module mounted on the POWER 1 slot of the redundant base unit (Q38RB/Q68RB/Q65WRB).

[2] The "power supply 2" indicates the redundant power supply module mounted on the POWER 2 slot of the redundant base unit (Q38RB/Q68RB/Q65WRB).

[3] The module whose first 5 digits of serial No. is "04012" or higher. However, for the multiple CPU system configuration, this applies to all CPU modules whose first 5 digits of serial No. are "07032" or higher.

[4] The module whose first 5 digits of serial No. is "10042" or higher.

A.6.16 Built-in I/O function

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1840	Axis 1 busy	OFF: Not busy ON: Busy	• This relay turns on when positioning control, OPR control, JOG operation, or absolute position restoration is started. This relay turns off when each control is completed. In positioning control, this relay turns off when the axis 1 decelerates and stops, and then "dwell time" elapsed. (This relay remains on while positioning control is being performed.) • This relay turns off when each control is ended due to such as an error or stop operation.	S (Every END processing)	New	LCPU
SM1841	Axis 1 positioning completion	OFF: Not completed ON: Completed	• This relay turns on when OPR control, position control, or absolute position restoration is completed. • This relay turns off when OPR control, positioning control, absolute position restoration, or JOG operation is started. • This relay remains off when JOG operation is completed. • This relay remains off when position control is stopped.	S (Instruction execution/ Status change)	New	LCPU
SM1842	Axis 1 OPR request	OFF: Machine OPR control completed ON: Machine OPR control started	• This relay turns on when the CPU module is powered on, is reset, or is set from STOP to RUN; or the drive unit ready signal turns off; or machine OPR control is started. • This relay turns off when machine OPR control is completed.	S (Every END processing)	New	LCPU
SM1843	Axis 1 OPR completion	OFF: Not completed ON: Completed	• This relay turns on when machine OPR control is completed. • This relay turns off when OPR control, positioning control, absolute position restoration, or JOG operation is started; or the CPU module is set from STOP to RUN; or the drive unit ready signal turns off.	S (Instruction execution/ Status change)	New	LCPU
SM1844	Axis 1 speed 0	OFF: Operating at speed other than 0 ON: Operating at speed 0	• This relay turns on when JOG operation or speed control in speed/position switching control set at a speed of "0" is started. • This relay turns on when speed is changed with a new speed value of "0", and turns off when speed is changed with a new speed value other than "0". • This relay turns off when SM1840 turns off.	S (Every END processing)	New	LCPU
SM1845	Axis 1 error	OFF: No error ON: Error	• This relay turns on if an error occurs. • The present error can be checked by SD1845. • This relay is turned off by turning on SM1850.		New	LCPU
SM1846	Axis 1 warning	OFF: No warning ON: Warning	• This relay turns on if a warning occurs. • The present warning can be checked by SD1846. • This relay is turned off by turning on SM1850.		New	LCPU
SM1847	Axis 1 start in busy status	OFF: No start attempted in busy status ON: Start attempted in busy status	• This relay turns on when positioning control, OPR control, JOG operation, or absolute position restoration is attempted while the axis 1 is in the busy status. The executed start instruction will be ignored. • This relay is reset by the user.	S (Instruction execution) /U	New	LCPU
SM1848	Axis 1 start instruction	OFF: Not executed ON: Being executed	• This relay turns on when positioning control by the start instruction (IPPSTRT1(P), IPDSTRT1(P), IPSIMUL(P), IPABRST1), JOG operation by the JOG start instruction (IPJOG1), or OPR control by the OPR start instruction (IPOPR1(P)) is started. • This relay turns off when positioning control, OPR control, or JOG operation is completed.	S (Instruction execution/ Status change)	New	LCPU

Tab. A-57: *Special relays (16): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1850	Axis 1 error reset	OFF → ON: Resets the Axis 1 error. OFF: Clears the reset status.	• Turning on this relay will turn off SM1845 and SM1846 and will clear the SD1845 and SD1846 values to "0". • Even if this relay is turned on, SM1845 will not turn off and the SD1845 value will not be cleared to "0" until SM1840 turns off.	U	New	LCPU
SM1851	Axis 1 OPR request off	OFF → ON: Axis 1 OPR request OFF: Cleared	Turning on this relay will forcibly turn off SM1842.			
SM1852	Axis 1 speed/ position switching	OFF: Disabled ON: Enabled	This relay stores whether to enable switching from speed control to position control in speed/position switching control.			
SM1860	Axis 2 busy	OFF: Not busy ON: Busy	• This relay turns on when positioning control, OPR control, JOG operation, or absolute position restoration is started. This relay turns off when each control is completed. In positioning control, this relay turns off when the axis 2 decelerates and stops, and then "dwell time" elapsed. (This relay remains on while positioning control is being performed.) • This relay turns off when each control is ended due to such as an error or stop operation.	S (Every END processing)	New	LCPU
SM1861	Axis 2 positioning completion	OFF: Not completed ON: Completed	• This relay turns on when OPR control, position control, or absolute position restoration is completed. • This relay turns off when OPR control, positioning control, absolute position restoration, or JOG operation is started. • This relay remains off when JOG operation is completed. • This relay remains off when position control is stopped.	S (Instruction execution/ Status change)	New	LCPU
SM1862	Axis 2 OPR request	OFF: Machine OPR control completed ON: Machine OPR control started	• This relay turns on when the CPU module is powered on, is reset, or is set from STOP to RUN; or the drive unit ready signal turns off; or machine OPR control is started. • This relay turns off when machine OPR control is completed.	S (Every END processing)	New	LCPU
SM1863	Axis 2 OPR completion	OFF: Not completed ON: Completed	• This relay turns on when machine OPR control is completed. • This relay turns off when OPR control, positioning control, absolute position restoration, or JOG operation is started; or the CPU module is set from STOP to RUN; or the drive unit ready signal turns off.	S (Instruction execution/ Status change)	New	LCPU
SM1864	Axis 2 speed 0	OFF: Operating at speed other than 0 ON: Operating at speed 0	• This relay turns on when JOG operation or speed control in speed/position switching control set at a speed of "0" is started. • This relay turns on when speed is changed with a new speed value of "0", and turns off when speed is changed with a new speed value other than "0". • This relay turns off when SM1860 turns off.	S (Every END processing)	New	LCPU
SM1865	Axis 2 error	OFF: No error ON: Error	• This relay turns on if an error occurs. • The present error can be checked by SD1865. • This relay is turned off by turning on SM1870.	S (Every END processing)	New	LCPU
SM1866	Axis 2 warning	OFF: No warning ON: Warning	• This relay turns on if a warning occurs. • The present warning can be checked by SD1866. • This relay is turned off by turning on SM1870.	S (Every END processing)	New	LCPU
SM1867	Axis 2 start in busy status	OFF: No start attempted in busy status ON: Start attempted in busy status	• This relay turns on when positioning control, OPR control, JOG operation, or absolute position restoration is attempted while the axis 2 is in the busy status. The executed start instruction will be ignored. • This relay is reset by the user.	S (Instruction execution) /U	New	LCPU
SM1868	Axis 2 start instruction	OFF: Not executed ON: Being executed	• This relay turns on when positioning control by the start instruction (IPPSTRT2(P), IPDSTRT2(P), IPSIMUL(P), IPABRST2), JOG operation by the JOG start instruction (IPJOG2), or OPR control by the OPR start instruction (IPOPR2(P)) is started. • This relay turns off when positioning control, OPR control, or JOG operation is completed.	S (Instruction execution/ Status change)	New	LCPU

Tab. A-57: *Special relays (16): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1870	Axis 2 error reset	OFF → ON: Resets the Axis 2 error. OFF: Clears the reset status.	• Turning on this relay will turn off SM1865 and SM1866 and will clear the SD1865 and SD1866 values to "0". • Even if this relay is turned on, SM1865 will not turn off and the SD1865 value will not be cleared to "0" until SM1860 turns off.	U	New	LCPU
SM1871	Axis 2 OPR request off	OFF → ON: Axis 2 OPR request OFF: Cleared	Turning on this relay will forcibly turn off SM1862.		New	LCPU
SM1872	Axis 2 speed/ position switching	OFF: Disabled ON: Enabled	This relay stores whether to enable switching from speed control to position control in speed/position switching control.		New	LCPU
SM1880	CH1 counter value greater (No.1)	OFF: Coincidence point (No.1) or smaller ON: Greater than coincidence point (No.1)	• This relay turns on when "current value of CH1 > coincidence output No.1 point setting value" is met. • This relay turns off when "current value of CH1 <= coincidence output No.1 point setting value" is met.	S (Every END processing)	New	LCPU
SM1881	CH1 counter value coincidence (No.1)	OFF: Not detected ON: Detected	• This relay turns on when "current value of CH1 = coincidence output No.1 point setting value" is met. • This relay is turned off by turning on CH1 coincidence signal No.1 reset command.	S (Status change/ Every END processing)	New	LCPU
SM1882	CH1 counter value smaller (No.1)	OFF: Coincidence point (No.1) or greater ON: Smaller than coincidence point (No.1)	• This relay turns on when "current value of CH1 < coincidence output No.1 point setting value" is met. • This relay turns off when "current value of CH1 >= coincidence output No.1 point setting value" is met.	S (Every END processing)	New	LCPU
SM1883	CH1 counter value greater (No.2)	OFF: Coincidence point (No.2) or smaller ON: Greater than coincidence point (No.2)	• This relay turns on when "current value of CH1 > coincidence output No.2 point setting value" is met. • This relay turns off when "current value of CH1 <= coincidence output No.2 point setting value" is met.	S (Every END processing)	New	LCPU
SM1884	CH1 counter value coincidence (No.2)	OFF: Not detected ON: Detected	• This relay turns on when "current value of CH1 = coincidence output No.2 point setting value" is met. • This relay is turned off by turning on CH1 coincidence signal No.2 reset command.	S (Status change/ Every END processing)	New	LCPU
SM1885	CH1 counter value smaller (No.2)	OFF: Coincidence point (No.2) or greater ON: Smaller than coincidence point (No.2)	• This relay turns on when "current value of CH1 < coincidence output No.2 point setting value" is met. • This relay turns off when "current value of CH1 >= coincidence output No.2 point setting value" is met.	S (Every END processing)	New	LCPU
SM1886	CH1 external preset (phase Z) request detection	OFF: Not detected ON: Detected	• This relay turns on when a preset request by phase Z (preset) terminal of CH1 is detected. • This relay is turned off by turning on CH1 external preset (phase Z) request detection clear command.	S (Every END processing)	New	LCPU
SM1887	CH1 error	OFF: No error ON: Error	• This relay turns on if the CH1 error occurs. • This relay turns off when an error cause is removed and CH1 error reset command is turned on.	S (Every END processing)	New	LCPU
SM1888	CH1 warning	OFF: No warning ON: Warning	• This relay turns on if a warning occurs in CH1. • This relay turns off when a warning cause is removed and CH1 error reset command is turned on.	S (Every END processing)	New	LCPU
SM1890	CH1 coincidence signal No.1 reset command	Resets CH1 counter value coincidence No.1.	• This relay is turned on to reset CH1 counter value coincidence No.1. • The command is valid while this relay is on. • The on time must be held for at least 2ms.	U	New	LCPU
SM1891	CH1 coincidence signal No.2 reset command	Resets CH1 counter value coincidence No.2.	• This relay is turned on to reset CH1 counter value coincidence No.2. • The command is valid while this relay is on. • The on time must be held for at least 2ms.	U	New	LCPU

Tab. A-57: *Special relays (16): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1892	CH1 coincidence output enable command	Controls outputs from CH1 coincidence output No.1 and No.2 terminals.	• This relay is turned on to perform coincidence output from CH1 coincidence output No.1 and CH1 coincidence output No.2 terminals. • The command is valid while this relay is on.	U	New	LCPU
SM1893	CH1 preset command	Presets the counter value.	• This relay is turned on to preset the counter value. • The command is valid at the rise of this relay (from OFF to ON). • The on and off time must be held for at least 2 ms.	U	New	LCPU
SM1894	CH1 count down command	Counts down pulses.	• This relay is turned on to count down pulses. • The command is valid while the Pulse input mode is either 1-phase multiple of n or 1-phase multiple of n (A phase only). • The command is valid while this relay is on.	U	New	LCPU
SM1895	CH1 count enable command	Starts counting.	• This relay is turned on to start counting. • The command is valid while this relay is on.	U	New	LCPU
SM1896	CH1 counter function selection start command	Starts the selected counter function.	• This relay is turned on to start the selected counter function. • When the count disabling function is selected, the command is valid while this relay is on. • When the latch counter function or the sampling counter function is selected, the command is valid at the rise of this relay (from OFF to ON). The on time must be held for at least 2 ms. • When the count disabling/preset function or the latch counter/preset function is selected, the command is invalid.	U	New	LCPU
SM1897	CH1 external preset (phase Z) request detection reset command	Resets CH1 external preset (phase Z) request detection.	• This relay is turned on to reset CH1 external preset (phase Z) request detection. • The command is valid at the rise of this relay (from OFF to ON). • The on and off time must be held for at least 2 ms.	U	New	LCPU
SM1898	CH1 pulse measurement start command	Starts pulse measurement.	• This relay is turned on to measure pulses. • The command is valid while this relay is on.	U	New	LCPU
SM1899	CH1 error reset command	Resets the CH1 error.	• This relay is turned on to reset the CH1 error. • The command is valid at the rise of this relay (from OFF to ON). • The on and off time must be held for at least 2 ms.	U	New	LCPU
SM1900	CH2 counter value greater (No.1)	OFF: Coincidence point (No.1) or smaller ON: Greater than coincidence point (No.1)	• This relay turns on when "current value of CH2 > coincidence output No.1 point setting value" is met. • This relay turns off when "current value of CH2 <= coincidence output No.1 point setting value" is met.	S (Every END processing)	New	LCPU
SM1901	CH2 counter value coincidence (No.1)	OFF: Not detected ON: Detected	• This relay turns on when "current value of CH2 = coincidence output No.1 point setting value" is met. • This relay is turned off by turning on CH2 coincidence signal No.1 reset command.	S (Status change/ Every END processing)	New	LCPU
SM1902	CH2 counter value smaller (No.1)	OFF: Coincidence point (No.1) or greater ON: Smaller than coincidence point (No.1)	• This relay turns on when "current value of CH2 < coincidence output No.1 point setting value" is met. • This relay turns off when "current value of CH2 >= coincidence output No.1 point setting value" is met.	S (Every END processing)	New	LCPU
SM1903	CH2 counter value greater (No.2)	OFF: Coincidence point (No.2) or smaller ON: Greater than coincidence point (No.2)	• This relay turns on when "current value of CH2 > coincidence output No.2 point setting value" is met. • This relay turns off when "current value of CH2 <= coincidence output No.2 point setting value" is met.			
SM1904	CH2 counter value coincidence (No.2)	OFF: Not detected ON: Detected	• This relay turns on when "current value of CH2 = coincidence output No.2 point setting value" is met. • This relay is turned off by turning on CH2 coincidence signal No.2 reset command.	S (Status change/ Every END processing)	New	LCPU

Tab. A-57: *Special relays (16): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1905	CH2 counter value smaller (No.2)	OFF: Coincidence point (No.2) or greater ON: Smaller than coincidence point (No.2)	• This relay turns on when "current value of CH2 < coincidence output No.2 point setting value" is met. • This relay turns off when "current value of CH2 >= coincidence output No.2 point setting value" is met.	S (Every END processing)	New	LCPU
SM1906	CH2 external preset (phase Z) request detection	OFF: Not detected ON: Detected	• This relay turns on when a preset request by phase Z (preset) terminal of CH2 is detected. • This relay is turned off by turning on CH2 external preset (phase Z) request detection clear command.			
SM1907	CH2 Error	OFF: No error ON: Error	• This relay turns on if the CH2 error occurs. • This relay turns off when an error cause is removed and CH2 error reset command is turned on.	S (Every END processing)	New	LCPU
SM1908	CH2 warning	OFF: No warning ON: Warning	• This relay turns on if a warning occurs in CH2. • This relay turns off when a warning cause is removed and CH2 error reset command is turned on.			
SM1910	CH2 coincidence signal No.1 reset command	Resets CH2 counter value coincidence No.1.	• This relay is turned on to reset CH2 counter value coincidence No.1. • The command is valid while this relay is on. • The on time must be held for at least 2ms.	U	New	LCPU
SM1911	CH2 coincidence signal No.2 reset command	Resets CH2 counter value coincidence No.2.	• This relay is turned on to reset CH2 counter value coincidence No.2. • The command is valid while this relay is on. • The on time must be held for at least 2ms.	U	New	LCPU
SM1912	CH2 coincidence output enable command	Controls outputs from CH2 coincidence output No.1 and No.2 terminals.	• This relay is turned on to perform coincidence output from CH2 coincidence output No.1 and CH2 coincidence output No.2 terminals. • The command is valid while this relay is on.	U	New	LCPU
SM1913	CH2 preset command	Presets the counter value.	• This relay is turned on to preset the counter value. • The command is valid at the rise of this relay (from OFF to ON). • The on and off time must be held for at least 2ms.	U	New	LCPU
SM1914	CH2 count down command	Counts down pulses.	• This relay is turned on to count down pulses. • The command is valid while the Pulse input mode is either 1-phase multiple of n or 1-phase multiple of n (A phase only). • The command is valid while this relay is on.	U	New	LCPU
SM1915	CH2 count enable command	Starts counting.	• This relay is turned on to start counting. • The command is valid while this relay is on.	U	New	LCPU
SM1916	CH2 counter function selection start command	Starts the selected counter function.	• This relay is turned on to start the selected counter function. • When the count disabling function is selected, the command is valid while this relay is on. • When the latch counter function or the sampling counter function is selected, the command is valid at the rise of this relay (from OFF to ON). The on time must be held for at least 2 ms. • When the count disabling/preset function or the latch counter/preset function is selected, the command is invalid.	U	New	LCPU
SM1917	CH2 external preset (phase Z) request detection reset command	Resets CH2 external preset (phase Z) request detection.	• This relay is turned on to reset CH2 external preset (phase Z) request detection. • The command is valid at the rise of this relay (from OFF to ON). • The on and off time must be held for at least 2ms.	U	New	LCPU
SM1918	CH2 pulse measurement start command	Starts pulse measurement.	• This relay is turned on to measure pulses. • The command is valid while this relay is on.	U	New	LCPU
SM1919	CH2 error reset command	Resets the CH2 error.	• This relay is turned on to reset the CH2 error. • The command is valid at the rise of this relay (from OFF to ON). • The on and off time must be held for at least 2 ms.	U	New	LCPU

Tab. A-57: *Special relays (16): Built-in I/O function*

A.6.17 Data logging

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1940	Data logging setting No.1 Data logging preparation	OFF: Not ready ON: Ready	This relay turns on when the system is ready for data logging. This relay remains on even after data logging is suspended. However, this relay turns off when data logging is stopped.	S (Initial)	New	LCPU
SM1941	Data logging setting No.1 Data logging start	OFF: Pause ON: Start	This relay is turned on to start data logging and is turned off to suspend data logging. (The related special relays will all turn off.)	U	New	LCPU
SM1942	Data logging setting No.1 Data logging collection	OFF: Not being collected ON: Being collected	This relay is on while data logging is being collected.	S (Status change)	New	LCPU
SM1943	Data logging setting No.1 Data logging end	OFF: Not ended ON: Ended	This relay turns on when data logging is ended. [Continuous is set for Logging type] The corresponding bit turns on when data logging is ended after data have been written by the number of storable files (Stop is set for Operation occurring when number of saved files is exceeded). [Trigger is set for Logging type] The corresponding bit turns on when the trigger condition is met, data are collected by the number of set times, and then the data are written to the SD memory card. This relay also turns on if an error occurs during data logging (except data logging error occurred by the execution of online change).	S (Status change)	New	LCPU
SM1944	Data logging setting No.1 Data logging trigger	OFF → ON: Triggered	• This relay turns on when the specified trigger condition is met. • This relay is turned on to meet the trigger condition.	S (Status change)/U	New	LCPU
SM1945	Data logging setting No.1 After data logging trigger	OFF: Not triggered ON: Triggered	This relay turns on after trigger logging is triggered. This relay remains on even after data logging is completed. This relay turns off when trigger logging is suspended or stopped.	S (Status change)	New	LCPU
SM1946	Data logging setting No.1 Data logging error	OFF: No error ON: Error	This relay turns on if a data logging error occurs. This relay is turned off by the registration of the setting or a stop command from LCPU Logging Configuration Tool.	S (Error)	New	LCPU
SM1947	Data logging setting No.1 Data storage in SD memory card	OFF: Not stored ON: Being stored	This relay is on while buffer memory data are being stored to a SD memory card by data logging.	S (Status change)	New	LCPU

Tab. A-58: *Special relays (17): Data logging*

Number	Name	Meaning	Description	Set by (if set)	ACPU M9□□□	Valid for:
SM1950 to SM1957	Data logging setting No. 2	Same as in data logging setting No. 1	Same as in data logging setting No. 1 (SM1940 to SM1947)	Same as in data logging setting No. 1	New	LCPU
SM1960 to SM1967	Data logging setting No. 3					
SM1970 to SM1977	Data logging setting No. 4					
SM1980 to SM1987	Data logging setting No. 5					
SM1990 to SM1997	Data logging setting No. 6					
SM2000 to SM2007	Data logging setting No. 7					
SM2010 to SM2017	Data logging setting No. 8					
SM2020 to SM2027	Data logging setting No. 9					
SM2030 to SM2037	Data logging setting No. 10					

Tab. A-58: *Special relays (17): Data logging*

A.7 Table of special registers

The special registers are internal registers with fixed applications in the programmable controller.

Therefore, they cannot be used like other special registers in a sequence program. However, data can be written to these registers in order to control the CPU module. Data is usually stored in binary format except another format is required.

The table below describes the meanings of the headings in the following table:

Item	Meaning
Number	Indicates the number of the special register.
Name	Indicates the name of the special register.
Meaning	Contains the function of the special register in brief.
Description	Contains a detailed description of the register.
Set by (if set)	Indicates whether the special relay is set by the system or the user. <Set by> S : Set by the system U : Set by the user (using a program, programming tool, GOT, or test operation frosm other external devices) S/U : Set by the system or user Is indicated only if the system set the status. <if set> END processing : Set during END processing Initial : Set during initial processing (Power ON, STOP->RUN) Status change : Set after status change Error : Set after error Instruction execution : Set during instruction execution Request : Set for user request (through SM, etc.) When condition occurs : Set when the condition is triggered When system is switched : Set when the system is switched (between the control system and the standby system) When RUN/STOP/RESET switch changed : Set when the RUN/STOP/RESET switch is changed Card removal : Set when a memory card is inserted or removed At write : Set when data is written to the CPU module
ACPU D9□□□	Indicates special register D9□□□ corresponding to the ACPU (Change and notation when contents changed. Incompatible with the Q00J/Q00/Q01 and QnPRH.) Items indicated as "New" were newly added to the QCPU or LCPU.
Valid for:	Indicates the corresponding CPU: • QCPU: All the System Q CPU modules • Q00J/Q00/Q01: Basic model QCPU • Qn(H): High Performance model QCPU • QnPH: Process CPU • QnPRH: Redundant CPU • QnU: Universal model QCPU • Q00UJ/Q00U/Q01U: Q00UJCPU, Q00UCPU, and Q01UCPU • LCPU: All the L series CPU modules • CPU module model: Only the specified model (Example: Q02UCPU, L26CPU-BT)

For detailed information on the following topic refer to the manuals:

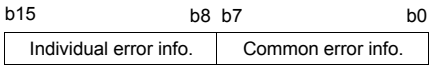
- Networks → Manuals for each network module
- SFC → MELSEC-Q/L/QnA Programming Manual (SFC)

NOTE *Do not change the values of special register set by system using a program or by test operation. Doing so may result in system down or communication failure.*

A.7.1 Diagnostic information

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD0	Diagnostic errors		Error codes for errors found by diagnosis are stored as BIN data. Contents identical to latest fault history information.	S (Error)	D9008 format change	QCPU LCPU
SD1	Clock time for diagnosis error occurrence		Year (last two digits) and month that SD0 data was updated is stored as BCD 2-digit code. Example: **October 1995** **H9510** b15 b8 b7 b0 Year (0 to 99) Month (1 to 31)	S (Error)	New	QCPU LCPU
SD2			The day and hour that SD0 was updated is stored as BCD 2-digit code. Example: **10 p.m. on 25th** **H2510** b15 b8 b7 b0 Day (1 to 31) | Hour (0 to 23)			
SD3			The minute and second that SD0 data was updated is stored as BCD 2-digit code. Example: **35 min 48s** **H3548** b15 b8 b7 b0 Minute (1 to 60) | Second (1 to 60)			

Tab. A-59: *Special registers (1): Diagnostic information*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD4	Error information categories	Error information category code	Category codes which help indicate what type of information is being stored in the common information areas (SD5 through SD15) and the individual information areas (SD16 through SD26) are stored here.  The common information category codes store the following codes: 0: No error 1: Module No. (Slot No./CPU No./base No.) [1, 2)] 2: File name/Drive name 3: Time (value set) 4: Program error location 5: Reason(s) for system switching (for Redundant CPU) 6: Reason(s) for tracking size excess error (for Redundant CPU) 7: Base No./power supply No. (Except for the Universal model QCPU and LCPU having the serial No. (first five digits) of "10041" or lower) 8: Tracking transmission data classification (for Redundant CPU) 1 For a multiple CPU system, the module No. or CPU No. is stored according to an error. (To determine whether a storage value is a module No. or CPU No., refer to each error code.) CPU No. 1: 1, CPU No. 2: 2, CPU No. 3: 3, CPU No. 4: 4 2 For the LCPU, only a slot No. is stored. The individual information category codes store the following codes: 0: No error 1: (Open) 2: File name/Drive name 3: Time (value actually measured) 4: Program error location 5: Parameter number 6: Annunciator number 7: Check instruction malfunction number (Except for the Basic model QCPU, Universal model QCPU, and LCPU) 8: Reason(s) for system switching failure (for Redundant CPU) 9: Failure information (for LCPU) 12: File diagnostic information (for Universal model QCPU and LCPU) 13: Parameter No./CPU No. (for Universal model QCPU)	S (Error)	New	QCPU LCPU

Tab. A-59: *Special registers (1): Diagnostic information*

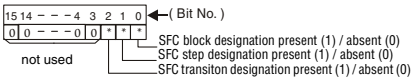
<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD5
SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td colspan="2">Error common information</td><td>

Common information corresponding to the error codes (SD0) is stored here.
The following ten types of information are stored here.
The error common information type can be determined by "common information category code" stored in SD4. (Values stored in "common information category code" correspond to the following (1) to (8).)

(1) Module No.
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SM5</td><td>Slot No./CPU No./Base No. 1, 2, 3, 4, 5)</td></tr>
<tr><td>SM6</td><td>I/O No. 6)</td></tr>
<tr><td>SM7
SM8
SM9
SM10
SM11
SM12
SM13
SM14
SM15</td><td>(Vacant)</td></tr>
</table>
1 For a multiple CPU system, the module No. or CPU No. is stored according to an error. (To determine whether a storage value is a module No. or CPU No., refer to each error code.)
CPU No. 1: 1, CPU No. 2: 2, CPU No. 3: 3, CPU No. 4: 4
2 If a fuse has been blown or an I/O module verification error occurs in a module on the MELSECNET/H remote I/O station, the network No. is stored in the upper 8 bits and the station No. is stored in the lower 8 bits. To determine a fuseblown module or a module where an I/O module verification error occurs, check the I/O No.
3 If an instruction is executed to the Basic model QCPU on the slot where the module cannot be mounted, "255" is stored in SD5.
4 The definitions of the base No. and slot No. are as follows:
[Base No.]
<table>
<tr><th>Base No.</th><th>Definition</th></tr>
<tr><td>0</td><td>Indicates the main base unit where a CPU module is mounted.</td></tr>
<tr><td>1 to 7</td><td>Indicates the extension base unit. The stage number setting made by the stage number setting connector on the extension base unit is the base No.
When stage number setting is extension 1: Base No. = 1
When stage number setting is extension 7: Base No. = 7</td></tr>
</table>
[Slot No.]
This number is used to identify the slot No. of a module where an error occurs. The "0" I/O slot (slot on the right of the CPU slot) on the main base unit is defined as "Slot No. = 0". The slot Nos. are assigned in sequence numbers in order of the main base unit and then the first extension base unit to 7th extension base unit. When the number of slots on base units has been set in the I/O assignment tab of the PLC Parameter dialog box, the slot Nos. are assigned by the number of set slots.
5 If a module is not mounted on any slots as set, FF_H is stored.
6 If $FFFF_H$ is stored in SD6 (I/O No.), this indicates that the I/O No. cannot be identified due to such as overlap of an I/O No. in the I/O assignment setting of the PLC Parameter dialog box. In this case, identify the error location using SD5.
</td><td>S
(Error)</td><td>New</td><td>QCPU
LCPU</td></tr>
</table>

Tab. A-59: *Special registers (1): Diagnostic information*

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD5
SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14

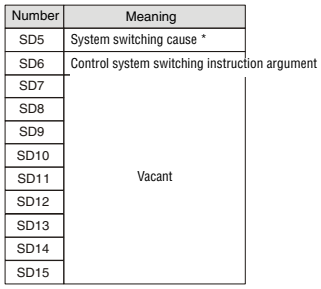
SD15</td><td colspan="2">Error common information</td><td>(2) File name/Drive name
Example:
File name = ABCDEFGH.IJK
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD5</td><td>Drive</td></tr>
<tr><td>SD6
SD7
SD8
SD9</td><td>File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD10</td><td>Extension 7) 2EH(.)</td></tr>
<tr><td>SD11</td><td>(ASCII code: 3 characters)</td></tr>
<tr><td>SD12
SD13
SD14
SD15</td><td>Vacant</td></tr>
</table>
b15 b0
<table>
<tr><td>B</td><td>A</td></tr>
<tr><td>D</td><td>C</td></tr>
<tr><td>F</td><td>E</td></tr>
<tr><td>H</td><td>G</td></tr>
<tr><td>I</td><td>.</td></tr>
<tr><td>K</td><td>J</td></tr>
</table>
(3) Time (value set)
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD5</td><td>Time: 1µs-steps (0 to 999 µs)</td></tr>
<tr><td>SD6</td><td>Time: 1ms-steps (0 to 65535 ms)</td></tr>
<tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>Vacant</td></tr>
</table>
(4) Program error location
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD5
SD6
SD7
SD8</td><td>File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD9</td><td>Extension 7) 2EH(.)</td></tr>
<tr><td>SD10</td><td>(ASCII code: 3 characters)</td></tr>
<tr><td>SD11</td><td>Pattern *</td></tr>
<tr><td>SD12</td><td>Block No.</td></tr>
<tr><td>SD13</td><td>Step / transition No.</td></tr>
<tr><td>SD14</td><td>Sequence step No. (L)</td></tr>
<tr><td>SD15</td><td>Sequence step No. (H)</td></tr>
</table>
* Contents of pattern data

</td><td>S
(Error)</td><td>New</td><td>QCPU
LCPU</td></tr>
</table>

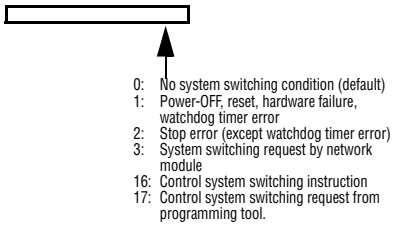
7) Meaning of the extensions:

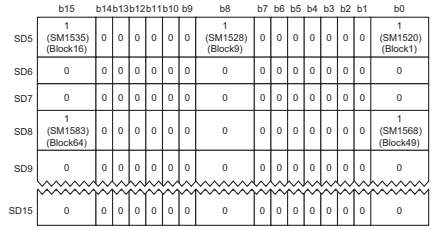
SD_n	SD_{n+1}		Extension Name	File type
Higher 8 bits	Lower 8 bits	Higher 8 bits		
51H	50H	41H	QPA	Parameters
51H	50H	47H	QPG	Program
51H	43H	44H	QCD	Device comment
51H	44H	49H	QDI	Initial device value
51H	44H	52H	QDR	File register
51H	44H	4CH	QDL	Local device (For High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU, and LCPU)
51H	54H	44H	QTD	Sampling trace data (For High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU, and LCPU)
51H	46H	44H	QFD	Breakdown history data (For High Performance model QCPU, Process CPU, and Redundant CPU)
51H	53H	54H	QST	SP.DEVST/S.DEVLD instruction file (for Universal model QCPU and LCPU)

Tab. A-59: *Special registers (1): Diagnostic information*

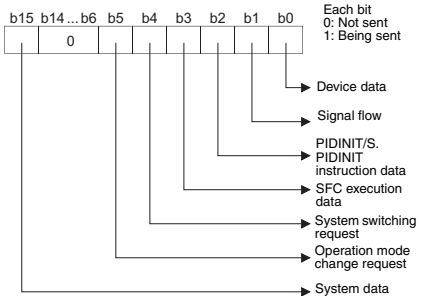
<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD5
SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14

SD15</td><td colspan="2">Error common information</td><td>(5) Reason(s) for system switching
 

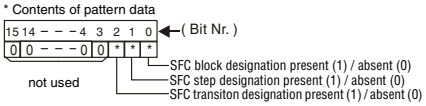
* Following shows the description:
 

(6) Reason(s) for tracking size excess error
The following shows block Nos. when data size that can be tracked (100K) is exceeded in the bit pattern of the corresponding special relay.
 
</td><td>S
(Error)</td><td>New</td><td>QnPRH</td></tr>
</table>

Tab. A-59: *Special registers (1): Diagnostic information*

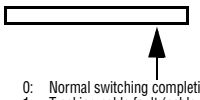
<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD5
SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14</td><td rowspan="2">Error common information</td><td rowspan="2"></td><td>(7) Base No./power supply No.
<table><tr><th>Number</th><th>Meaning</th></tr><tr><td>SD5</td><td>Base No.</td></tr><tr><td>SD6</td><td>Power supply No.</td></tr><tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>Vacant</td></tr></table>1: Power supply 1 fault
2: Power supply 2 fault
Redundant power supply module mounted on POWER 1 resp. POWER 2 slot of redundant base unit (Q38RB, Q68RB, Q65WRB)</td><td rowspan="2">S
(Error)</td><td rowspan="2">New</td><td>Qn(H) 1)
QnPH 1)
QnPRH
QnU 2)</td></tr>
<tr><td>SD15</td><td>(8) Tracking transmission data classification
This register stores a data type during tracking.
<table><tr><th>Number</th><th>Meaning</th></tr><tr><td>SD5</td><td>Data type *</td></tr><tr><td>SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>Vacant</td></tr></table>* Description of data type:
 
</td><td>QnPRH</td></tr>
</table>

Tab. A-59: *Special registers (1): Diagnostic information*

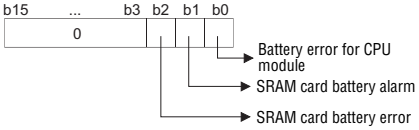
<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD16
SD17
SD18
SD19
SD20
SD21
SD22
SD23
SD24
SD25
SD26</td><td colspan="2">Error individual information</td><td>
• Individual information corresponding to the error codes (SD0) is stored here.
• The following eight types of information are stored here.
• The error individual information type can be determined by "individual information category code" stored in SD4. (Values stored in "individual information category code" correspond to the following (1) to (8), (12) and (13).
(1) Empty
(2) File name/Drive name
Example:
File name = ABCDEFGH.IJK
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16</td><td>Drive</td></tr>
<tr><td>SD17
SD18
SD19
SD20</td><td>File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD21</td><td>Extension [1] 2EH(.)</td></tr>
<tr><td>SD22</td><td>(ASCII code: 3 characters)</td></tr>
<tr><td>SD23
SD24
SD25
SD26</td><td>Vacant</td></tr>
</table>
b15 b0
<table>
<tr><td>B</td><td>A</td></tr>
<tr><td>D</td><td>C</td></tr>
<tr><td>F</td><td>E</td></tr>
<tr><td>H</td><td>G</td></tr>
<tr><td>I</td><td>.</td></tr>
<tr><td>K</td><td>J</td></tr>
</table>
(3) Time (value actually measured)
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16</td><td>Time: 1µs-steps (0 to 999 µs)</td></tr>
<tr><td>SD17</td><td>Time: 1ms-steps (0 to 65535 ms)</td></tr>
<tr><td>SD18
SD19
SD20
SD21
SD22
SD23
SD24
SD25
SD26</td><td>Vacant</td></tr>
</table>
(4) Program error location
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16
SD17
SD18
SD19</td><td>File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD20</td><td>Extension [1] 2EH (.)</td></tr>
<tr><td>SD21</td><td>(ASCII code: 3 characters)</td></tr>
<tr><td>SD22</td><td>Pattern *</td></tr>
<tr><td>SD23</td><td>Block No.</td></tr>
<tr><td>SD24</td><td>Step / transition No.</td></tr>
<tr><td>SD25</td><td>Sequence step No. (L)</td></tr>
<tr><td>SD26</td><td>Sequence step No. (H)</td></tr>
</table>
* Contents of pattern data


[1] For extension names, refer to table on page A-160.
</td><td>S
(Error)</td><td>New</td><td>QCPU
LCPU</td></tr>
</table>

Tab. A-59: *Special registers (1): Diagnostic information*

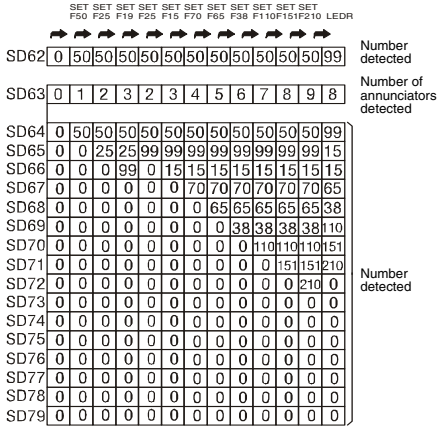
<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD26</td><td>Error individual information</td><td></td><td>
(5) Parameter number
(6) Annunciator number
(7) CHK instruction malfunction number
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16</td><td>Parameter No.*</td></tr>
<tr><td>SD17</td><td rowspan="10">Vacant</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td></tr>
<tr><td>SD22</td></tr>
<tr><td>SD23</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table>
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16</td><td>No.</td></tr>
<tr><td>SD17</td><td rowspan="10">Vacant</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td></tr>
<tr><td>SD22</td></tr>
<tr><td>SD23</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table>
* For details of the parameter No., refer to the following: User's Manual (Function Explanation, Program Fundamentals) for the CPU module used
(8) Reason(s) for system switching failure
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SM16</td><td>System switching prohibition condition *</td></tr>
<tr><td>SM17</td><td rowspan="10">(Vacant)</td></tr>
<tr><td>SM18</td></tr>
<tr><td>SM19</td></tr>
<tr><td>SM20</td></tr>
<tr><td>SM21</td></tr>
<tr><td>SM22</td></tr>
<tr><td>SM23</td></tr>
<tr><td>SM24</td></tr>
<tr><td>SM25</td></tr>
<tr><td>SM26</td></tr>
</table>
* Following shows the description:

0: Normal switching completion (default)
1: Tracking cable fault (cable removal, cable fault, internal circuit fault, hardware fault)
2: Hardware failure, power OFF, reset or watchdog timer error occurring in standby system
3: Hardware failure, power OFF, reset or watchdog timer error occurring in control system
4: Preparing for tracking
5: Time limit exceeded
6: Standby system is in stop error (except watchdog timer error)
7: Operation differs between two systems (in backup mode only)
8: During memory copy from control system to standby system
9: Online program change
10: Error detected by network module of standby system
11: System switching being executed
12: Online module change in progress
</td><td>S
(Error)</td><td>New</td><td>QCPU
LCPU</td></tr>
</table>

Tab. A-59: *Special registers (1): Diagnostic information*

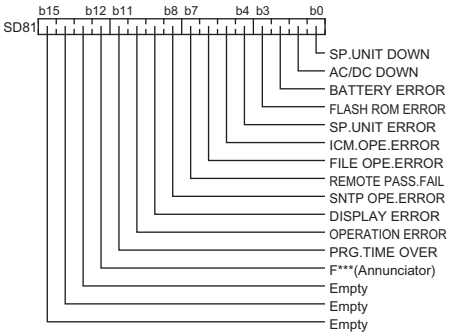
<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td rowspan="3">SD26</td><td rowspan="3">Error individual information</td><td rowspan="3"></td><td>(9) Failure information
<table><tr><th>Number</th><th>Meaning</th></tr><tr><td>SM16</td><td>Failure information 1</td></tr><tr><td>SM17</td><td>Failure information 2</td></tr><tr><td>SM18</td><td>Failure information 3</td></tr><tr><td>SM19</td><td>Failure information 4</td></tr><tr><td>SM20</td><td>Failure information 5</td></tr><tr><td>SM21</td><td>Failure information 6</td></tr><tr><td>SM22</td><td>Failure information 7</td></tr><tr><td>SM23</td><td>Failure information 8</td></tr><tr><td>SM24</td><td>Failure information 9</td></tr><tr><td>SM25</td><td>Failure information 10</td></tr><tr><td>SM26</td><td>Failure information 11</td></tr></table></td><td>S
(Error)</td><td rowspan="3">New</td><td>LCPU</td></tr>
<tr><td>(12) File diagnostic information
<table><tr><th>Number</th><th colspan="2">Meaning</th></tr><tr><td>SM16</td><td>Failure information (H)</td><td>Drive No. (L)</td></tr><tr><td>SM17</td><td colspan="2" rowspan="4">File name
(ASCII: 8 characters)</td></tr><tr><td>SM18</td></tr><tr><td>SM19</td></tr><tr><td>SM20</td></tr><tr><td>SM21</td><td>Extension 1)</td><td>2EH(.)</td></tr><tr><td>SM22</td><td colspan="2">(ASCII: 3 characters)</td></tr><tr><td>SM23</td><td colspan="2" rowspan="2">Failure information 2
(CRC value that is read)</td></tr><tr><td>SM24</td></tr><tr><td>SM25</td><td colspan="2" rowspan="2">Failure information 3
(CRC value that is calculated)</td></tr><tr><td>SM26</td></tr></table>1) For extension names, refer to table on page A-160.</td><td rowspan="2">U</td><td>QnU
LCPU</td></tr>
<tr><td>(13) Parameter No./CPU No.
<table><tr><th>Number</th><th>Meaning</th></tr><tr><td>SM16</td><td>Parameter No. *</td></tr><tr><td>SM17</td><td>CPU No. (1 to 4)</td></tr><tr><td>SM18</td><td rowspan="10">(Vacant)</td></tr><tr><td>SM19</td></tr><tr><td>SM20</td></tr><tr><td>SM21</td></tr><tr><td>SM22</td></tr><tr><td>SM23</td></tr><tr><td>SM24</td></tr><tr><td>SM25</td></tr><tr><td>SM26</td></tr></table>* For details of the parameter No., refer to the following: User's Manual (Function Explanation, Program Fundamentals) for the CPU module used</td><td>QnU</td></tr>
<tr><td>SD50</td><td>Error reset</td><td>Error number that performs error reset</td><td>Stores error number that performs error reset</td><td>U</td><td>New</td><td>QCPU
LCPU</td></tr>
<tr><td>SD51</td><td>Battery low latch</td><td>Bit pattern indicating where battery voltage drop occurred</td><td>All corresponding bits go ON when battery voltage drops. Subsequently, these remain ON even after battery voltage has been returned to normal.
 

Bits 1 and 2 are not available for the Basic model QCPU and LCPU.
If an alarm occurs, data can be held within the time specified for battery low.
The error indicates full discharge of a battery.</td><td>S
(Error)</td><td>New</td><td>QCPU
LCPU</td></tr>
</table>

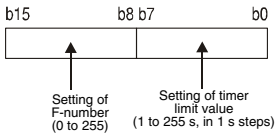
Tab. A-59: *Special registers (1): Diagnostic information*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD52	Battery low	Bit pattern indicating where battery voltage drop occurred	• Same configuration as SD51 above • After an alarm is detected (the alarm bit turns on), the alarm bit turns off if an error is detected (the error bit turns on). (Universal model QCPU only) • Subsequently, goes OFF when battery voltage is restored to normal.	S (Error)	New	QCPU LCPU
SD53	AC DOWN detection	Number of times for AC DOWN	• A value stored in this register is incremented by 1 whenever the input voltage falls to or below 85% (AC power)/65% (DC power) of the rating during operation of the CPU module. • The counter repeats increment and decrement of the value; (0 ==> 32767 ==> -32768 ==> 0)	S (Error)	D9005	QCPU LCPU
SD60	Blown fuse number	Number of module with blown fuse	Value stored here is the lowest station number of the module with the blown fuse, divided by 16.	S (Error)	D9000	QCPU
SD61	I/O module verification error	I/O module verification error module number	The lowest number of the module where the I/O module verification number took place.	S (Error)	D9002	QCPU LCPU
SD62	Annunciator number		The first annunciator number to be detected is stored here.	S (Instruction execution)	D9009	QCPU LCPU
SD63	Number of annunciators		Stores the number of annunciators searched.	S (Instruction execution)	D9124	
SD64	Table of detected annunciator numbers	Annunciator detection number	When F goes ON due to OUT F or SET F, the F numbers which go progressively ON from SD64 through SD79 are registered. F numbers turned OFF by RST F are deleted from SD64 to SD79, and are shifted to the data register following the data register where the deleted F numbers had been stored. Execution of the LEDR instruction shifts the contents of SD64 to SD79 up by one. After 16 annunciators have been detected, detection of the 17th will not be stored from SD64 through SD79. 	S (Instruction execution)	D9125 D9126 D9127 D9128 D9129 D9130 D9131 D9132 New	QCPU LCPU
SD65						
SD66						
SD67						
SD68						
SD69						
SD70						
SD71						
SD72						
SD74						
SD75						
SD76						
SD77						
SD78						
SD79						
SD80	CHK number		Error codes detected by the CHK instruction are stored as BCD code.	S (Instruction execution)	New	Qn(H) QnPH QnPRH

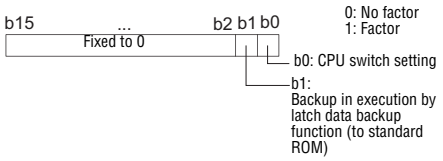
Tab. A-59: *Special registers (1): Diagnostic information*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD81	Continuation error cause		This register stores a continuation error cause  The SD82 bits are all empty.	S (Error)	New	LCPU
SD82						
SD84	Continuation error clear		This register stores a continuation error to be cleared in bit pattern. This register has the same bit pattern as that of SD81 and SD82.	U	New	LCPU
SD85						

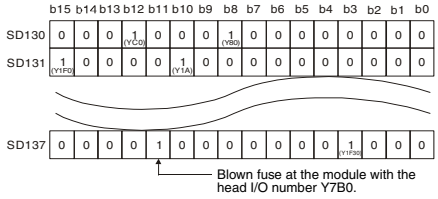
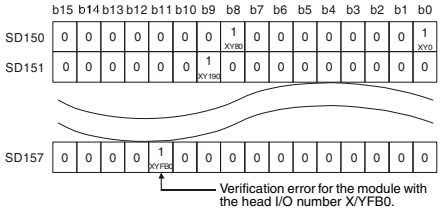
Tab. A-59: *Special registers (1): Diagnostic information*

Number	Name	Meaning	Description		Set by (if set)	ACPU register D9□□□	Valid for:
SD90	Step transition watchdog timer setting value (Enabled only when SFC program exists)	F number for timer set value and time over error	Corresponds to SM90	F numbers that are set ON at setting value of step transition watchdog timer and watchdog timer over errors.  Timer is started by turning SM90 through SM99 ON during active step, and if the transition conditions for the relevant steps are not met within the timer limits, the designated annunciator (F) will go ON.	U	D9108	Qn(H) QnPH QnPRH
SD91			Corresponds to SM91			D9109	
SD92			Corresponds to SM92			D9110	
SD93			Corresponds to SM93			D9111	
SD94			Corresponds to SM94			D9112	
SD95			Corresponds to SM95			D9113	
SD96			Corresponds to SM96			D9114	
SD97			Corresponds to SM97			New	
SD98			Corresponds to SM98			New	
SD99			Corresponds to SM99			New	

Tab. A-59: Special registers (1): Diagnostic information (continued)

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD100	Transmission speed	Stores the transmission speed specified in the serial communication setting.	K96: 9600 bps, K192: 19.2 kbps, K384: 38.4 kbps, K576: 57.6 kbps, K1152: 115.2 kbps	S (power on or reset)	New	Q00/Q01 Q00UJ Q00U Q01U Q02U 4)
SD101	Communication settings	Stores the settings for serial communication	Bit 4 = OFF: Without sumcheck Bit 4 = ON: With sumcheck Bit 5 = OFF: Online program correction disabled Bit 5 = ON: Online program correction enabled The other bits have no function.		New	
SD102	Message waiting time	Stores the waiting time specified in the serial communication setting.	0: No waiting time 1 to FH: Waiting time (unit: 10 ms) Default: 0		New	
SD105	CH1 transmission speed setting (RS-232)	Stores the present transmission speed.	K96: 9600 bps, K192: 19.2 kbps, K384: 38.4 kbps, K576: 57.6 kbps, K1152: 115.2 kbps This register holds a value stored in RS-232 connection even after communication in other than RS-232 connection is started. (When no connection is made, this register stores "1152".)	S	New	Qn(H) QnPH QnPRH QnU 3) LCPU
SD110	Data sending result	Stores the data sending result when the serial communication is used.	Stores the error code which occured during transmission using the serial communication.	S (Error)	New	Q00/Q01 Q00UJ Q00U Q01U Q02U 4)
SD111	Data receiving result	Stores the data receiving result when the serial communication is used.	Stores the error code which occured when data was received using the serial communication.	S (Error)	New	
SD118	Amount of battery consumption		This register stores a battery consumption rate. [Value range] • 1 or 2: Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UD(E)CPU, Q04UD(E)HCPU, L02CPU • 1 to 3: Q06UD(E)HCPU, L26CPU-BT • 1 to 4: Q10UD(E)HCPU, Q20UD(E)HCPU, Q13UD(E)HCPU, Q26UD(E)HCPU • 1 to 5: Q50UDEHCPU, Q100UDEHCPU	S (Status change)	New	QnU LCPU
SD119	Battery lifeprolonging factor		This register stores a value indicating a cause that has the battery life-prolonging function enabled. While this register is other than "0", the battery life-prolonging function is enabled.  	S (Status change)	New	QnU LCPU

Tab. A-59: Special registers (1): Diagnostic information (continued)

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD130 SD131 SD132 SD133 SD134 SD135 SD136 SD137	Modules with blown fuse	The bit pattern (16 Bit) indicates the modules with a blown fuse. 0: No blown fuse 1: Blown fuse detected	• The number of output modules whose fuses have blown are input as a bit pattern in units of 16 points. If the module numbers are set by parameter, the parameter-set numbers are stored. • Blown fuses of remote station output modules will be detected also. • A set bit is not automatically cleared when the module with the blown fuse is replaced. The flag is cleared by an error reset operation. 	S (Error)	New	Q00J/ Q00/ Q01
SD150 SD151 SD152 SD153 SD154 SD155 SD156 SD157	I/O module verification error	The bit pattern (16 Bit) indicates the modules with verification errors. 0: No I/O verification error 1: I/O verification error present	• When the power is turned on, the module numbers of the I/O modules whose information differs from the registered I/O module information are set in this register (in units of 16 points). If the I/O module numbers are set by parameter, the parameter-set numbers are stored.  • Not cleared even if the module is replaced with another one. The storage value is cleared by clearing the error.	S (Error)	New	

[1] The module whose first 5 digits of serial No. is "07032" or higher.

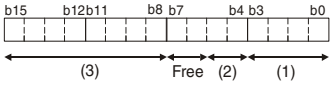
[2] The module whose first 5 digits of serial No. is "10042" or higher.

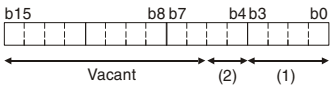
[3] This applies to Universal model QCPUs except for the Built-in Ethernet port QCPU.

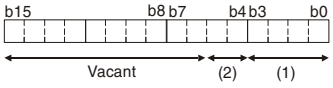
[4] The module whose first 5 digits of serial No. is "10102" or higher.

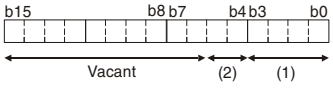
Tab. A-59: Special registers (1): Diagnostic information (continued)

A.7.2 System information

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td rowspan="4">SD200</td><td rowspan="4">State of CPU switch</td><td rowspan="4"></td><td>The CPU switch state is stored in the following format:
 

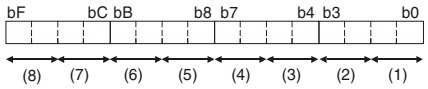
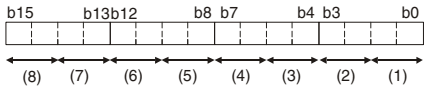
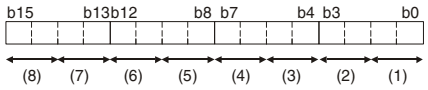
(1) CPU switch status (0): RUN (1): STOP (2): L.CLR
(2) Memory card switch: Always OFF
(3) DIP-Switch: b8 to b12 correspond to SW1 through SW5 of system setting switch 1. 0: OFF, 1: ON b13 to b15 are vacant.</td><td rowspan="2">S (Every END processing)</td><td rowspan="2">New</td><td>Qn(H)
QnPH
QnPRH</td></tr>
<tr><td>The CPU switch state is stored in the following format:
 

(1) CPU switch status (0): RUN (1): STOP
(2) Memory card switch: Always OFF</td><td>Q00J/Q00/ Q01</td></tr>
<tr><td>The CPU switch state is stored in the following format:
 

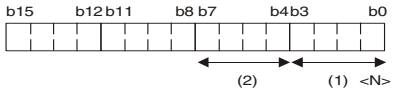
(1) CPU switch status (0): RUN (1): STOP
(2) Memory card switch: Always OFF</td><td rowspan="2">S (when RUN/STOP/ RESET switch changed)</td><td>New</td><td>QnU</td></tr>
<tr><td>The CPU switch state is stored in the following format:
 

(1) CPU switch status 0: RUN 1: STOP
(2) SD memory card switch 0: Not usable 1: Usable</td><td>New</td><td>LCPU</td></tr>
</table>

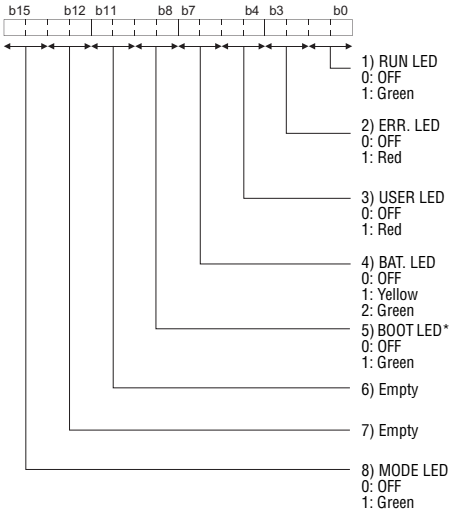
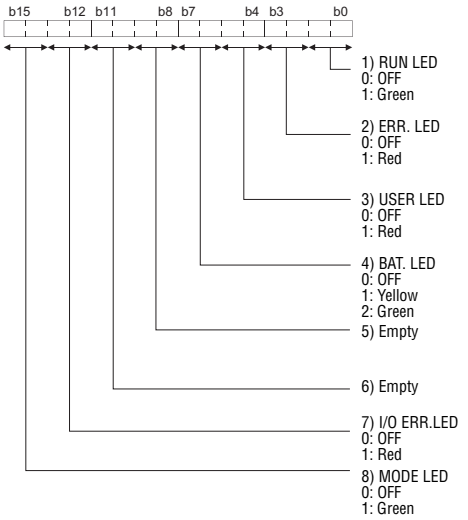
Tab. A-60: *Special registers (2): System information*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD201	LED status	State of CPU-LED	• The following bit patterns are used to store the statuses of the LEDs of the CPU. • 0: OFF, 1: ON, 2: Flicker  (1): RUN (5): BOOT (2): ERROR (6): Vacant (3): USER (7): Vacant (4): BAT.ALARM (8): MODE (0: OFF, 1: Green, 2: Orange) For the Basic model QCPU, 3) to 8) are left empty.	S (Status change)	New	Q00J/Q00/ Q01 Qn(H) QnPH QnPRH
			Information concerning which of the following states the LEDs on the CPU are stored in the following bit patterns: 0 is off, 1 is on, and 2 is flicker  (1): RUN (5): BOOT * (2): ERROR (6): Empty (3): USER (7): Empty (4): BAT.ALARM (8): MODE * For the Q00UJCPU, Q00UCPU, and Q01UCPU, 5) is left empty.	S (Status change)	New	QnU
			Information concerning which of the following states the LEDs on the CPU are stored in the following bit patterns: 0 is off, 1 is on, and 2 is flicker  (1): RUN (5): Empty (2): ERROR (6): Empty (3): USER (7): I/O ERR. (4): BAT.ALARM (8): MODE	S (Status change)	New	LCPU
SD202	LED off	Bit pattern of LED that is turned off	Stored bit patterns of LEDs turned off (Only USER and BOOT* enabled) Turned off at 1, not turned off at 0 * For the Q00UJCPU, Q00UCPU, and Q01UCPU, the BOOT LED cannot be specified.	U	New	Qn(H) QnPH QnPRH QnU
			Stored bit patterns of LEDs turned off (Only USER enabled) Turned off at 1, not turned off at 0			LCPU

Tab. A-60: *Special registers (2): System information*

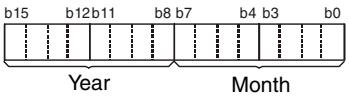
Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD203	Operating state of CPU		The CPU operating state is stored as indicated in the following figure:  (1): Operating state of CPU 0: RUN 2: STOP 3: PAUSE (2): STOP/PAUSE cause 0: Switch 1: Remote contact 2: Peripheral, computer link, or operation from some other remote source 3: Internal program instruction 4: Error Remark: The item detected first is stored. (However, for the Universal model QCPU and LCPU, the latest cause after operation status change is stored.)	S (Every END processing)	D9015 (format change)	QCPU LCPU

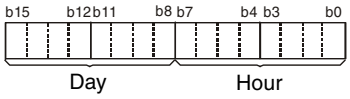
Tab. A-60: *Special registers (2): System information*

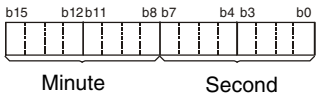
Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD204	LED display color	CPU-LED display color	The LED display color of the LED status shown in SD201 1) to 8).  * For the Q00UJCPU, Q00UCPU, and Q01UCPU, 5) is left empty.	S (Status change)	New	QnU
			The LED display color of the LED status shown in SD201 1) to 8).  			LCPU

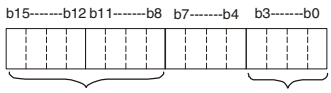
Tab. A-60: *Special registers (2): System information*

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD207</td><td rowspan="3">LED display priority ranking</td><td>Priorities 1 to 4</td><td rowspan="3">• The priority of the LED indication in the case of an error is set by a cause number. (For the Basic model QCPU, only the annunciator (cause number 7) is available.)
• For the Universal model QCPU and LCPU, specify whether to enable or disable LED indication of the error that has priority when an error occurs.
• The setting areas for priorities are as follows:
<table><tr><th></th><th>b15 to b12</th><th>b11 to b8</th><th>b7 to b4</th><th>b3 to b0</th></tr><tr><td>SD207</td><td>Priority 4</td><td>Priority 3</td><td>Priority 2</td><td>Priority 1</td></tr><tr><td>SD208</td><td>Priority 8</td><td>Priority 7</td><td>Priority 6</td><td>Priority 5</td></tr><tr><td>SD209</td><td>Priority 12</td><td>Priority 11</td><td>Priority 10</td><td>Priority 9</td></tr></table>(Priority 11 is valid when Redundant CPU is used.)
(Priority 12 is valid when LCPU is used.)

[Default value]
SD207 = 4321H (0000H for Basic model QCPU)
SD208 = 8765H (0700H for Basic model QCPU)
(0765H for Redundant CPU)
SD209 = 00A9H (0000H for Basic model QCPU)
(0B09H for Redundant CPU, CBA9H for LCPU)
• No display is made if "0" is set.
• For the Basic model QCPU, the ERR. LED lights up upon turn-on of the annunciator, if "7" is stored in any of Priorities 1 to 11.
• For the Basic model QCPU, the ERR. LED does not light up upon turn-on of the annumciator, if "7" is not stored in any of Priorities 1 to 11.
However, even if "7" has been set, information concerning CPU operation stop (including parameter settings) errors will be indicated by the LEDs without conditions.</td><td rowspan="3">U</td><td>D9038</td><td rowspan="3">Q00J/ Q00/ Q01 1)
Qn(H)
QnPH
QnPRH
QnU
LCPU</td></tr>
<tr><td>SD208</td><td>Priorities 5 to 8</td><td>D9039 (format change)</td></tr>
<tr><td>SD209</td><td>Priorities 9 to 12</td><td>New</td></tr>
<tr><td>SD210</td><td rowspan="4">Clock data</td><td>Clock data (year, month)</td><td>The year (last two digits) and month are stored as BCD code at SD210 as shown below:
 

Example: July 1993 = 9307H</td><td rowspan="3">S/U (Request)</td><td>D9025</td><td rowspan="3">QCPU
LCPU</td></tr>
<tr><td>SD211</td><td>Clock data (day, hour)</td><td>The day and hour are stored as BCD code at SD211 as shown below:
 

Example: 31., 10 a.m. = 3110H</td><td>D9026</td></tr>
<tr><td>SD212</td><td>Clock data (minute, second)</td><td>The minutes and seconds (after the hour) are stored as BCD code at SD212 as shown below:
 

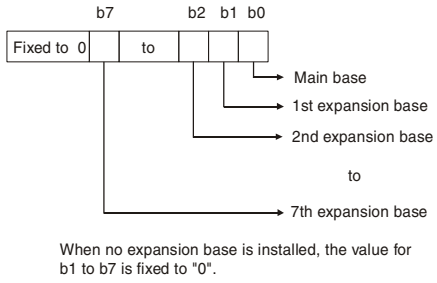
Example: 35 min., 48 sec. = 3548H</td><td>D9027</td></tr>
<tr><td>SD213</td><td>Clock data (day of the week)</td><td>The day of the week is stored as BCD code at SD213 as shown below:
 

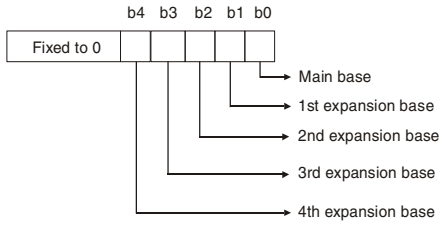
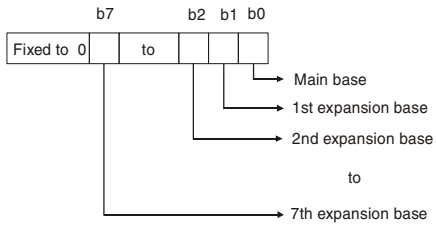
<table><tr><th colspan="2">Day of week</th></tr><tr><td>0</td><td>Sunday</td></tr><tr><td>1</td><td>Monday</td></tr><tr><td>2</td><td>Tuesday</td></tr><tr><td>3</td><td>Wednesday</td></tr><tr><td>4</td><td>Thursday</td></tr><tr><td>5</td><td>Friday</td></tr><tr><td>6</td><td>Saturday</td></tr></table></td><td>S/U (Request)</td><td>D9028</td><td>QCPU
LCPU</td></tr>
</table>

Tab. A-60: *Special registers (2): System information*

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD220</td><td rowspan="10">LED display data</td><td rowspan="10">Display indicator data</td><td colspan="3">LED display ASCII data (16 characters) stored here.
For the Basic model QCPU, an error message (up to 16 ASCII characters) is stored. (Including a message for the case the annunciator is on.)
<table><tr><th></th><th>b15 to b8</th><th>b7 to b0</th></tr><tr><td>SD220</td><td>15th character from the right</td><td>16th character from the right</td></tr><tr><td>SD221</td><td>13th character from the right</td><td>14th character from the right</td></tr><tr><td>SD222</td><td>11th character from the right</td><td>12th character from the right</td></tr><tr><td>SD223</td><td>9th character from the right</td><td>10th character from the right</td></tr><tr><td>SD224</td><td>7th character from the right</td><td>8th character from the right</td></tr><tr><td>SD225</td><td>5th character from the right</td><td>6th character from the right</td></tr><tr><td>SD226</td><td>3rd character from the right</td><td>4th character from the right</td></tr><tr><td>SD227</td><td>1st character from the right</td><td>2nd character from the right</td></tr></table>For the Basic model QCPU, Universal model QCPU or LCPU, HMI data at the time of CHK instruction execution are not stored.</td><td rowspan="10">S (Status change)</td><td rowspan="10">New</td><td rowspan="10">QCPU
LCPU</td></tr>
<tr><td>SD221</td></tr>
<tr><td>SD222</td></tr>
<tr><td>SD223</td></tr>
<tr><td>SD224</td></tr>
<tr><td>SD225</td></tr>
<tr><td>SD226</td></tr>
<tr><td>SD227</td></tr>
<tr><td>SD235</td><td>Module to which online module change is being performed</td><td>(The header I/O number of the module to which online module change is being performed)/10H</td><td colspan="3">The value of the header I/O number of which the online module change is being performed is divided by 10H and stored here.</td><td>S (During online module change)</td><td>New</td><td>QnPH
QnPRH</td></tr>
<tr><td>SD240</td><td>Base mode</td><td>0: Automatic mode
1: Detail mode</td><td colspan="3">Stores the base mode</td><td>S (Initial)</td><td>New</td><td rowspan="2">QCPU</td></tr>
<tr><td>SD241</td><td>Number of extension bases</td><td>0: Basic only
1 to 7: Number of extension bases</td><td colspan="3">Stores the number of extension bases being installed</td><td>S (Initial)</td><td>New</td></tr>
</table>

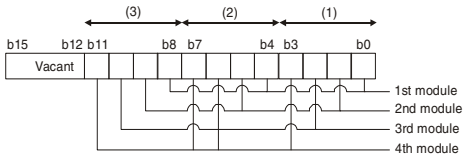
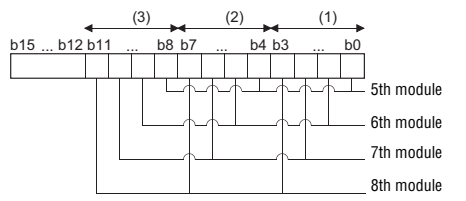
Tab. A-60: *Special registers (2): System information*

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td rowspan="3">SD242</td><td>A/Q base differentiation</td><td>0: QA**B is installed (A mode)
1: Q**B is installed (Q mode)</td><td> 

When no expansion base is installed, the value for b1 to b7 is fixed to "0".</td><td rowspan="3">S (Initial)</td><td rowspan="3">New</td><td>Qn(H)
QnPH
QnPRH</td></tr>
<tr><td rowspan="2">Installed Q base presence/ absence</td><td rowspan="2">Base type differentiation
0: Base not installed
1: Q**B is installed</td><td> 
</td><td>Q00J/Q00/ Q01</td></tr>
<tr><td> 

• For the Q00UJCPU, the bits for the third to seventh extension bases are fixed to "0".
• For the Q00UCPU, Q01UCPU, and Q02UCPU, the bits for the fifth to seventh extension bases are fixed to "0".</td><td>QnU</td></tr>
<tr><td>SD243
SD244</td><td colspan="2">Number of base slots</td><td>The number of slots being installed is stored in the respective areas for the main base and the extension bases.
<table>
<tr><th></th><th>b15 to b12</th><th>b11 to b8</th><th>b7 to b4</th><th>b3 to b0</th></tr>
<tr><td>SD243</td><td>Extension 3</td><td>Extension 2</td><td>Extension 1</td><td>Main</td></tr>
<tr><td>SD244</td><td>Extension 7</td><td>Extension 6</td><td>Extension 5</td><td>Extension 4</td></tr>
</table>
• For the Q00UJCPU, the bits for the third to seventh extension bases are fixed to "0".
• For the Q00UCPU, Q01UCPU, and Q02UCPU, the bits for the fifth to seventh extension bases are fixed to "0".</td><td>S (Initial)</td><td>New</td><td>Qn(H)
QnPH
QnPRH
QnU</td></tr>
<tr><td>SD243
SD244</td><td colspan="2">Number of base slots (Operation status)</td><td>The number of slots being installed is stored in the respective areas for the main base and the extension bases.
<table>
<tr><th></th><th>b15 to b12</th><th>b11 to b8</th><th>b7 to b4</th><th>b3 to b0</th></tr>
<tr><td>SD243</td><td>Extension 3</td><td>Extension 2</td><td>Extension 1</td><td>Main</td></tr>
<tr><td>SD244</td><td>Extension 7</td><td>Extension 6</td><td>Extension 5</td><td>Extension 4</td></tr>
</table></td><td>S (Initial)</td><td>New</td><td>Q00J/Q00/ Q01</td></tr>
<tr><td>SD245
SD246</td><td colspan="2">Number of base slots (Mounting status)</td><td>The number of slots where modules are actually mounted is stored in the area corresponding to each base unit as shown below.
<table>
<tr><th></th><th>b15 to b12</th><th>b11 to b8</th><th>b7 to b4</th><th>b3 to b0</th></tr>
<tr><td>SD245</td><td>Extension 3</td><td>Extension 2</td><td>Extension 1</td><td>Main</td></tr>
<tr><td>SD246</td><td>Fixed to 0</td><td>Fixed to 0</td><td>Fixed to 0</td><td>Extension 4</td></tr>
</table></td><td>S (Initial)</td><td>New</td><td>Qn(H)
QnPH
QnPRH
QnU</td></tr>
</table>

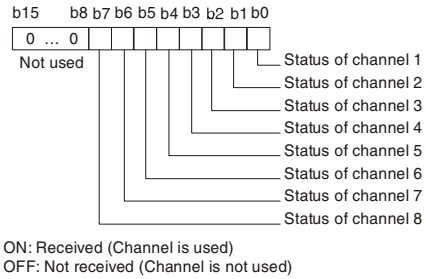
Tab. A-60: *Special registers (2): System information*

Number	Name	Meaning		Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD250	Loaded maximum I/O	Loaded maximum I/O No.		When SM250 goes from OFF to ON, the upper 2 digits of the final I/O number plus 1 of the modules loaded are stored as BIN values.	S (Request END)	New	Qn(H) QnPH QnPRH
				The first two digits of the number, which is the last I/O number of the mounted modules plus 1, are stored.	S (Initial)		Q00J/Q00/ Q01 QnU LCPU
SD254	MELSECNET/ 10/H information	Number of modules installed		Indicates the number of mounted MELSECNET/10 modules or MELSECNET/H modules.	S (Initial)	New	QCPU
SD255		Information from 1st module	I/O No.	Indicates the I/O number of mounted MELSECNET/10 modules or MELSECNET/H modules.			
SD256			Network No.	Indicates the network No. of mounted MELSECNET/10 modules or MELSECNET/H modules.			
SD257			Group Number	Indicates the group number of mounted MELSECNET/10 modules or MELSECNET/H modules.			
SD258			Station No.	Indicates the station number of mounted MELSECNET/10 modules or MELSECNET/H modules.			
SD259			Standby information	In the case of standby stations, the module number of the standby station is stored. (1 to 4)			Qn(H) QnPH QnPRH QnU [2]
SD260 – SD264		Information from 2nd module		Configuration is identical to that for the first module.			
SD265 – SD269		Information from 3rd module		Configuration is identical to that for the first module.			Qn(H) QnPH QnPRH QnU [3]
SD270 – SD274		Information from 4th module		Configuration is identical to that for the first module.			
SD280	CC-Link error	Error detection status		 (1) When Xn0 of the installed CC-Link module goes ON, the bit corresponding to the station switches ON. (2) When either Xn1 or XnF of the installed CC-Link module switch OFF, the bit corresponding to the station switches ON. (3) Switches ON when the CPU cannot communicate with the installed CC-Link module. The above modules are numbered in order of the head I/O numbers. (However, the one where parameter setting has not been made is not counted.)	S (error)	New	Qn(H) QnPH QnPRH
SD281				 (1) When Xn0 of the installed CC-Link module goes ON, the bit corresponding to the station switches ON. (2) When either Xn1 or XnF of the installed CC-Link module switch OFF, the bit corresponding to the station switches ON. (3) Switches ON when the CPU cannot communicate with the installed CC-Link module. The above modules are numbered in order of the head I/O numbers. (However, the one where parameter setting has not been made is not counted.)	S (error)	New	Qn(H) [4] QnPH [4] QnPRH [5]

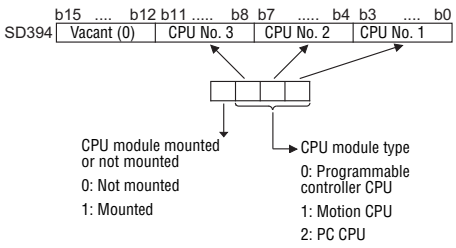
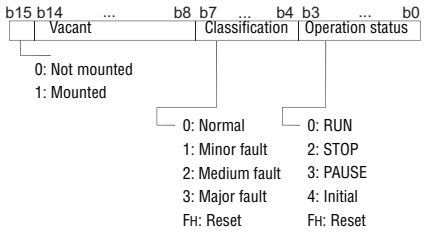
Tab. A-60: *Special registers (2): System information*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD286	Device assignment	Points assigned to M (for extension)	• The number of points assigned to M is stored with 32 bits. • The number of 32k or less points can be assigned to M.	S (Initial)	New	QnU [6] LCPU
SD287						
SD288		Points assigned to B (for extension)	• The number of points assigned to B is stored with 32 bits. • The number of 32k or less points can be assigned to B.			
SD289						
SD290	Device allocation (Same as parameter contents)	Number of points allocated for X	Stores the number of points currently set for X	S (Initial)	New	QCPU LCPU
SD291		Number of points allocated for Y	Stores the number of points currently set for Y			
SD292		Number of points allocated for M	Stores the number of points currently set for M			
SD293		Number of points allocated for L	Stores the number of points currently set for L			
SD294		Number of points allocated for B	Stores the number of points currently set for B			
SD295		Number of points allocated for F	Stores the number of points currently set for F			
SD296		Number of points allocated for SB	Stores the number of points currently set for SB			
SD297		Number of points allocated for V	Stores the number of points currently set for V			
SD298		Number of points allocated for S	Stores the number of points currently set for S			
SD299		Number of points allocated for T	Stores the number of points currently set for T			
SD300		Number of points allocated for ST	Stores the number of points currently set for ST			
SD301		Number of points allocated for C	Stores the number of points currently set for C			
SD302		Number of points allocated for D	Stores the number of points currently set for D			
SD303		Number of points allocated for W	Stores the number of points currently set for W			
SD304		Number of points allocated for SW	Stores the number of points currently set for SW			
SD305	Device assignment (Index register)	16-bit modification Number of points assigned for Z	Stores the number of points of index register (Z) used for the 16-bit modification area. (Depending on the index modification setting for ZR in the parameter setting.)	S (Initial)	New	QnU LCPU
SD306	Device assignment (Same as parameter contents)	Number of points assigned for ZR (for extension)	The number of points for ZR is stored (except the number of points of extended data register (D) and extended link register (W)). The number of points assigned to ZR is stored into this register only when 1k point or more is set for the extended data register (D) or extended link register (W).			QnU [7] LCPU
SD307						
SD308	Device assignment (assignment including the number of points set to the extended data register (D) and extended link register (W))	Number of points assigned for D (for inside + for extension)	The total points of the data register (D) in the internal device memory area and the extended data register (D) are stored as a 32-bit binary value.			
SD309						
SD310		Number of points assigned for W (for inside + for extension)	The total points of the link register (W) in the internal device memory area and the extended link register (W) are stored as a 32-bit binary value.			
SD311						
SD315	Time reserved for communication processing	Time reserved for communication processing	• Reserves the designated time for communication processing with a programming tool or other units. • The greater the value is designated, the shorter the response time for communication with other devices (programming tool, serial communication units) becomes. • Setting range: 1 to 100 ms. If the specified value is out of range, it is assumed to no setting. The scan time becomes longer by the specified time.	U	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH

Tab. A-60: *Special registers (2): System information*

Number	Name	Meaning		Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD329	Online change (inactive block) target block number	SFC block number		• While online change (inactive block) is executed (SM329 is on.), this register stores the target SFC block number. • In other than the above status, this register stores FFFFH.	S (Status change)	New	QnU [8]
SD340	Ethernet information	Number of modules installed		Indicates the number of modules installed on Ethernet.	S (Initial)	New	QCPU
SD341		Information from 1st module	I/O number	Ethernet I/O number of the first module installed.			
SD342			Network number	Ethernet network number of the first module installed.			
SD343			Group number	Ethernet group number of the first module installed.			
SD344			Station number	Ethernet station number of the first module installed.			
SD345 and SD346			Vacant	Vacant (the Ethernet IP adress of the first module is stored in buffer memory.			Qn(H) QnPH QnPRH QnU [2]
SD347			Vacant	Vacant (the Ethernet error code of the first module is read with the ERRRD instruction.			
SD348 to SD354		Information from 2nd module		Configuration is identical to that for the first module.			
SD355 to SD361		Information from 3rd module		Configuration is identical to that for the first module.			Qn(H) QnPH QnPRH QnU [3]
SD362 to SD368		Information from 4th module		Configuration is identical to that for the first module.			
SD380	Ethernet instruction reception status	Instruction reception status of the 1st module			S (Initial)	New	QnPRH
SD381		Instruction reception status of the 2nd module		Configuration is identical to that for the first module.			
SD382		Instruction reception status of the 3rd module		Configuration is identical to that for the first module.			
SD383		Instruction reception status of the 4th module		Configuration is identical to that for the first module.			

Tab. A-60: *Special registers (2): System information*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD393	Multiple CPU system information	Number of multiple CPUs	The number of CPU modules that comprise the multiple CPU system is stored. (1 to 3, Empty also included)	S (Initial)	New	Q00/Q01 [1] QnU
SD394		CPU mounting information	This register stores information on the CPU module types of CPU No.1 to No.3 and whether or not the CPU modules are mounted. 			Q00/Q01 [1]
SD395		Multiple CPU number	Stores the number of the CPU when operated in a multi-CPU system. 1: CPU No. 1; 2: CPU No. 2; 3: CPU No. 3; 4: CPU No. 4	S (Initial)	New	Q00/Q01 [1] QnH QnPH QnU
SD396		CPU No. 1 operation status	The operation information of each CPU No. is stored. (The information on the number of multiple CPUs indicated in SD393 is stored.) 	S (END processing error)	New	Q00/Q01 [1] QnU
SD397		CPU No. 2 operation status			New	Q00/Q01 [1] QnU [7]
SD398		CPU No. 3 operation status			New	QnU [3]
SD399		CPU No. 4 operation status			New	

Tab. A-60: *Special registers (2): System information*

[1] Function version is B or later.
[2] The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.
[3] The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.
[4] The module whose first 5 digits of serial No. is "08032" or higher.
[5] The module whose first 5 digits of serial No. is "09012" or higher.
[6] The module whose first 5 digits of serial No. is "10042" or higher.
[7] The Universal model QCPU except the Q00UJCPU.
[8] This applies when the first five digits of the serial number is "12052" or higher.

A.7.3 System clocks/counters

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD412	1 second counter	Number of counts in 1-second units	Following programmable controller CPU RUN, 1 is added each second. Count repeats from 0 to 32767 to -32768 to 0.	S (Status change)	D9022	QCPU LCPU
SD414	n = 1 second steps	2n second clock units	Stores value n of 2n second clock (Default is 30). Setting can be made between 1 and 32767.	U	New	
SD415	n = 1 ms steps	2n ms clock units	Stores value n of 2n ms clock (Default is 30). Setting can be made between 1 and 32767.	U	New	Qn(H) QnPH QnPRH QnU LCPU
SD420	Scan counter	Number of counts in each scan	Incremented by 1 for each scan execution after the PC CPU is set to RUN. (Not incremented for each scan of an initial execution type program.) Count repeats from 0 to 32767 to -32768 to 0.	S (Every END processing)	New	
			Incremented by 1 for each scan execution after the PC CPU is set to RUN. Count repeats from 0 to 32767 to -32768 to 0.			Q00J/Q00/Q01
SD430	Low speed scan counter	Number of counts in each scan	Incremented by 1 for each scan execution after the PC CPU is set to RUN. Count repeats from 0 to 32767 to -32768 to 0. Used only for low speed execution type programs.	S (Every END processing)	New	Qn(H) QnPH

Tab. A-61: *Special registers (3): System clocks/counters*

A.7.4 Scan information

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD500	Execution program No.		Program number of program currently being executed is stored as BIN value.	S (Status change)	New	Qn(H) QnPH QnPRH QnU LCPU
SD510	Low speed program No.		Program number of low speed program currently being executed is stored as BIN value. Enabled only when SM510 is ON.	S (Every END processing)	New	QnH QnPH
SD520	Current scan time	Current scan time (ms value)	Stores current scan time (in 1 ms units) Range from 0 to 65535	S (Every END processing)	D9018 (format change)	QCPU LCPU
SD521		Current scan time (µs value)	Stores current scan time (in 100 µs units, in increments of 1µs for the Universal model QCPU and LCPU) Range from 0 to 900 (0 to 999 for the Universal model QCPU and LCPU). (Example) A current scan of 23.6 ms would be stored as follows: D520 = 23 D521 = 600		New	
SD522	Initial scan time	Initial scan time (ms value)	Stores scan time for first scan (in 1 ms units). Range from 0 to 65535	S (First END processing)	New	Qn(H) QnPH QnPRH QnU LCPU
SD523		Initial scan time (µs value)	Stores scan time for first scan (in 100 µs units, in increments of 1µs for the Universal model QCPU and LCPU). Range from 0 to 900 (0 to 999 for the Universal model QCPU and LCPU).			
SD524	Minimum scan time	Minimum scan time (ms value)	Stores minimum value of scan time (in 1 ms units). Range from 0 to 65535	S (Every END processing)	New	Q00J/ Q00/ Q01
SD525		Minimum scan time (µs value)	Stores minimum value of scan time (in 100 µs units). Range from 0 to 900			
SD526	Maximum scan time	Maximum scan time (ms value)	Stores meximum value of scan time, excepting the first scan. (in 1 ms units). Range from 0 to 65535	S (Every END processing)		
SD527		Maximum scan time (µs value)	Stores maximum value of scan time, excepting the first scan. (in 100 µs units). Range from 0 to 900			
SD524	Minimum scan time	Minimum scan time (ms value)	Stores minimum value of scan time except that of an initial execution type program (in 1 ms units). Range from 0 to 65535	S (Every END processing)	D9017 (format change)	Qn(H) QnPH QnPRH QnU LCPU
SD525		Minimum scan time (µs value)	Stores minimum value of scan time except that of an initial execution type program (in 100 µs units; in increments of 1µs for the Universal model QCPU and LCPU). Range from 0 to 900 (0 to 999 for the Universal model QCPU and LCPU)		New	
SD526	Maximum scan time	Maximum scan time (ms value)	Stores maximum value of scan time, except that of an initial execution type program (in 1 ms units). Range from 0 to 65535	S (Every END processing)	D9019 (format change)	
SD527		Maximum scan time (µs value)	Stores maximum value of scan time, except that of an initial execution type program (in 100 µs units, in increments of 1µs for the Universal model QCPU and LCPU).). Range from 0 to 900 (0 to 999 for the Universal model QCPU and LCPU)		New	

Tab. A-62: *Special registers (4): Scan information*

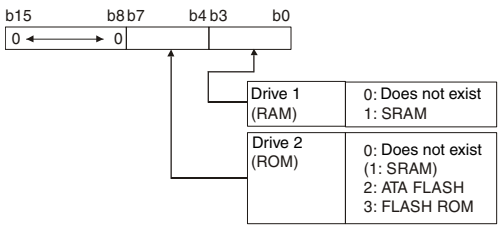
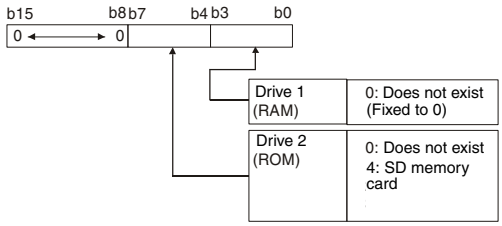
Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD528	For low speed execution type programs current scan time	Current scan time (ms value)	Stores current scan time for low speed execution type program (in 1 ms units). Range from 0 to 65535	S (Every END processing)	New	Qn(H) QnPH
SD529		Current scan time (µs value)	Stores current scan time for low speed execution type program (in 100 µs units). Range from 0 to 900			
SD532	Minimum scan time for low speed execution type programs	Minimum scan time (ms value)	Stores minimum value of scan time for low speed execution type program (in 1 ms units). Range from 0 to 65535	S (Every END processing)	New	
SD533		Minimum scan time (µs value)	Stores minimum value of scan time for low speed execution type program (in 100 µs units). Range from 0to 900			
SD534	Maximum scan time for low speed execution type programs	Maximum scan time (ms value)	Stores the maximum scan time for all except low speed execution type program's first scan (in 1 ms units). Range from 0 to 65535	S (Every END processing)	New	
SD535		Maximum scan time (µs value)	Stores the maximum scan time for all except low speed execution type program's first scan (in 100 µs units). Range from 0 to 900			
SD540	END processing time	END processing time (ms value)	Stores time from completion of scan program to start of next scan (in 1 ms units). Range from 0 to 65535	S (Every END processing)	New	Q00J/ Q00/ Q01
SD541		END processing time (µs value)	Stores time from completion of scan program to start of next scan (in 100 µs units). Range from 0 to 900			
SD540	END processing time	END processing time (ms value)	Stores time from completion of scan program to start of next scan (in 1 ms units). Range from 0 to 65535	S (Every END processing)	New	Qn(H) QnPH QnPRH QnU LCPU
SD541		END processing time (µs value)	Stores time from completion of scan program to start of next scan (in 100 µs units, in increments of 1µs for the Universal model QCPU and LCPU). Range from 0 to 900 (0 to 999 for the Universal model QCPU and LCPU)			
SD542	Constant scan wait time	Constant scan wait time (ms value)	Stores wait time when constant scan time has been set (in 1 ms units). Range from 0 to 65535	S (First END processing)	New	QCPU LCPU
SD543		Constant scan wait time (µs value)	Stores wait time when constant scan time has been set (in 100 µs units, in increments of 1µs for the Universal model QCPU and LCPU). Range from 0 to 900 (0 to 999 for the Universal model QCPU and LCPU)			
SD544	Cumulative execution time for low speed execution type programs	Cumulative execution time for low speed execution type programs (ms value)	Stores cumulative execution time for low speed execution type programs (in 1 ms units). Range from 0 to 65535 Cleared to 0 after the end of one scan of a low-speed execution type program.	S (Every END processing)	New	Qn(H) QnPH
SD545		Cumulative execution time for low speed execution type programs (µs value)	Stores cumulative execution time for low speed execution type programs (in 100 µs units). Range from 0 to 900 Cleared to 0 after the end of one scan of a low-speed execution type program.			
SD546	Execution time for low speed execution type programs	Execution time for low speed execution type programs (ms value)	Stores low speed program execution time during 1 scan (in 1 ms units). Range from 0 to 65535 Stores each scan	S (Every END processing)	New	
SD547		Execution time for low speed execution type programs (µs value)	Stores low speed program execution time during 1 scan (in 100 µs units). Range from 0 to 900 Stores each scan			

Tab. A-62: *Special registers (4): Scan information*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD548	Scan program execution time	Scan program execution time (ms value)	Stores execution time for scan execution type program during 1 scan (in 1 ms units). Range from 0 to 65535 Stores each scan	S (Every END processing)	New	Q00J/ Q00/ Q01 QnU LCPU
SD549		Scan program execution time (μs value)	Stores execution time for scan execution type program during 1 scan (in 100 μs units, in increments of 1μs for the Universal model QCPU and LCPU). Range from 0 to 900 (0 to 999 for the Universal model QCPU and LCPU) Stores each scan			
SD548	Scan program execution time	Scan program execution time (ms value)	Stores execution time for scan execution type program during 1 scan (in 1 ms units). Range from 0 to 65535 Stores each scan	S (Every END processing)	New	Qn(H) QnPH QnPRH
SD549		Scan program execution time (μs value)	Stores execution time for scan execution type program during 1 scan (in 100 μs units). Range from 0 to 900 Stores each scan			
SD550	Service interval measurement module	Unit/module No.	Sets I/O number for module that measures service interval.	U	New	Qn(H) QnPH QnPRH
SD551	Service interval time	Module service interval (ms value)	When SM551 is ON, stores service interval for module designated by SD550 (in 1 ms units). Range from 0 to 65535	S (Request)	New	
SD552		Module service interval (μs value)	When SM551 is ON, stores service interval for module designated by SD550 (in 100 μs units). Range from 0 to 900			

Tab. A-62: *Special registers (4): Scan information*

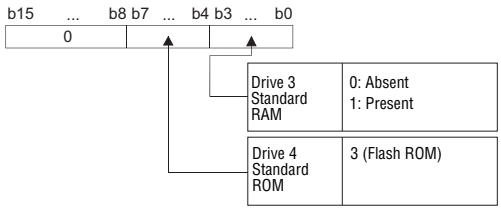
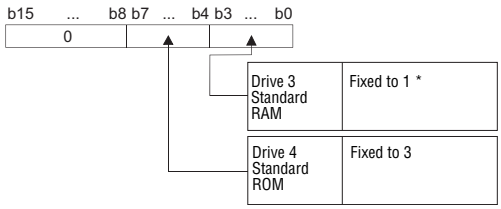
A.7.5 Memory cards

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD600	Memory card models		Indicates type of memory card installed. 	S (Initial and card removal)	New	Qn(H) QnPH QnPRH QnU
			Indicates type of memory card installed. 	S (Initial and card removal)	New	LCPU
SD602	Drive 1 (RAM) capacity		Drive 1 capacity is stored in 1 k byte units	S (Initial and card removal)	New	Qn(H) QnPH QnPRH QnU [2)]
SD603	Drive 2 (ROM) capacity		Drive 2 capacity is stored in 1 k byte units For the Q2MEM-8MBA, a value stored to this register depends on the product control number of the ATA card. For details, refer to the following manual: User's Manual (Hardware Design, Maintenance and Inspection) for the CPU module used	S (Initial and card removal)	New	

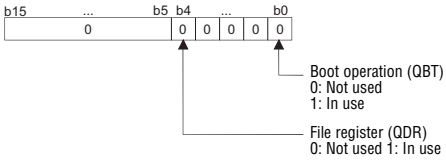
Tab. A-63: *Special registers (5): Memory cards*

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td rowspan="3">SD604</td><td rowspan="3">Memory card use conditions</td><td rowspan="3"></td><td>The use conditions for memory card are stored as bit patterns (in use when ON).
The significance of these bit patterns is indicated below:
<table><tr><td>b0: Boot operation (QBT)</td><td>b8: Not used</td></tr><tr><td>b1: Parameters (QPA)</td><td>b9: CPU fault history (QFD)</td></tr><tr><td>b2: Device comments (QCD)</td><td>b10: Not used</td></tr><tr><td>b3: Device initial value (QDI)</td><td>b11: Local device (QDL)</td></tr><tr><td>b4: File register (QDR)</td><td>b12: Not used</td></tr><tr><td>b5: Sampling trace (QTD)</td><td>b13: Not used</td></tr><tr><td>b6: Not used</td><td>b14: Not used</td></tr><tr><td>b7: Not used</td><td>b15: Not used</td></tr></table></td><td>S
(Status change)</td><td>New</td><td>Qn(H)
QnPH
QnPRH</td></tr>
<tr><td>The use conditions for memory card are stored as bit patterns (in use when ON).
The significance of these bit patterns is indicated below:
<table><tr><td>b0: Boot operation (QBT) 1)</td><td>b8: Not used</td></tr><tr><td>b1: Parameters (QPA)</td><td>b9: Not used</td></tr><tr><td>b2: Device comments (QCD)</td><td>b10: Not used</td></tr><tr><td>b3: Device initial value (QDI) 2)</td><td>b11: Local device (QDL)</td></tr><tr><td>b4: File register (QDR)</td><td>b12: Not used</td></tr><tr><td>b5: Sampling trace (QTD)</td><td>b13: Not used</td></tr><tr><td>b6: Not used</td><td>b14: Not used</td></tr><tr><td>b7: Backup data (QBP) 3)</td><td>b15: Not used</td></tr></table>1 This bit turns on at boot start and turns off at the completion.
2 This bit turns on when the writing of initial device values is started and turns off at the completion.
3 This bit can be used when the first five digits of the serial No. is "10102" or higher.</td><td>S
(Status change)</td><td>New</td><td>QnU 2)</td></tr>
<tr><td>The use conditions for memory card are stored as bit patterns (in use when ON).
The significance of these bit patterns is indicated below:
<table><tr><td>b0: Boot operation (QBT) 1)</td><td>b8: Not used</td></tr><tr><td>b1: Parameters (QPA)</td><td>b9: Not used</td></tr><tr><td>b2: Device comments (QCD)</td><td>b10: Not used</td></tr><tr><td>b3: Device initial value (QDI) 2)</td><td>b11: Local device (QDL) 3)</td></tr><tr><td>b4: File register (QDR) 3)</td><td>b12: Not used</td></tr><tr><td>b5: Sampling trace (QTD)</td><td>b13: Data logging setting (QLG) 4)</td></tr><tr><td>b6: Not used</td><td>b14: Not used</td></tr><tr><td>b7: Backup data (QBP)</td><td>b15: Not used</td></tr></table>1 This bit turns on at boot start and turns off at the completion.
2 This bit turns on when the writing of initial device values is started and turns off at the completion.
3 Fixed at "0".
4 This bit turns on when data logging setting is registered and turns off at the completion or stop of data logging.</td><td>S
(Status change)</td><td>New</td><td>LCPU</td></tr>
<tr><td>SD606</td><td rowspan="2">Drive 2 (Memory card ROM) capacity</td><td>Drive 2 storage capacity (lower bits)</td><td rowspan="2">These registers store the drive 2 storage capacity (unit: 1M byte).
(Free space value after formatting is stored.)</td><td rowspan="2">S
(Initial and card removal)</td><td rowspan="2">New</td><td rowspan="2">LCPU</td></tr>
<tr><td>SD607</td><td>Drive 2 storage capacity (upper bits)</td></tr>
</table>

Tab. A-63: *Special registers (5): Memory cards*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD616	Free space in drive 2 (Memory card ROM)	Free space in drive 2 (lower bits)	These registers store free space value in the drive 2 (unit: 1M byte).	S (Status change)	New	LCPU
SD617		Free space in drive 2 (upper bits)				
SD620	Drive 3/4 types		Indicates usage status of drives 3 and 4: 	S (Initial)	New	Q00J/ Q00/ Q01
			Indicates usage status of drives 3 and 4:  * For the Q00UJCPU, the drive 3 (Standard RAM) type is fixed at "0".	S (Initial)	New	Qn(H) QnPH QnPRH QnU LCPU
SD622	Drive 3 (RAM) capacity		Drive 3 capacity is stored in 1k byte units	S (Initial)	New	Q00J/ Q00/ Q01
			Drive 3 capacity is stored in 1k byte units. (Free space value after formatting is stored.)			Qn(H) QnPH QnPRH QnU LCPU
SD623	Drive 4 (ROM) capacity		Drive 4 capacity is stored in 1k byte units	S (Initial)	New	Q00J/ Q00/ Q01
			Drive 4 capacity is stored in 1k byte units. (Free space value after formatting is stored.)			Qn(H) QnPH QnPRH QnU LCPU

Tab. A-63: *Special registers (5): Memory cards*

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td rowspan="4">SD624</td><td rowspan="4">Drive 3 and 4 use conditions</td><td rowspan="4"></td><td>The use conditions for drives 3 and 4 are stored as bit patterns
 
</td><td rowspan="4">S
(Status change)</td><td rowspan="4">New</td><td>Q00J/
Q00/
Q01</td></tr>
<tr><td>The use conditions for drives 3 and 4 are stored as bit patterns
(In use when ON)
The significance of these bit patterns is indicated below:
<table><tr><td>b0: Boot operation (QBT)</td><td>b8: Not used</td></tr><tr><td>b1: Parameters (QPA)</td><td>b9: CPU fault history (QFD)</td></tr><tr><td>b2: Device comments (QCD)</td><td>b10: Not used</td></tr><tr><td>b3: Device initial value (QDI)</td><td>b11: Local device (QDL)</td></tr><tr><td>b4: File register (QDR)</td><td>b12: Not used</td></tr><tr><td>b5: Sampling trace (QTD)</td><td>b13: Not used</td></tr><tr><td>b6: Not used</td><td>b14: Not used</td></tr><tr><td>b7: Not used</td><td>b15: Not used</td></tr></table></td><td>Qn(H)
QnPH
QnPRH</td></tr>
<tr><td>The use conditions for drives 3 and 4 are stored as bit patterns (in use when ON).
The significance of these bit patterns is indicated below:
<table><tr><td>b0: Not used</td><td>b8: Module error log [2]</td></tr><tr><td>b1: Parameters (QPA)</td><td>b9: Not used</td></tr><tr><td>b2: Device comments (QCD)</td><td>b10: Not used</td></tr><tr><td>b3: Device initial value (QDI) [1]</td><td>b11: Local device (QDL)</td></tr><tr><td>b4: File register (QDR)</td><td>b12: Not used</td></tr><tr><td>b5: Sampling trace (QTD)</td><td>b13: Not used</td></tr><tr><td>b6: Not used</td><td>b14: Not used</td></tr><tr><td>b7: Not used</td><td>b15: Not used</td></tr></table>[1] This bit turns on when the writing of initial device values is started and turns off at the completion.
[2] This bit can be used when the first five digits of the serial No. is "11043" or higher.</td><td>QnU</td></tr>
<tr><td>The use conditions for drives 3 and 4 are stored as bit patterns (in use when ON).
The significance of these bit patterns is indicated below:
<table><tr><td>b0: Not used</td><td>b8: Module error log</td></tr><tr><td>b1: Parameters (QPA)</td><td>b9: Not used</td></tr><tr><td>b2: Device comments (QCD)</td><td>b10: Not used</td></tr><tr><td>b3: Device initial value (QDI) [1]</td><td>b11: Local device (QDL)</td></tr><tr><td>b4: File register (QDR)</td><td>b12: Not used</td></tr><tr><td>b5: Sampling trace (QTD)</td><td>b13: Data logging setting (QLG) [2]</td></tr><tr><td>b6: Not used</td><td>b14: Not used</td></tr><tr><td>b7: Not used</td><td>b15: Not used</td></tr></table>[1] This bit turns on when the writing of initial device values is started and turns off at the completion.
[2] This bit turns on when data logging setting is registered and turns off at the completion or stop of data logging.</td><td>LCPU</td></tr>
</table>

Tab. A-63: *Special registers (5): Memory cards*

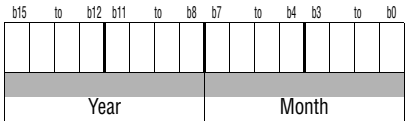
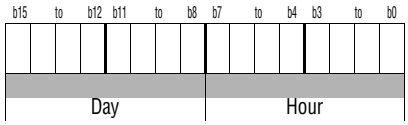
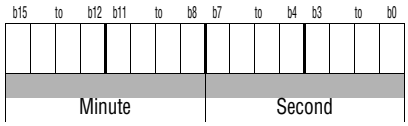
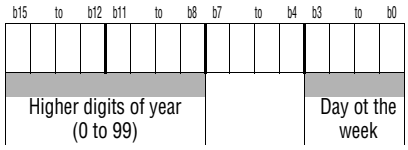
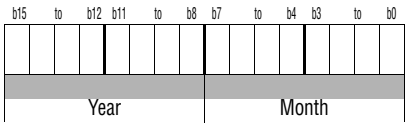
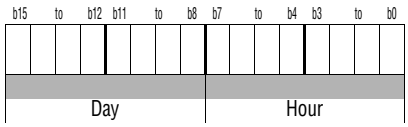
<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD640</td><td>File register drive</td><td>Drive number</td><td>Stores drive number being used by file register. *
* For the LCPU, this register is fixed at drive 3.</td><td>S
(Status change) 4)</td><td>New</td><td>Q00J/ Q00/ Q01
Qn(H)
QnPH
QnPRH
QnU 3)
LCPU</td></tr>
<tr><td>SD641
SD642</td><td rowspan="5">File register file name</td><td></td><td>Stores file register file name (MAIN.QDR) as ASCII code.
b15 b8 b7 b0
SD641: 2nd character (A) | 1st character (M)
SD642: 4th character (N) | 3rd character (I)
SD643: 6th character ( ) | 5th character ( )
SD644: 8th character ( ) | 7th character ( )
SD645: 1st char. of extension (Q) | 2EH (.)
SD646: 3rd char. of extension (R) | 2nd char. of extension (D)</td><td>S
(Initial)</td><td rowspan="5">New</td><td>Q00J/ Q00/ Q01</td></tr>
<tr><td>SD643
SD644</td><td></td><td>Stores file register file name (with extension) selected at parameters or by use of QDRSET instruction as ASCII code.
b15 b8 b7 b0
SD641: 2nd character | 1st character
SD642: 4th character | 3rd character
SD643: 6th character | 5th character
SD644: 8th character | 7th character
SD645: 1st char. of extension | 2EH (.)
SD646: 3rd char. of extension | 2nd char. of extension</td><td>S
(Status change)</td><td>Qn(H)
QnPH
QnPRH
QnU 3)</td></tr>
<tr><td>SD645
SD646</td><td></td><td>Stores file register file name (with extension) selected at parameters as ASCII code.
b15 b8 b7 b0
SD641: 2nd character | 1st character
SD642: 4th character | 3rd character
SD643: 6th character | 5th character
SD644: 8th character | 7th character
SD645: 1st char. of extension | 2EH (.)
SD646: 3rd char. of extension | 2nd char. of extension</td><td>S
(Status change)</td><td>LCPU</td></tr>
<tr><td rowspan="2">SD647</td><td rowspan="2">File register capacity</td><td rowspan="2">Stores the data capacity of the currently selected file register in 1 K word units.</td><td>S
(Status change)</td><td rowspan="2">New</td><td>Qn(H)
QnPH
QnPRH
QnU 3)
LCPU</td></tr>
<tr><td>S
(Initial)</td><td>Q00J/ Q00/ Q01</td></tr>
<tr><td>SD648</td><td>File register block number</td><td>Stores the currently selected file register block number.</td><td>S
(Status change) 4)</td><td>D9035</td><td>Q00J/ Q00/ Q01
Qn(H)
QnPH
QnPRH
QnU 3)
LCPU</td></tr>
</table>

Tab. A-63: *Special registers (5): Memory cards*

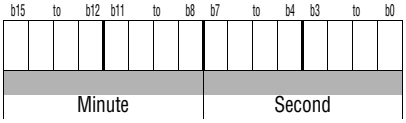
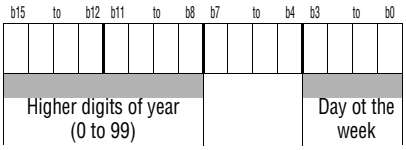
<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD650</td><td colspan="2">Comment drive</td><td>Stores the comment drive number selected at the parameters or by the QCDSET instruction.</td><td>S
(Status change)</td><td>New</td><td rowspan="7">Qn(H)
QnPH
QnPRH
QnU
LCPU</td></tr>
<tr><td>SD651
SD652
SD653
SD654
SD655
SD656</td><td colspan="2">Comment file name</td><td>Stores the comment file name (with extension) selected at the parameters or by the QCDSET instruction in ASCII code.
<table>
<tr><th></th><th>b15 b8 b7 b0</th><th></th></tr>
<tr><td>SD651</td><td>2nd character</td><td>1st character</td></tr>
<tr><td>SD652</td><td>4th character</td><td>3rd character</td></tr>
<tr><td>SD653</td><td>6th character</td><td>5th character</td></tr>
<tr><td>SD654</td><td>8th character</td><td>7th character</td></tr>
<tr><td>SD655</td><td>1st char. of extension</td><td>2EH (.)</td></tr>
<tr><td>SD656</td><td>3rd char. of extension</td><td>2nd char. of extension</td></tr>
</table></td><td>S
(Status change)</td><td>New</td></tr>
<tr><td>SD660</td><td rowspan="7">Boot operation designation file</td><td>Boot designation file drive number</td><td>Stores the drive number where the boot designation file (*.QBT) is being stored.</td><td>S
(Initial)</td><td>New</td><td rowspan="7">Qn(H)
QnPH
QnPRH
QnU [2)]
LCPU</td></tr>
<tr><td>SD661
SD662
SD663
SD664
SD665
SD666</td><td>File name of boot designation file</td><td>Stores the file name of the boot designation file (*.QBT).
<table>
<tr><th></th><th>b15 b8 b7 b0</th><th></th></tr>
<tr><td>SD661</td><td>2nd character</td><td>1st character</td></tr>
<tr><td>SD662</td><td>4th character</td><td>3rd character</td></tr>
<tr><td>SD663</td><td>6th character</td><td>5th character</td></tr>
<tr><td>SD664</td><td>8th character</td><td>7th character</td></tr>
<tr><td>SD665</td><td>1st char. of extension</td><td>2EH (.)</td></tr>
<tr><td>SD666</td><td>3rd char. of extension</td><td>2nd char. of extension</td></tr>
</table></td><td>S
(Initial)</td><td>New</td></tr>
<tr><td>SD670</td><td colspan="2">Parameter enable drive information</td><td>This register stores the number of a drive where valid parameters have been stored. *
•0: Drive 0 (program memory)
•1: Drive 1 (SRAM card)
•2: Drive 2 (Flash card/ATA card)
•4: Drive 4 (standard ROM)

* For the Q00UJCPU, Q00UCPU, and Q01UCPU, only drives 0 and 4 are parameter-valid drives.</td><td>S
(Initial)</td><td>New</td><td>QnU</td></tr>
<tr><td></td><td colspan="2"></td><td>This register stores the number of a drive where valid parameters have been stored.
•0: Drive 0 (program memory)
•2: Drive 2 (SD memory card)
•4: Drive 4 (standard ROM)</td><td></td><td></td><td>LCPU</td></tr>
<tr><td>SD671</td><td>Status of latch data backup function</td><td>Status display</td><td>This register stores the execution status of latch data backup in the following bit pattern.
<table>
<tr><th colspan="2">Status</th><th>Presence/ absence of backup data</th><th>Restore operation at turning power supply ON from OFF</th></tr>
<tr><td>0</td><td>No backup data</td><td>Absent</td><td>Restoring not execute</td></tr>
<tr><td>1</td><td>Restore ready completion</td><td rowspan="4">Present</td><td>Restoring executed when turning power supply ON from OFF the following time</td></tr>
<tr><td>2</td><td>Restore execution completion [1)]</td><td>Restoring not executed</td></tr>
<tr><td>3</td><td>Backup execution wait [2]</td><td>Restoring not executed</td></tr>
<tr><td>4</td><td>Restore repeated execution ready completion</td><td>Restoring executed when turning power supply ON from OFF</td></tr>
</table>
1) Indicates status immediately after restoration.
2) Indicates status after the CPU module is powered off and then on while the CPU module is in the "2: Restore execution completion" status.</td><td>S
(Status change)</td><td>New</td><td>QnU
LCPU</td></tr>
</table>

Tab. A-63: *Special registers (5): Memory cards*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD672	Backup information	Backup time (Year and month)	This register stores the year (last two digits) and the month when data were backed-up in 2-digit BCD. Example: July, 1993 = 9307H 	S (At write)	New	QnU LCPU
SD673		Backup time (day and hour)	This register stores the day and the hour when data were backed-up in a 2-digit BCD. Example: 31st, 10 a.m. = 3110H 	S (At write)	New	QnU LCPU
SD674		Backup time (Minute and second)	This register stores the minute and the second when data were backed-up in a 2-digit BCD. Example: 35 min., 48 sec. = 3548H 	S (At write)	New	QnU LCPU
SD675		Backup time (Year and day of week)	This register stores the year (first two digits) and the day of the week when data were backed-up in 2-digit BCD. Example: 1993, Friday = 1905H  Day of the week: 0 Sunday; 1 Monday; 2 Tuesday; 3 Wednesday; 4 Thursday; 5 Friday; 6 Saturday	S (At write)	New	QnU LCPU
SD672	Backup restoration information	Restore time (Year and month)	This register stores the year (last two digits) and the month when data were restored in 2-digit BCD. Example: July, 1993 = 9307H 	S (Initial)	New	QnU LCPU
SD673		Restore time (day and hour)	This register stores the day and the hour when data were restored in a 2-digit BCD. Example: 31st, 10 a.m. = 3110H 	S (Initial)	New	QnU LCPU

Tab. A-63: *Special registers (5): Memory cards*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD674	Backup restoration information	Restore time (Minute and second)	This register stores the minute and the second when data were restored in a 2-digit BCD. Example: 35 min., 48 sec. = 3548H 	S (Initial)	New	QnU LCPU
SD675	Backup information	Restore time (Year and day of week)	This register stores the year (first two digits) and the day of the week when data were restored in 2-digit BCD. Example: 1993, Friday = 1905H  Day of the week: 0 Sunday; 1 Monday; 2 Tuesday; 3 Wednesday; 4 Thursday; 5 Friday; 6 Saturday	S (Initial)	New	QnU LCPU
SD681	Program memory write (transfer) status	Write (transfer) status display (percentage)	This register stores the progress of writing (transfer) to the program memory (flash ROM) in percentage (0 to 100%). (When a write (transfer) command is given, "0" is stored in this register.)	S (At write)	New	QnU LCPU
SD682 SD683	Program memory write count index	Write count index up to present	This register stores the index value of write count of the program memory (flash ROM)* up to the present in 32-bit binary. When the index value exceeds 100 thousand times, "FLASH ROM ERROR" (error code: 1610) occurs. (The index value will be counted even after it exceeds 100 thousand.) * The write count does not equal to the index value. (Since the maximum write count of the flash ROM has been increased by the system, 1 is added about every two writing operations.)	S (At write)	New	QnU LCPU
SD686	Standard ROM write (transfer) status	Write (transfer) status display (percentage)	This register stores the progress of writing (transfer) to the standard ROM (flash ROM) in percentage (0 to 100%). When a write (transfer) command is given, "0" is stored in this register.	S (At write)	New	QnU LCPU
SD687 SD688	Standard ROM write count index	Write count index up to present	This register stores the index value of write count of the standard ROM (flash ROM)* up to the present in 32-bit binary. When the index value exceeds 100 thousand times, "FLASH ROM ERROR" (error code: 1610) occurs. (The index value will be counted even after it exceeds 100 thousand.) * The write count does not equal to the index value. (Since the maximum write count of the flash ROM has been increased by the system, 1 is added to the index value when the total write data size after the previous count-up reaches about 1M byte.)	S (At write)	New	QnU LCPU
SD689	Backup error factor		This register stores the cause of an error that occurred during backup. • 0H: No error • 100H: Memory card not inserted • 200H: Backup data size exceeded • 300H: Memory card write-protect setting • 400H: Memory card write error • 500H: Backup data read error (program memory) • 503H: Backup data read error (standard RAM) • 504H: Backup data read error (standard ROM) • 510H: Backup data read error (system data) • 600H: Backup preparation was performed while latch data was being backed up to the standard ROM. • 601H: Backup preparation was performed during online change. • 602H: Backup preparation was performed while a FTP client connected to the CPU module in FTP connection is present.	S (Error)	New	QnU 1) LCPU

Tab. A-63: *Special registers (5): Memory cards*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD690	Backup status		Stores the current backup status. • 0: Before backup • 1: Being prepared • 2: Ready • 3: Being executed • 4: Completed • FF: Backup error	S (Status change)	New	QnU [1)] LCPU
SD691	Backup execution status	Backup execution status display (percentage)	• This register stores the progress of backup to the memory card in percentage (0 to 100%). • "0" is stored at the start of backup.	S (Status change)	New	QnU [1)] LCPU
SD692	Restoration error factor	Factor of error occurred in restoration	Stores the cause of an error that occurred in restoration. • 800H: The CPU module model name does not match. • 801H: The backup data file does not match or the reading of backup data from the memory card is not completed. • 810H: Writing backup data to the restoration drive is not completed.	S (Error)	New	QnU [1)] LCPU
SD693	Restoration status	Current restoration status	Stores the current restoration status. • 0: Before restoration • 1: Being executed • 2: Completed • FF: Restoration error (In automatic restoration, "0: Before restoration" is stored at the completion of restoration.)	S (Status change)	New	QnU [1)] LCPU
SD694	Restoration execution status	Restoration execution status display (Percentage)	• This register stores the progress of restoration to the CPU module in percentage (0 to 100%). • "0" is stored at the start of restoration.	S (Status change)	New	QnU [1)] LCPU
SD695	Specification of writing to standard ROM instruction count		• This register stores the maximum number of executions of the writing to standard ROM instruction (SP.DEVST) per day. • When the number of executions of the writing to standard ROM instruction exceeds the number of times set by SD695, "OPERATION ERROR" (error code: 4113) occurs. • The setting range of this register is 1 to 32767. If "0" or a value outside the range has been set, "OPERATION ERROR" (error code: 4113) occurs at execution of the writing to standard ROM instruction.	U	New	QnU LCPU
SD696	Available memory in memory card		This register stores a free space value in a memory card in 32-bit binary.	S (Backup in operation)	New	QnU [1)]
SD697						
SD696	Free memory card space at backup	Free memory card space at backup (lower bits)	This register stores a free space value in a SD memory card if the free space is insufficient for storing the backup data and resulting in a backup error. (unit: byte) This register is cleared to "0" when backup is completed.	S (Backup in operation)	New	QnU [1)]
SD697		Free memory card space at backup (upper bits)				LCPU
SD698	Backup data capacity	Backup data size (lower bits)	This register stores backup data size in 32-bit binary.	S (Backup in operation)	New	QnU [1)] LCPU
SD699		Backup data size (upper bits)				

Tab. A-63: *Special registers (5): Memory cards*

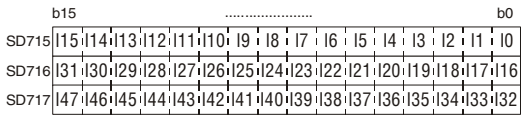
[1] The module whose first 5 digits of serial No. is "10102" or higher. (Except the Q00UJCPU, Q00UCPU, and Q01UCPU)

[2] The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

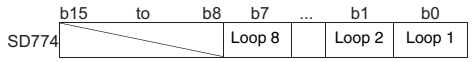
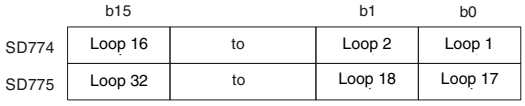
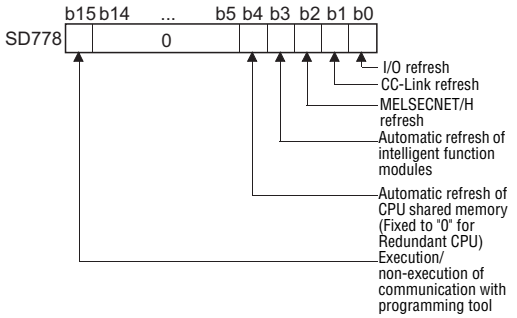
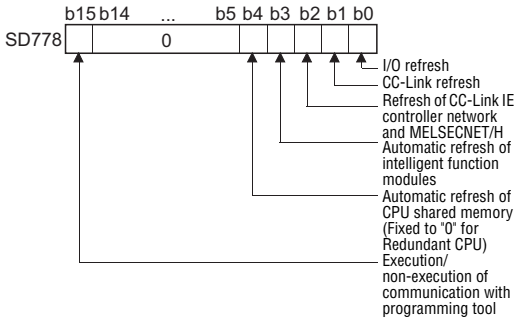
[3] The Universal model QCPU except the Q00UJCPU.

[4] On the Basic model QCPU, data is set at STOP to RUN or RSET instruction execution after parameter execution.

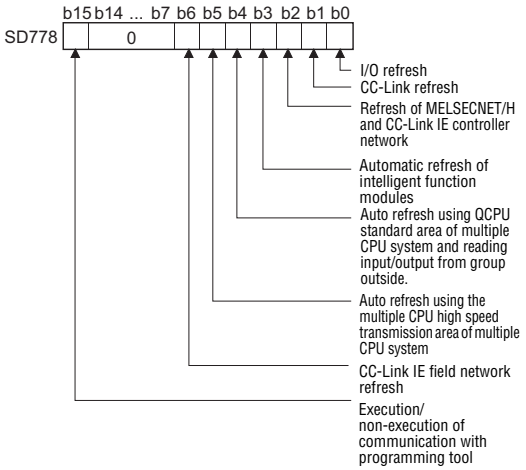
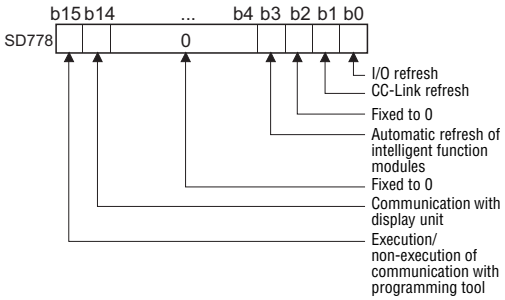
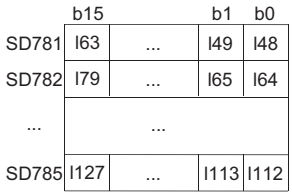
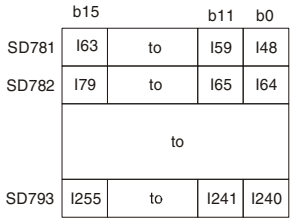
A.7.6 Instruction related registers

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td>SD705
SD706</td><td colspan="2">Mask pattern</td><td>During block operations, turning SM705 ON makes it possible to use the mask pattern being stored at SD705 (or at SD705 and SD706 if double words are being used) to operate on all data in the block with the masked values.</td><td>U</td><td>New</td><td>Q00J/ Q00/ Q01
Qn(H)
QnPH
QnPRH</td></tr>
<tr><td>SD715
SD716
SD717</td><td>IMASK instruction mask pattern</td><td>Mask pattern</td><td>Patterns masked by use of the IMASK instruction are stored in the following manner:
 
</td><td>S
(During execution)</td><td>New</td><td>QCPU
LCPU</td></tr>
<tr><td>SD718
SD719</td><td colspan="2">Accumulator</td><td>For use as replacement for accumulators used in A-series programs.</td><td>S/U</td><td>New</td><td></td></tr>
<tr><td>SD720</td><td colspan="2">Program No. destination for PLOAD instruction</td><td>Stores the program number of the program to be loaded by the PLOAD instruction when designated. The destination range is from 1 to 124.</td><td>U</td><td>New</td><td>Qn(H)
QnPH</td></tr>
<tr><td>SD738
SD739
SD740
SD741
SD742
SD743
SD744
SD745
SD746
SD747
SD748
SD749
SD750
SD751
SD752
SD753
SD754
SD755
SD756
SD757
SD758
SD759
SD760
SD761
SD762
SD763
SD764
SD765
SD766
SD767
SD768
SD769</td><td colspan="2">Message storage</td><td>Stores the message designated by the MSG instruction.
b15 ← → b8 b7 ← → b0
SD738 2nd character 1st character
SD739 4th character 3rd character
SD740 6th character 5th character
SD741 8th character 7th character
SD742 10th character 9th character
SD743 12th character 11th character
SD744 14th character 13th character
SD745 16th character 15th character
SD746 18th character 17th character
SD747 20th character 19th character
SD748 22nd character 21th character
SD749 24th character 23th character
SD750 26th character 25th character
SD751 28th character 27th character
SD752 30th character 29th character
SD753 32th character 31st character
SD754 34th character 33rd character
SD755 36th character 35th character
SD756 38th character 37th character
SD757 40nd character 39th character
SD758 42nd character 41st character
SD759 44th character 43rd character
SD760 46th character 45th character
SD761 48th character 47th character
SD762 50th character 49th character
SD763 52nd character 51st character
SD764 54th character 53rd character
SD765 56th character 55th character
SD766 58th character 57th character
SD767 60th character 59th character
SD768 62nd character 61st character
SD769 64th character 63rd character</td><td>S
(During execution)</td><td>New</td><td>QnH</td></tr>
</table>

Tab. A-64: *Special registers (6): Instruction related registers*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD774	PID limit setting (for complete derivative)	0: Limit set 1: Limit not set	Designate the limit for each PID loop as follows: 	U	New	Q00J/Q00/Q01 [1]
SD774 and SD775			Designate the limit for each PID loop as follows: 			Qn(H) QnPRH QnU LCPU
SD778	Refresh processing selection when the COM/CCOM instruction is executed	b0 to b14: (Default: 0) 0: Do not refresh 1: Refresh b15 bit 0: Communication with CPU module is executed 1: Communication with CPU module is non-executed	• Selects whether or not the data is refreshed when the COM instruction is executed. • Designation of SD778 is made valid when SM775 turns ON.  • Refresh between multiple CPUs by the COM instruction is performed under the following conditions. Data reception from another CPU: When b4 of SD778 is "1" Data transmission from host CPU: When b15 of SD778 is "0"	U	New	Q00J/Q00/Q01 [1] Qn(H) [2]
			• Selects whether or not the data is refreshed when the COM instruction is executed. • Designation of SD778 is made valid when SM775 turns ON.  • Refresh between multiple CPUs by the COM instruction is performed under the following conditions. Data reception from another CPU: When b4 of SD778 is "1" Data transmission from host CPU: When b15 of SD778 is "0" • When b2 of SD778 is 1, both the CC-Link IE controller network and MELSECNET/H perform a refresh. Therefore, when refresh point is large, processing time for the COM instruction is extended.			Qn(H) [4] QnPH [3] QnPRH

Tab. A-64: *Special registers (6): Instruction related registers*

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Description</th><th>Set by (if set)</th><th>ACPU register D9□□□</th><th>Valid for:</th></tr>
<tr><td rowspan="2">SD778</td><td rowspan="2">Refresh processing selection when the COM/CCOM instruction is executed</td><td>b0 to b14:
(Default: 0)
0: Do not refresh
1: Refresh b15 bit
0: Communication with CPU module is executed
1: Communication with CPU module is nonexecuted</td><td>• Selects whether or not the data is refreshed when the COM/CCOM instruction is executed.
• Designation of SD778 is made valid when SM775 turns ON.
 
</td><td>U</td><td>New</td><td>QnU</td></tr>
<tr><td>b0, b1, b3, b14:
(Default: 0)
0: Do not refresh
1: Refresh b15 bit
0: Communication with peripheral device is executed
1: Communication with peripheral device is non-executed</td><td>• Selects whether or not the data is refreshed when the COM/CCOM instruction is executed.
• Designation of SD778 is made valid when SM775 turns ON.
 
</td><td>U</td><td>New</td><td>LCPU</td></tr>
<tr><td>SD781 to SD785</td><td rowspan="2">Mask pattern of IMASK instruction</td><td rowspan="2">Mask pattern</td><td>Stores the mask pattern masked by the IMASK instruction as follows:
 
</td><td rowspan="2">S
(During execution)</td><td rowspan="2">New</td><td>Q00J/Q00/Q01</td></tr>
<tr><td>SD781 to SD793</td><td>Stores the mask pattern masked by the IMASK instruction as follows:
 

The Q00UJCPU, Q00UCPU, and Q01UCPU cannot use SD786 to SD793.</td><td>Qn(H)
QnPH
QnPRH
QnU
LCPU</td></tr>
</table>

Tab. A-64: *Special registers (6): Instruction related registers*

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD794	PID limit setting (for incomplete derivative)	0: With limit 1: Without limit	Designate the limit for each PID loop as follows: b15 to b8 b7 ... b1 b0 SD794 | | Loop 8 | | Loop 2 | Loop 1	U	New	Q00J/Q00/Q01 [1]
SD794 to SD795			Designate the limit for each PID loop as follows: b15 ... b1 b0 SD794 Loop 16 ... Loop 2 Loop 1 SD795 Loop 32 ... Loop 18 Loop 17			Qn(H) [4] QnPRH QnU LCPU
SD796	Maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction setting (for CPU No.1)	Maximum number of blocks range for dedicated instructions Range: 1 to 7 (Default: 2 or when setting other than 1 to 7, the register operates as 7).[6]	Specifies the maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction (target CPU = CPU No. 1). When the multiple CPU high-speed transmission dedicated instruction is executed to the CPU No.1, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM796 is turned ON, which is used as the interlock signal for consecutive execution of the multiple CPU high-speed transmission dedicated instruction.	U (At 1 scan after RUN)	New	QnU [5]
SD797	Maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction setting (for CPU No. 2)		Specifies the maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction (target CPU = CPU No. 2). When the multiple CPU high-speed transmission dedicated instruction is executed to the CPU No.2, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM797 is turned ON, which is used as the interlock signal for consecutive execution of the multiple CPU high-speed transmission dedicated instruction.		New	QnU [5]
SD798	Maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction setting (for CPU No.3)		Specifies the maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction (target CPU = CPU No. 3). When the multiple CPU high-speed transmission dedicated instruction is executed to the CPU No.3, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM798 is turned ON, which is used as the interlock signal for consecutive execution of the multiple CPU high-speed transmission dedicated instruction.		New	QnU [5]
SD799	Maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction setting (for CPU No.4)		Specifies the maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction (target CPU = CPU No. 4). When the multiple CPU high-speed transmission dedicated instruction is executed to the CPU No.4, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM799 is turned ON, which is used as the interlock signal for consecutive execution of the multiple CPU high-speed transmission dedicated instruction.		New	QnU [5]

Tab. A-64: *Special registers (6): Instruction related registers*

[1] Function version is B or later.

[2] The module whose first 5 digits of serial No. is "04012" or higher.

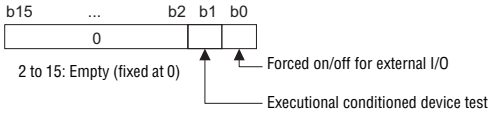
[3] The module whose first 5 digits of serial No. is "07032" or higher.

[4] The module whose first 5 digits of serial No. is "09012" or higher.

[5] The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

[6] The range is from 1 to 9 for the Q03UDCPU, Q04UDCPU, and Q06UDHCP whose first 5 digits of serial number is "10012" or lower. (Default: 2 Or when setting other than 1 to 9, the register operates as 9).

A.7.7 Debugging

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD840	Debug function usage		This register indicates the status of the debug function usage as shown below.  (0: Not used, 1: Used)	S (Status change)	New	QnU [1)] LCPU

Tab. A-65: *Special registers (7): Debugging*

[1] The module whose first 5 digits of serial No. is "10042" or higher.

A.7.8 Redundant CPU information (host system CPU information)

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD952	History of memory copy from control system to standby system	Latest status of memory copy from control system to standby system	This register stores a value indicating the completion status of the latest memory data copy from the control system to the standby system. • The value same as the SD1596 value is stored at completion or abend of the memory data copy from the control system to the standby system. • Since data have been backed up in case of power failure, this register holds the value indicating the latest memory data copy status from the control system to the standby system. • This register is cleared to 0 by latch clear.	S (Status change)	New	QnPRH

Tab. A-66: *Special registers (8): Redundant CPU information (host system CPU information)*

A.7.9 Remote password count

Number	Name	Meaning	Description	Set by (if set)	ACPU register D9□□□	Valid for:
SD979	Direct MELSOFT connection	Count of unlock processing failures	These registers store the number of mismatched password entries. Range: 0 to 0FFFEH (0FFFFH when the range is exceeded)	S (Status change)	New	QnU [1)] LCPU
SD980	Connection 1 to 16					
SD997	MELSOFT connection using UDP port					
SD998	MELSOFT connection using TCP port					
SD999	FTP communication port					

Tab. A-67: *Special registers (9): Remote password count*

[1] This applies to the Built-in Ethernet port QCPU.

A.7.10 Conversion from A series to System Q or L series

For a conversion from the MELSEC A series to the MELSEC System Q or L series the special registers D9000 through D9255 (A series) correspond to the diagnostic special registers SD1000 to SD1255 for QCPU or LCPU after the A to Q/L conversion. (Note that the Basic model QCPU and Redundant CPU do not support the A to Q/L conversion.)

These diagnostic special registers are all set by the system and cannot be changed by a user-program. Users intending to change the contents of these registers should alter their programs so that only real QCPU or LCPU diagnostic special registers are applied.

An exception are the special registers D9200 through D9255. The data in these registers can be changed by the user. Therefore, the user can change the data in the diagnostic special registers SD1200 to SD1255 after the conversion.

Refer to the manuals of the CPUs and the networks MELSECNET and MELSECNET/B for detailed information on the special registers of the A series.

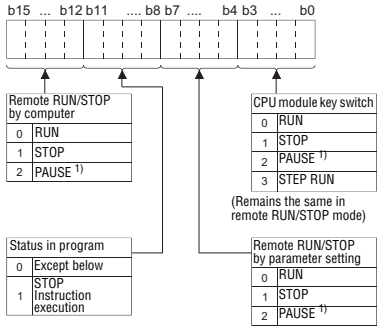
NOTE *To use the converted special register in the High Performance model QCPU, Process CPU, Universal model QCPU, or LCPU, check "Use special relay/special register from SM/SD1000" under "A-PLC Compatibility Setting".*
Project window ⇒ [Parameter] ⇒ [PLC Parameter] ⇒ [PLC System]
Note that the processing time will increase when the converted special register is used.

How to read the Special Register for Modification column.

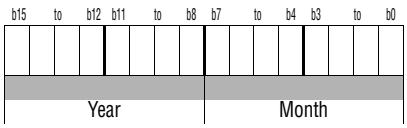
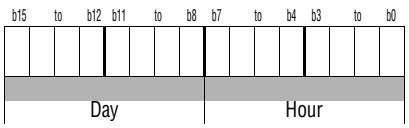
- If the special register number for QCPU or LCPU is provided, correct the program using it.
- If no special register is specified (–), the converted special register can be used.
- If the special register cannot be used in QCPU or LCPU, this is indicated as "No function for QCPU/LCPU".

<table>
<tr><th>ACPU Special Register</th><th>Special Register after Conversion</th><th>Special Register for Modification</th><th>Name</th><th>Meaning</th><th>Details</th><th>Valid for:</th></tr>
<tr><td>D9000</td><td>SD1000</td><td>—</td><td>Fuse blown</td><td>Number of module with blown fuse</td><td>• If a module with blown fuse is detected, the lowest first I/O number of the module is stored in hexadecimal. (Example: If a fuse blown is occurred in the output module with output number Y50 to Y6F, "50" is stored in hexadecimal.) To monitor the number by a programming tool, monitor in hexadecimal.(This register is cleared when contents in SD1100 to SD1107 are all reset to "0".)
• Output modules on remote I/O stations are also checked for blown fuse.</td><td>Qn(H)
QnPH
QnU 1)</td></tr>
<tr><td>D9001</td><td>SD1001</td><td>—</td><td>Fuse blown</td><td>Number of module with blown fuse</td><td>• If any fuse is blown, this register stores a number corresponding to each setting switch number or to slot number of the base unit.
<table>
<tr><th colspan="2">AJ02 I/O module</th><th colspan="2">Extension base unit</th></tr>
<tr><th>Setting switch</th><th>Stored data</th><th>Base unit Slot No.</th><th>Stored data</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>4</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>5</td></tr>
<tr><td>2</td><td>2</td><td>2</td><td>6</td></tr>
<tr><td>3</td><td>3</td><td>3</td><td>7</td></tr>
<tr><td>4</td><td>4</td><td colspan="2" rowspan="4"></td></tr>
<tr><td>5</td><td>5</td></tr>
<tr><td>6</td><td>6</td></tr>
<tr><td>7</td><td>7</td></tr>
</table>
• For the remote I/O station, the value of (module I/O No./ 10H) + 1 is stored.</td><td>Qn(H)
QnPH</td></tr>
</table>

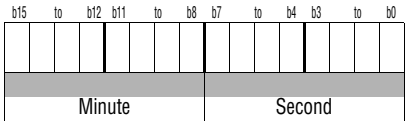
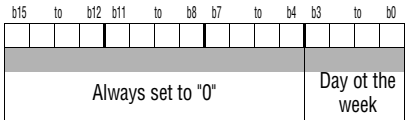
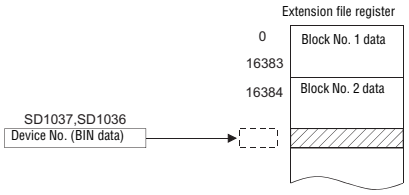
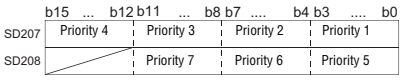
Tab. A-68: *Special registers (10): Conversion from A series to System Q or L series*

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Valid for:
D9002	SD1002	—	I/O module verification error	I/O module verification error module number	• If the status of the I/O modules changes from that obtained at power-on, the lowest first I/O number of the module is stored in hexadecimal.(Example: If a module verification error is occurred on the output module with output numbers Y50 to Y6F, "50" is stored in hexadecimal.) To monitor the number by a programming tool, monitor in hexadecimal.(This register is cleared when contents in SD1116 to SD1123 are all reset to "0".) • I/O module verification is conducted on I/O modules on remote I/O stations.	Qn(H) QnPH QnU 1) LCPU
D9005	SD1005	—	AC DOWN counter	Number of times for AC DOWN	• A value stored in this register is incremented by one whenever the input voltage falls to or below 85% (AC power) or 65 % (DC power) of the rating during operation of the CPU module. • The counter starts the routine: counts up from 0 to 32767, then counts down to -32768 and then again counts up to 0.	Qn(H) QnPH QnU 1) LCPU
D9008	SD1008	SD0	Self-diagnostic error	Self-diagnostic error number	This register stores the error code of an error detected by self-diagnostics.	Qn(H) QnPH QnU 1) LCPU
D9009	SD1009	SD62	Annunciator detection	F number at which external failure has occurred	• When any of F0 to F2047 (default device setting) is turned on by the OUT F or SET F instruction, the F number that has been detected earliest among the F numbers that have turned on is stored in BIN code. • SD1009 can be cleared by RST F or LEDR instruction. If another F number has been detected, the clearing of SD1009 causes the next number to be stored in SD1009.	
D9010	SD1010	No function for QCPU/LCPU	Error step	Step number at which operation error has occurred.	If an operation error occurred during execution of an application instruction, the number of the step having the error is stored. The contents of SD1010 are updated upon every operation error.	Qn(H) QnPH
D9011	SD1011		Error step	Step number at which operation error has occurred.	If an operation error occurred during execution of an application instruction, the number of the step having the error is stored. Because the step number is stored in SD1011 when SM1011 turns from off to on, the data in SD1011 are not updated unless SM1011 is cleared by a user program	
D9014	SD1014		I/O control mode	I/O control mode number	The I/O control mode that has been set is returned in any of the following numbers. • 0: Both input and output in direct mode • 1: Input in refresh mode, output in direct mode • 3: Both input and output in refresh mode	
D9015	SD1015	SD203	Operating state of CPU		Operation status of a CPU module is stored as shown below.  1 For the High Performance model QCPU and Process CPU, if the CPU module is running and SM1040 is off, the CPU module remains in the RUN status even though it is set to the PAUSE status.	Qn(H) QnPH QnU 1) LCPU

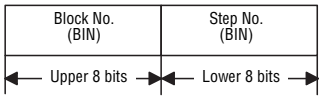
Tab. A-68: *Special registers (10): Conversion from A series to System Q or L series*

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Valid for:
D9016	SD1016	No function for QCPU/LCPU	Program number	Stores sequence program under execution as BIN value	This register stores any of the values from 0 to B, indicating which program is currently running. 0: Main program (ROM) 1: Main program (RAM) 2: Subprogram 1 (RAM) 3: Subprogram 2 (RAM) 4: Subprogram 3 (RAM) 5: Subprogram 1 (ROM) 6: Subprogram 2 (ROM) 7: Subprogram 3 (ROM) 8: Main program (EEPROM) 9: Subprogram 1 (EEPROM) A: Subprogram 2 (EEPROM) B: Subprogram 3 (EEPROM)	Qn(H) QnPH
D9017	SD1017	SD520	Scan time	Minimum scan time (10 ms units)	If a scan time value is smaller than the value in SD1017, the SD1017 value is updated in the END processing. Therefore the minimum value of scan time is stored in SD1017.	Qn(H) QnPH QnU [1)] LCPU
D9018	SD1018	SD524		Scan time (10 ms units)	This register stores a scan time in every END processing.	Qn(H) QnPH QnU [1)] LCPU
D9019	SD1019	SD526		Maximum scan time (10 ms units)	If a scan time value is greater than the value in SD1019, the SD1019 value is updated in END processing. Therefore the maximum value of scan time is stored in SD1019.	Qn(H) QnPH QnU [1)] LCPU
D9020	SD1020	No function for QCPU/LCPU	Constant scan	Constant scan time (User sets in 10 ms units)	This register stores an interval value in units of 10ms to run a program at regular intervals. • 0: No constant scan function • 1 to 200: Constant scan function available (executing at a interval of setting value × 10ms)	Qn(H) QnPH
D9021	SD1021	—	Scan time	Scan time (in 1 ms units)	This register stores scan time in every END processing.	Qn(H) QnPH QnU [1)] LCPU
D9022	SD1022	SD412	Time	Time	• The value is incremented by one every second after RUN. • The counter starts the routine: counts up from 0 to 32767, then counts down to -32768 and then again counts up to 0.	
D9025	SD1025	SD210	Clock data	Clock data (year, month)	This register stores the year (last two digits) and the month in 2-digit BCD. Example: July, 1993 = 9307H 	Qn(H) QnPH QnU [1)] LCPU
D9026	SD1026	SD211		Clock data (day, hour)	This register stores the day and the hour in a 2-digit BCD. Example: 31st, 10 a.m. = 3110H 	

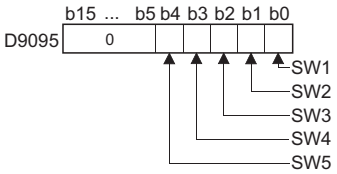
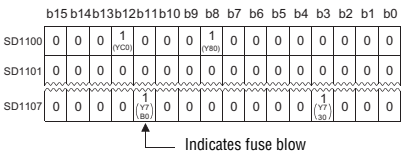
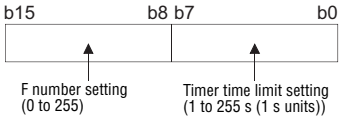
Tab. A-68: *Special registers (10): Conversion from A series to System Q or L series*

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Valid for:
D9027	SD1027	SD212	Clock data	Clock data (minute, second)	This register stores the minute and the second in 2-digit BCD. Example: 35 min., 48 sec. = 3548H 	Qn(H) QnPH QnU [1] LCPU
D9028	SD1028	SD213		Clock data (day of week)	This register stores the the day of the week in 2-digit BCD. Example: Friday = 0005H  Day of the week: 0 Sunday; 1 Monday; 2 Tuesday; 3 Wednesday; 4 Thursday; 5 Friday; 6 Saturday	
D9035	SD1035	SD648	Extension file register	Use block No.	Stores the block No. of the extension file register being used in BCD code.	Qn(H) QnPH QnU [1] LCPU
D9036	SD1036	No function for QCPU/LCPU	Extension file register for designation of device number	Device number when individual devices from extension file register are directly accessed	Designate the device number for the extension file register for direct read and write in 2 words at SD1036 and SD1037 in BIN data. Use consecutive numbers beginning with R0 of block No. 1 to designate device numbers. 	QnH QnPH
D9037	SD1037					
D9038	SD1038	SD207	LED display priority ranking	Priorities 1 to 4	This register stores priority of errors to be indicated by the ERROR LED (on or flash) by using cause numbers. 	QnH QnPH
D9039	SD1039	SD208		Priorities 5 to 7	• Configuration of the priority setting areas is as shown above. • For details, refer to the following manual: User's manual of the CPU module used Type ACPU/QCPU-A (A Mode) Programming Manual (Fundamentals)	

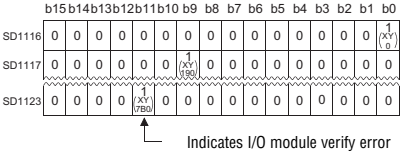
Tab. A-68: *Special registers (10): Conversion from A series to System Q or L series*

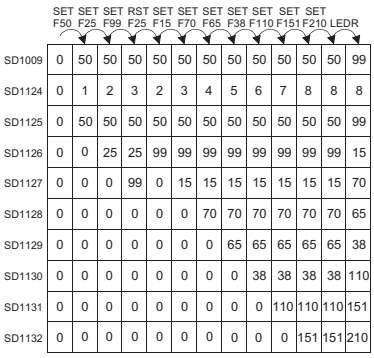
ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Valid for:
D9044	SD1044	No function for QCPU/LCPU	For sampling trace	Step or time during sampling trace	To operate the STRA or STRAR instruction of a sampling trace by turning on or off SM803 with a programming tool, use the value stored in SD1044 as the sampling trace condition. • When "Each scan" is selected: 0 • When a timing is specified: setting value (Unit: 10ms)	QnH QnPH
D9049	SD1049		Work area for SFC	Block number of extension file register	• This register stores the block No. of the extended file register used as a work area for executing the SFC program. • This register stores "0" when SM320 is off and when empty area of 16K bytes or smaller is used (16K byte or less is too small to be used as block No.1 for an extended file register).	QnH QnPH
D9050	SD1050		SFC program error number	Error code generated by SFC program	This register stores an error code of the error occurred in the SFC program. • 0 : No error • 80: SFC program parameter error • 81: SFC code error • 82: Number of steps of simultaneous execution exceeded • 83: Block start error • 84: SFC program operation error	QnH QnPH
D9051	SD1051		Error block	Block number where error occurred	This register stores the number of the block in the SFC program where an error occurred. For error 83, the number of the block where the program was started is stored.	QnH QnPH
D9052	SD1052		Error step	Step number where error occurred	• This register stores the number of the step in the SFC program where error 83 occurred. • For error 80, 81, and 82, "0" is stored. • For error 83, the block starting step number is stored.	QnH QnPH
D9053	SD1053		Error transition	Transition condition number where error occurred	This register stores the number of the transition condition in the SFC program where error code 84 occurred. For error codes 80, 81, 82, and 83, "0" is stored.	QnH QnPH
D9054	SD1054		Error sequence step	Sequence step number where error occurred	This register stores the sequence step number of transfer condition and operation output in the SFC program where error 84 occurred.	QnH QnPH
D9055	SD1055	SD812	Status latch	Status latch step	• This register stores the number of the step where a status latch was executed. • When a status latch was executed in a main sequence program, the step No. is stored. • When a status latch was executed in a SFC program, the block number and step number are stored. 	QnH QnPH
D9072	SD1072	No function for QCPU/LCPU	PLC communications check	Data check of serial communication module	The serial communication module automatically reads and writes data in a single loopback test to perform communication check.	QnH QnPH

Tab. A-68: *Special registers (10): Conversion from A series to System Q or L series*

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Valid for:
D9085	SD1085	No function for QCPU/LCPU	Register for setting time check value		Sets the time check time of the data link instructions (ZNRD, ZNWR) for the MELSECNET/10. • Setting range: 1s to 65535s (1 to 65535) • Unit: second • Default: 10 s (If 0 has been set)	QnH QnPH
D9090	SD1090		Microcomputer subroutine input data area start device number	Depends on microcomputer package.	For details, refer to the following manual: Manual for respective microcomputer package	QnH QnPH
D9091	SD1091		Detailed error code	Self-diagnosis detailed error code	This register stores description of the error cause of an instruction error.	Qn(H) QnPH QnU 1) LCPU
D9094	SD9094	SD251	Head I/O number for replacement	Head I/O number for replacement	This register stores the first two digits of the start I/O number of an I/O module, which is to be removed and mounted online (with power on). Example: Input module with I/O No. X2F0 → H2F	QnH QnPH
D9095	SD1095	SD200	DIP switch information		This register stores a status of the DIP switch of the CPU module in the following format. • 0: OFF • 1: ON 	QnH QnPH
D9100	SD1100	—	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown	The number of an output module whose fuse has blown is stored in the following bit pattern (in units of 16 points).(If the module number has been set by parameter, the parameter-set number is stored.)  • Output modules on remote I/O stations are also checked for blown fuse.(This register must be cleared by a program because the bit status remains unchanged even after clearing the error.)	Qn(H) QnPH QnU 1)
D9101	SD1101					
D9102	SD1102					
D9103	SD1103					
D9104	SD1104					
D9105	SD1105					
D9106	SD1106					
D9107	SD1107					
D9108	SD1108	—	Step transfer monitoring timer setting	Timer setting value and the F number at time out	• This register stores a value set for step transition monitoring timer and the number of an annunciator (F number) that turns on if the monitoring timer times out.  • Turning on any of relays SM1108 to SM1114 activates a monitoring timer. If the transition condition for the step is not established before the time-out time, the annunciator (F) turns on.	Qn(H) QnPH
D9109	SD1109					
D9110	SD1110					
D9111	SD1111					
D9112	SD1112					
D9113	SD1113					
D9114	SD1114					

Tab. A-68: *Special registers (10): Conversion from A series to System Q or L series*

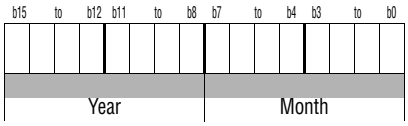
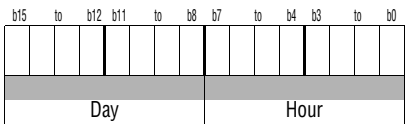
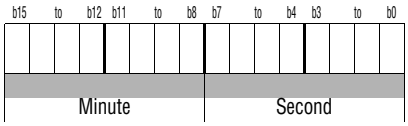
<table>
<tr><th>ACPU Special Register</th><th>Special Register after Conversion</th><th>Special Register for Modification</th><th>Name</th><th>Meaning</th><th>Details</th><th>Valid for:</th></tr>
<tr><td>D9116</td><td>SD1116</td><td rowspan="8">—</td><td rowspan="8">I/O module verification error</td><td rowspan="8">Bit pattern in units of 16 points, indicating the modules with verification errors</td><td rowspan="8">• If the status of the I/O module changes from that obtained at power-on, the module No. (unit: 16 points) is stored in the following bit pattern. (When I/O module numbers have been set by the parameter, the parameter-set numbers are stored.)
 

• I/O module verification is conducted on I/O modules on remote I/O stations. (If normal status is restored, clear is not performed. Therefore, it is required to perform clear by user program.)</td><td rowspan="8">Qn(H)
QnPH
QnU [1]
LCPU</td></tr>
<tr><td>D9117</td><td>SD1117</td></tr>
<tr><td>D9118</td><td>SD1118</td></tr>
<tr><td>D9119</td><td>SD1119</td></tr>
<tr><td>D9120</td><td>SD1120</td></tr>
<tr><td>D9121</td><td>SD1121</td></tr>
<tr><td>D9122</td><td>SD1122</td></tr>
<tr><td>D9123</td><td>SD1123</td></tr>
<tr><td>D9124</td><td>SD9124</td><td>SD63</td><td colspan="2">Annunciator detection quantity</td><td>When any of F0 to F2047 (default device setting) is turned on by the SET F instruction, a value in SD1124 is incremented by one (up to a maximum of 16).
When the RST F or LEDR instruction is executed, it is decremented by one.</td><td>Qn(H)
QnPH
QnU [1]
LCPU</td></tr>
<tr><td>D9125</td><td>SD9125</td><td>SD64</td><td rowspan="8">Annunciator detection number</td><td rowspan="8"></td><td rowspan="8">When any of F0 to F2047 (default device setting) are turned on by the SET F instruction, the annunciator numbers (F numbers) that are turned on are stored in SD1125 to SD1132 in order.
The F numbers turned off by the RST F instruction is deleted from this register, and the F numbers stored after the deleted F numbers are shifted to the previous registers. When the LEDR instruction is executed, the contents of SD1125 to SD1132 are shifted upward by 1.
When there are eight annunciator detections, the next one is not stored in SD1125 to SD1132.
 
</td><td rowspan="8">Qn(H)
QnPH
QnU [1]
LCPU</td></tr>
<tr><td>D9126</td><td>SD9126</td><td>SD65</td></tr>
<tr><td>D9127</td><td>SD9127</td><td>SD66</td></tr>
<tr><td>D9128</td><td>SD9128</td><td>SD67</td></tr>
<tr><td>D9129</td><td>SD9129</td><td>SD68</td></tr>
<tr><td>D9130</td><td>SD9130</td><td>SD69</td></tr>
<tr><td>D9131</td><td>SD9131</td><td>SD70</td></tr>
<tr><td>D9132</td><td>SD9132</td><td>SD71</td></tr>
</table>

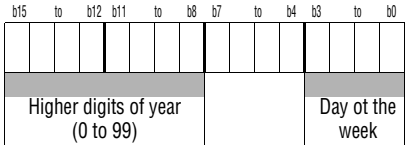
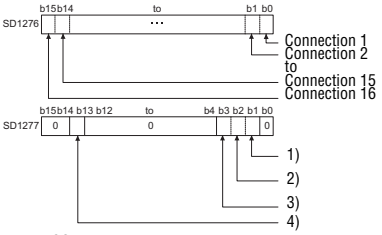
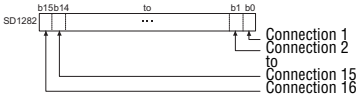
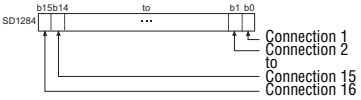
Tab. A-68: *Special registers (10): Conversion from A series to System Q or L series*

[1] The relevant modules are as follows:
- The Universal model QCPU whose serial number (first five digits) is "10102" or higher.
- Q00UJCPU, Q00UCPU, Q01UCPU

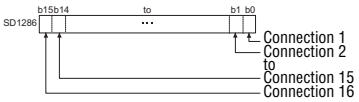
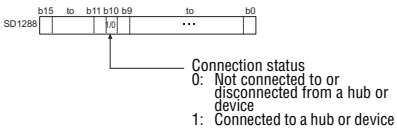
A.7.11 Built-in Ethernet port QCPU and built-in Ethernet function

Number	Name		Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1260	IP address in-use		IP address (lower digits)	These register store an IP address of the built-in Ethernet port.	S (Initial)	New	QnU 3)
SD1261			IP address (upper digits)				
SD1262			Subnet mask pattern (lower digits)	• These register store a subnet mask pattern of the built-in Ethernet port. • When a subnet mask pattern is not set, "0" is stored.			
SD1263			Subnet mask pattern (upper digits)				
SD1264			Default router IP address (lower digits)	• These register store a default router IP address of the built-in Ethernet port. • When a default router IP address is not stored, "0" is stored.			
SD1265			Default router IP address (upper digits)				
SD1270	Time setting function	Operation result	Stores operation result.	This register stores the operation result of the time setting function. 0: Not executed 1: Success 0FFFFH: Failure	S (Status change)	New	QnU 1) LCPU
SD1271		Execution time	Stores time acquired with time setting function.	This register stores the year (last two digits) and the month that the time setting function was executed in 2-digit BCD. Example: July, 1993 = 9307H 	S (Status change)	New	QnU 1) LCPU
SD1272				This register stores the day and the hour that the time setting function was executed in a 2-digit BCD. Example: 31st, 10 a.m. = 3110H 	S (Status change)	New	QnU 1) LCPU
SD1273				This register stores the minute and the second that the time setting function was executed in a 2-digit BCD. Example: 35 min., 48 sec. = 3548H 	S (Status change)	New	QnU 1) LCPU

Tab. A-69: *Special registers (11): Built-in Ethernet port QCPU and built-in Ethernet function*

Number	Name		Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1274	Time setting function	Execution time	Stores time acquired with time setting function.	This register stores the year (first two digits) and the day of the week that the time setting was executed in 2-digit BCD. Example: 1993, Friday = 1905H  Day of the week: 0 Sunday; 1 Monday; 2 Tuesday; 3 Wednesday; 4 Thursday; 5 Friday; 6 Saturday	S (Status change)	New	QnU [1] LCPU
SD1275		Required response time	Stores time required for clock time aquisition.	This register stores the time required for a clock value to be set on the CPU after being sent to the SNTP server. Range: A0 to 0FFFEH (Unit: ms) 0FFFFH when the above limit is exceeded.	S (Status change)	New	QnU [1] LCPU
SD1276 SD1277	Forced connection invalidation		Specifies forced connection invalidation.	This register is specified to forcibly invalidate a connection by a program. After being specified invalid, the connection stops communication and does not respond. (When a remote password is used and frequent unlock processing errors occur on a connection, this register is useful for temporarily disabling access.)  1) MELSOFT communication port (UDP/IP) 2) MELSOFT communication port (TCP/IP) 3) FTP communication port 4) Direct connection to MELSOFT • 0: Valid (default) • 1: Invalid	U	New	QnU [1] LCPU
SD1282	Open completion signal		Stores open completion status	This register stores the open completion status of a socket communication connection. (when the open method is socket communication). Bits for connections other than that of socket communication are always "0".  • 0: Open processing is not completed. • 1: Open processing is completed.	S (Status change)	New	QnU [2] LCPU
SD1284	Open request signal		Stores open request status	This register stores the open request status of a socket communication connection. Bits for connections other than that of socket communication are always "0".  • 0: No open request • 1: In open request			

Tab. A-69: *Special registers (11): Built-in Ethernet port QCPU and built-in Ethernet function*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1286	Reception status signal	Stores reception status	This register stores the receive status of a socket communication connection. Bits for connections other than that of socket communication are always "0".  TCP (standard receive mode) • 0: Data not received • 1: Data received TCP (fixed-length receive mode) • 0: Data not received or received data size is not the size of receive buffer. • 1: Received data size reached to the receive buffer size. UDP • 0: Data not received • 1: Data received	S (Status change)	New	QnU [2] LCPU
SD1288	Built-in Ethernet port connection status	Stores connection status of built-in Ethernet port	This register stores a connection status of the built-in Ethernet port.  	S (Status change)	New	QnU [2] LCPU
SD1292	IP address setting	IP address (lower digits)	• Specify an IP address to be stored in the IP address storage area (flash ROM). Range: 00000001H to DFFFFFFEH (0.0.0.1 to 223.255.255.254) • When writing to or clearing the IP address storage area (flash ROM) is completed, the values of the IP address stored in the IP address storage area (flash ROM) are stored.	S (Status change) / U	New	QnU [3]
SD1293		IP address (upper digits)				
SD1294		Subnet mask pattern (lower digits)	• Specify a subnet mask pattern to be stored in the IP address storage area (flash ROM). Range: C0000000H to FFFFFFFCH (192.0.0.0 to 255.255.255.252), 00000000H (blank) • When writing to or clearing the IP address storage area (flash ROM) is completed, the values of the subnet mask pattern stored in the IP address storage area (flash ROM) are stored.			
SD1295		Subnet mask pattern (upper digits)				
SD1296		Default router IP address (lower digits)	• Specify a default router IP address to be stored in the IP address storage area (flash ROM). Range: 00000001H to DFFFFFFEH (0.0.0.1 to 223.255.255.254), 00000000H (blank) • When writing to or clearing the IP address storage area (flash ROM) is completed, the values of the default router IP address stored in the IP address storage area (flash ROM) are stored.			
SD1297		Default router IP address (upper digits)				
SD1298	IP address storage area write error factor	Stores error factor when failing to write to IP address storage area	This register stores an error factor occurred when writing to the IP address storage area (flash ROM). (Links with SM1294.) • 0H: No error • 100H: The values of SD1292 to SD1297 are out of the setting range. • 200H: Write error • 300H: Writing is not available because other function is being executed. • 400H: Writing is not available because the IP address storage area is being cleared	S (Status change)	New	QnU [3]

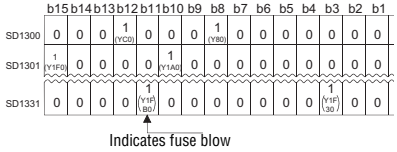
Tab. A-69: *Special registers (11): Built-in Ethernet port QCPU and built-in Ethernet function*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1299	IP address storage area clear error factor	Stores error factor when failing to clear IP address storage area	This register stores an error factor occurred when clearing the IP address storage area (flash ROM). (Links with SM1297.) • 0H: No error • 200H: Clear error • 300H: Clearing is not available because other function is being executed. • 400H: Clearing is not available because the IP address storage area is being written.	S (Status change)	New	QnU [3)]

Tab. A-69: *Special registers (11): Built-in Ethernet port QCPU and built-in Ethernet function*

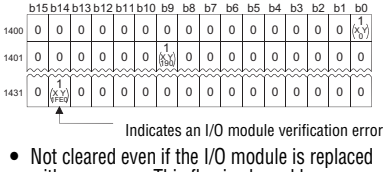
[1] This applies to the Built-in Ethernet port QCPU.
[2] The built-in Ethernet port QCPU whose serial number (first five digits) is "11012" or higher is targeted.
[3] This applies to the built-in Ethernet port QCPU whose first five digits of serial No. is "11082" or higher.

A.7.12 Fuse blown module

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1300	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown 0: No blown fuse 1: Blown fuse present	• The number of an output module whose fuse has blown is stored in the following bit pattern (in units of 16 points). (If the module numbers are set by parameter, the parameter-set numbers are stored.) • The status of the blown fuse of an output module on a remote station is also detected.  • Not cleared even if the blown fuse is replaced with a new one. The numbers are cleared by clearing the error.	S (Error)	D9100	Qn(H) QnPH QnPRH QnU
SD1301					D9101	
SD1302					D9102	
SD1303					D9103	
SD1304					D9104	
SD1305					D9105	
SD1306					D9106	
SD1307					D9107	
SD1308					New	
SD1309 to SD1330						
SD1331						

Tab. A-70: *Special registers (12): Fuse blown module*

A.7.13 I/O module verification

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1400	I/O module verify error	Bit pattern in units of 16 points, indicating the modules with verification errors 0: No I/O verification errors 1: I/O verification error present	• If the status of the I/O module changes from that obtained at power-on, the module No. is stored in the following bit pattern.(If the I/O numbers are set by parameter, the parameter-set numbers are stored.) • Also detects I/O module information.  • Not cleared even if the I/O module is replaced with a new one. This flag is cleared by error resetting operation.	S (Error)	D9116	Qn(H) QnPH QnPRH QnU LCPU
SD1401					D9117	
SD1402					D9118	
SD1403					D9119	
SD1404					D9120	
SD1405					D9121	
SD1406					D9122	
SD1407					D9123	
SD1408					New	
SD1409 to SD1430						
SD1431						

Tab. A-71: *Special registers (13): I/O module verification*

A.7.14 Process control instruction

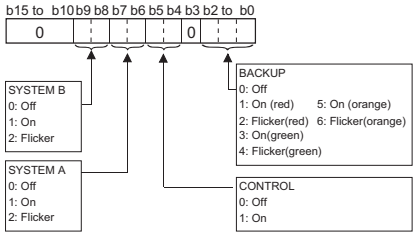
Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1500 SD1501	Basic period	Basic period time	Set the basic period (1 second units) use for the process control instruction using floating-point data. Floating points data = SD1501 SD1500	U	New	QnPH
SD1502	Process control instruction detail error code		This register indicates the details of an error occurred by executing a process control instruction.			
SD1503	Process control instruction generated error location		Shows the error process block that occurred in the process control instruction.	S (error)		
SD1506 SD1507	Dummy device		Used to specify dummy devices by a process control instruction.			QnPH QnPR
SD1508	Function availability selection for process control instruction	b0 Bumpless function availability setting for the S.PIDP instruction 0: Enabled 1: Disabled (Default: 0)	This register stores whether to enable functions for process control instructions. b15 b14 to b2 b1 b0 SD1508 0 0 0 1/0 Bumpless function availability for the S.PIDP instruction	U		

Tab. A-72: *Special registers (14): Process control instruction*

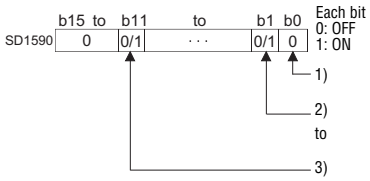
A.7.15 Redundant system (host system CPU information)

The special registers SD1585 to SD1596 store information of the host CPU module.

These special registers are valid only for the redundant system. SD1585 to SD1596 are cleared to "0" in a stand-alone system.

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1585	Redundant system LED status	4 LED states • BACKUP • CONTROL • SYSTEM A • SYSTEM B	The LED status of BACKUP, CONTROL, SYSTEM A, or SYSTEM B is stored in the following format: 	S (Status change)	New	QnPRH
SD1588	Reason(s) for system switching	Reason(s) for system switching that occurred in host station	Stores the reason(s) for system switching on the host system. The following values are stored corresponding to the methods for system switching: This register is initialized with zero (0) stored when the system is powered on from off or is reset. 0: Initial value (control system has never been switched) 1: Power off, Reset, H/W failure, WDT error 2: CPU stop error (except WDT) 3: System switching request from network module 16: System switching dedicated instruction 17: System switching request from a programming tool	S (when condition occurs)		QnPRH
SD1589	Reason(s) for system switching failure conditions	Reason(s) for system switching failure No	• If a system switching is failed, any of the following value is stored in this register. 0: System switching complete (default) 1: Tracking cable is not connected, tracking cable failure, or internal circuit failure 2: H/W failure, power-off, reset, watchdog timer error on the standby system 3: H/W failure, power-off, reset, WDT error on the control system 4: Preparing tracking communication 5: Communication timeout 6: Stop error on the standby system (except for watchdog timer error) 7: Operation differs between both systems (detected only in the back up mode) 8: During memory copy from control system to standby system 9: Performing program online change 10: Detecting a failure of network module on the standby system 11: System is being switched • Resets to "0" when host system is powered on. • Zero is stored in this register upon completion of system switching.	S (when system is switched)		QnPRH

Tab. A-73: *Special registers (15): Redundant system (host system CPU information)*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1590	Network module number, which requested system switching		• When system switching is requested from a network module in the host system, the bit corresponding to the module that received the request turns on.  1) Module 0: CPU module is invalid as it is 2-slot model 2) Module 1: Module on right side of CPU module 3) Module 11: Module at rightmost end of 12-slot base (Q312B) • The system turns off the bit after the error is removed by a user. • For the number for modules where system switching is requested from a network module in other system, refer to SD1690.	S (Error/Status change)	New	QnPRH
SD1595	Memory copy target I/O number		Before SM1595 is turned from off to on, the I/O No. of the memory copy destination (Standby system CPU module: 3D1H) is stored in this register.	U	New	QnPRH
SD1596	Memory copy status		Stores the execution result of Memory copy function. • 0: Memory copy is complete • 4241H: Standby system power supply off • 4242H: Tracking cable is disconnected or is damaged • 4247H: Memory copy is being executed • 4248H: Unsupported memory copy destination I/O number	S (Status change)	New	QnPRH

Tab. A-73: *Special registers (15): Redundant system (host system CPU information)*

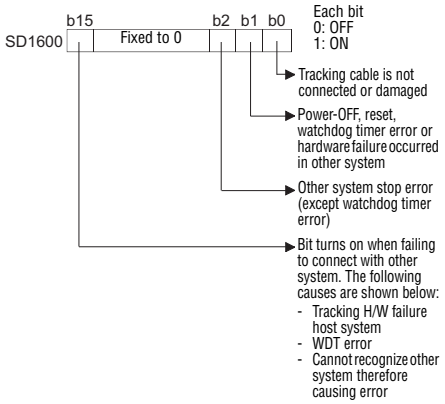
A.7.16 Redundant system (other system CPU information)

The special registers SD1600 to SD1690 store diagnostic information and system information of other system CPU.

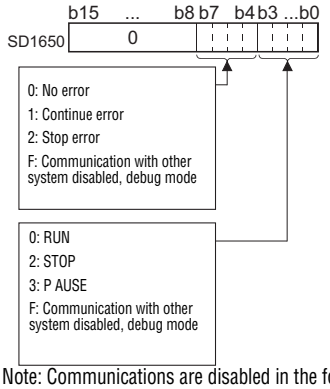
The special registers SD1600 to SD1650 are valid only when the redundant system is in backup mode and invalid in separate mode.

The special registers SD1651 to SD1690 are valid when the redundant system is in backup mode or in separate mode.

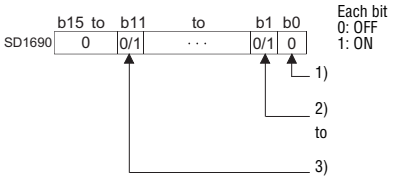
The registers from SD1600 to SD1690 are cleared to "0" in a stand-alone system.

Number	Name	Meaning	Description	Set by (if set)	Corresponding host CPU SD□□□	Valid for:
SD1600	System error information		• If an error is detected by the error check for redundant system, the corresponding bit shown below turns on. That bit turns OFF when the error is cleared after that.  • If any of b0, b1, b2 and b15 is on, the other bits are off. • In the debug mode, b0, b1, b2 and b15 are all off.	S (Every END processing)		QnPRH
SD1601	System switching results		Reason(s) for system switching is stored. • When a system is switched, the reason for system switching is stored in SD1601 of both systems. • This register is initialized with zero (0) stored when the system is powered on from off or is reset. • The following shows the values stored in this register. 0: Initial value (control system has never been switched) 1: Power-off, reset, H/W failure, or watchdog timer error [1)] 2: Stop error (except for watchdog timer error) 3: A system switching request from network module 16: Control system switching instruction 17: System switching request from a programming tool [1] When the system is switched upon the power-off or reset of the control system, "1" is not stored in SD1601 of the new standby system.	S (when system is switched)		QnPRH
SD1602	System switching dedicated instruction parameter		• This register stores the argument to the instruction when a system is switched by the SP.CONTSW instruction. (The argument for the SP.CONTSW instruction is stored in SD1602 of both systems upon system switching.) • SD1602 is only valid when "16" is stored in SD1601. • SD1602 is updated only when a system is switched by the control system switching instruction.			QnPRH

Tab. A-74: *Special registers (16): Redundant system (other system CPU information)*

Number	Name	Meaning	Description	Set by (if set)	Corresponding host CPU SD□□□	Valid for:
SD1610	Other system diagnostic error	Diagnostic error code	• This register stores an error code for the error occurred on other system. • The value in SD0 of the CPU module on other system is reflected.	S (Every END processing)	SD0	QnPRH
SD1611 SD1612 SD1613	Other system diagnostic error occurrence time	Diagnostic error occurrence time	• Stores the date and time when diagnostics error occurred corresponding to error code stored in SD1610. • Data format is the same as SD1 to SD3. • The values in SD1 to SD03 of the CPU module on other system are reflected.		SD1 to SD3	QnPRH
SD1614	Other system error information category	Error information category code	• This register stores the category code of error information and individual information of the error that ccurred on the other system. • Data format is the same as SD4. • The value in SD4 of the CPU module on other system is reflected.	S (Every END processing)	SD4	QnPRH
SD1615 to SD1625	Other system error common information	Error common information	• Stores the common information corresponding to the error code of the error that ccurred on the other system. • Data composition is the same as SD5 to SD15. • The values in SD5 to SD15 of the CPU module on other system are reflected.	S (Every END processing)	SD5 to SD15	QnPRH
SD1626 to SD1636	Other system error individual information	Error individual information	• Stores the individual information corresponding to the error code of the error that ccurred on the other system. • Data composition is the same as SD16 to SD26. • The values in SD16 to SD26 of the CPU module on other system are reflected.	S (Every END processing)	SD16 to SD26	QnPRH
SD1649	Standby system error cancel command	Error code of the error to be cleared	• This register stores the error code of the error to clear by clearing a standby system error. • The standby system error is cleared by turning SM1649 from off to on after storing the error code of the error to clear. • The least significant digit (ones place) of the error code in this register is ignored.(The errors corresponding to error codes 4100 to 4109 are cleared by storing 4100 in this register.)	S (Every END processing)		QnPRH
SD1650	Other system operating information		This register stores the operating status of the CPU module on the other system in the following bit pattern. When communications with other systems are disabled or the system is in the debug mode, "00FFH" is stored.  Note: Communications are disabled in the following states. • The other system is powered off or is being reset. • A hardware failure has occurred on the host or the other system. • A watchdog timer error has occurred on the host or the other system. • Tracking cable is not connected. Tracking cable is broken or failed.	S (Every END processing)		QnPRH

Tab. A-74: *Special registers (16): Redundant system (other system CPU information)*

Number	Name	Meaning	Description	Set by (if set)	Corresponding host CPU SD□□□	Valid for:
SD1690	Number of network module, which requested system switching on host (control) system		Stores the number of the network module which a system switch request was initiated, using the following format.  1) Module 0: CPU module is invalid as it is 2-slot model 2) Module 1: Module on right side of CPU module 3) Module 11: Module at rightmost end of 12-slot base (Q312B) • Turns off automatically by system, after network error is reset by user. • To find the number for the module where system switching is requested from a network module in the host system, refer to SD1590.	S (Every END processing)		QnPRH

Tab. A-74: *Special registers (16): Redundant system (other system CPU information)*

A.7.17 Redundant system (tracking information)

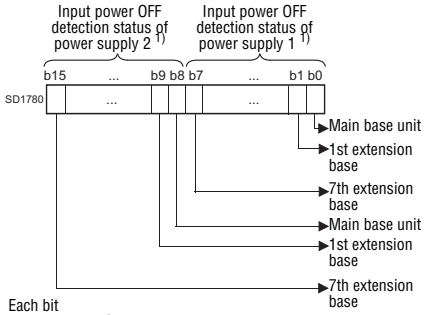
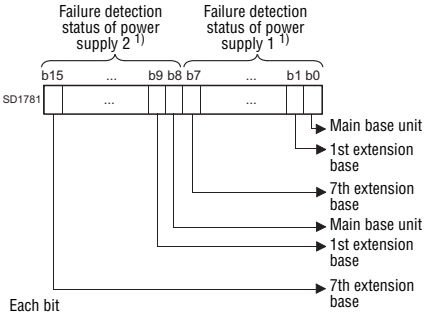
The special registers SD1700 to SD1799 are valid only for redundant systems. These registers are cleared to "0" in a stand-alone system.

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1700	Tracking error detection count		• A value in this register is incremented by one upon tracking error. • The counter starts the routine: counts up from 0 to 32767, then junps to -32768 and then again counts up to 0.	S (Error)	New	QnPRH
SD1710	Waiting time for online program change (standby system)		• This register stores the waiting time required for starting the online program change in the standby system after completion of that in the control system. The value is specified in units of seconds. • If online program change is not requested even after it is completed in the control system, the CPU modules in both of the system determine that it is a failure of an online program change for redundancy.In this case, both system CPU modules resume the consistency check for the systems that have been on hold during the online program change. Also, the control system is set to accept another request of online program change for redundancy. • When both systems are powered on, 90 seconds are set to SD1710 as the default value. • Set the value within the range 90 to 3600 seconds. When the setting is 0 to 89 seconds, it is regarded as 90 seconds for operation. If the setting is outside the allowed range, it is regarded other than 0 to 3600 seconds for operation. • The waiting time for a start of online program change to the standby system CPU module is checked according to the SD1710 setting during online change of multiple blocks and online change of batch of files for redundancy.	S (Initial) / U		

Tab. A-75: *Special registers (17): Redundant system (tracking information)*

A.7.18 Redundant power supply module information

The special registers SD1780 to SD1789 are valid only for redundant power supply systems. These registers are cleared to "0" in a stand-alone power supply system.

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1780	Power supply off detection status		• This register stores status of the redundant power supply module (Q63RP and Q64RP) with input power off, in the following bit pattern.  Each bit 0: Input power ON status/ No redundant power supply module 1: Input power OFF status • When the main base unit is not the redundant power main base unit (Q38RB), "0" is stored. • When configuring multiple CPU, the status is stored to 1st CPU module.	S (Every END processing)	New	Qn(H) 2) QnPH 2) QnPRH QnU 3)
SD1781	Power supply failure detection status		• This register stores failure detection status of the redundant power supply module (Q63RP and Q64RP) in the following bit pattern. (After a failure is detected on a redundant power supply module, the bit corresponding to the failed module turns to "0" upon turning off the module.)  Each bit 0: Redundant power supply module failure not detected/ No redundant power supply module 1: Redundant power supply module failure detected (Detectable for power supply module only) • When the main base unit is not the redundant power main base unit (Q38RB), "0" is stored. • When configuring multiple CPU, the status is stored to 1st CPU module.	S (Every END processing)	New	

Tab. A-76: *Special registers (18): Redundant power supply module information*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1782	Momentary power failure detection counter for power supply 1 [1]		• Counts the number of times of momentary power failure of the power supply 1/2. • Monitors the status of the power supply 1/ 2 mounted on the redundant power main base unit (Q38RB). • Status of power supply 1/ 2 mounted on the redundant extension base unit is not monitored. • When the CPU module starts, the counter of the power supply 1/ 2 is cleared to 0. • If the input power to one of the redundant power supply modules is turned OFF, the corresponding counter is cleared to 0. • The counter is incremented by one upon momentary power failure on the power supply 1 or 2. (The counter repeats increment and decrement of the value; 0 → 32767 → –32768 → 0. (The value is displayed within the range of 0 to 65535 in the system monitor screen of programming tool.)) • Stores 0 when the main base unit is not the redundant power main base unit (Q38RB). • When configuring multiple CPU, the status is stored to 1st CPU module.	S (Every END processing)	New	Qn(H) [2] QnPH [2] QnPRH QnU [3]
SD1783	Momentary power failure detection counter for power supply 2 [2]			S (Every END processing)	New	

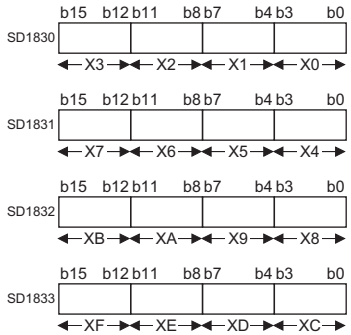
Tab. A-76: Special registers (18): Redundant power supply module information

[1] The "power supply 1" indicates the redundant power supply module mounted on the POWER 1 slot of the redundant base unit (Q38RB/Q68RB/Q65WRB).
The "power supply 2" indicates the redundant power supply module mounted on the POWER 2 slot of the redundant base unit (Q38RB/Q68RB/Q65WRB).

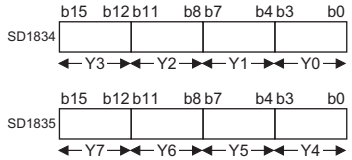
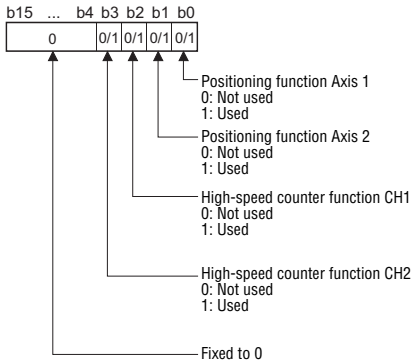
[2] The module whose first 5 digits of serial No. is "07032" or higher. However, for the multiple CPU system configuration, this applies to all CPU modules whose first 5 digits of serial No. are "07032" or higher.

[3] The module whose first 5 digits of serial No. is "10042" or higher.

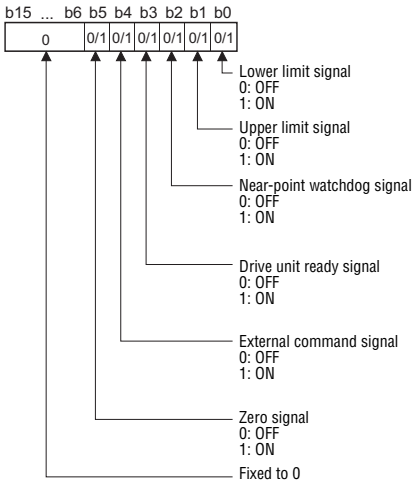
A.7.19 Built-in I/O function

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1830 SD1831 SD1832 SD1833	Functions selected for input signals		These registers store values indicating functions assigned for input signals. Each signal has 4-bit data.  Values are stored according to functions as follows: • 0: General-purpose input function • 1: Interrupt input function • 2: Pulse catch function • 3: Positioning function • 4: High-speed counter function	S (Every END processing)	New	LCPU

Tab. A-77: Special registers (19): Built-in I/O function

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1834 SD1835	Functions selected for output signals		These registers store values indicating functions assigned for output signals. Each signal has 4-bit data.  Values are stored according to functions as follows: • 0: General-purpose input function • 3: Positioning function • 4: High-speed counter function	S (Every END processing)	New	LCPU
SD1836	Operating status of positioning and high-speed counter functions		This register stores the operating status of the positioning function and high-speed counter function. 	S (Every END processing)	New	LCPU
SD1840 SD1841	Axis 1 current feed value		These registers store the current position value when the position where OPR control is completed is set as a base point. "0" is stored at power-on or reset of the CPU module. An OP address is stored at the completion of machine OPR control. This register is cleared to "0" when speed control in speed/position switching control is started. • When the current feed value is changed, the value after current value change is stored. • The current position read from a servo amplifier is stored at the completion of absolute position restoration.[1, 2)] [1] Range: -2147483648 to 2147483647 pulses [2] Since the internal update cycle of the storage value is 1 ms, the information of the current feed value may be older than the actual command position by 1 ms at maximum depending on the refresh timing at END processing.	S (Every END processing)	New	LCPU
SD1842 SD1843	Axis 1 current speed		These registers store the current speed. (Fractions are not stored. If the current speed is slower than 1 pulse/s, "0" may be displayed.) [1, 2)] [1] Range: 0 to 200000 pulses [2] Since the internal update cycle of the storage value is 1 ms, the information of the current speed value may be older than the actual speed by 1 ms at maximum depending on the refresh timing at END processing.	S (Every END processing)	New	LCPU

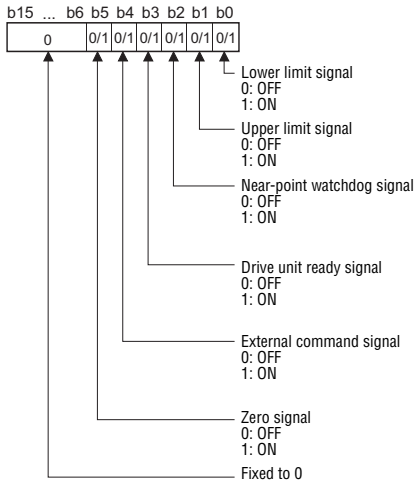
Tab. A-77: *Special registers (19): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1844	Axis 1 axis operation status		This register stores the axis operating status. • –1: Error occurring • 0: Standing by • 1: Stopped • 2: In JOG operation • 3: In OPR • 4: In position control • 5: In speed-position control (speed) (speed control in speed/position switching control) • 6: In speed-position control (position) (position control in speed/position switching control) • 7: Decelerating (axis stop ON) • 8: Decelerating (JOG start OFF) • 9: In high-speed OPR • 10: In speed control • 11: Analyzing	S (Every END processing)	New	LCPU
SD1845	Axis 1 error code		• This register stores the error code of the present axis error. • If another error occurs while an axis error occurs, the latest error code is not stored. • This register is cleared to "0" by turning on SM1850.	S (Every END processing)	New	LCPU
SD1846	Axis 1 warning code		• This register stores the warning code of the present axis warning. • If another axis warning occurs, the latest warning code is written to this register. • This register is cleared to "0" by turning on SM1850.	S (Every END processing)	New	LCPU
SD1847	Axis 1 external I/O signal		• This register stores the on/off status of external I/O signals. • When an OPR method with the OPR retry function (Near-point dog method, Count 1, Count 2) is performed, a value indicating the status of the upper limit signal or the lower limit signal is stored to the external command signal. [1)]  1 When an OPR method with the OPR retry function is performed even once, a value indicating the status of the upper limit signal or the lower limit signal will be stored until when speed/position switching control is started.	S (Every END processing)	New	LCPU
SD1848 SD1849	Axis 1 movement amount after near-point dog ON		• These registers store "0" when machine OPR control is started. • After machine OPR control is completed, these registers store a travel distance from the point where the near-point watchdog signal turns on (The point is set to "0".) to the point where machine OPR control is completed. Range: 0 to 2147483647 pulses • When an OPR method is set to Stopper 3, these registers always store "0".	S (Every END processing)	New	LCPU

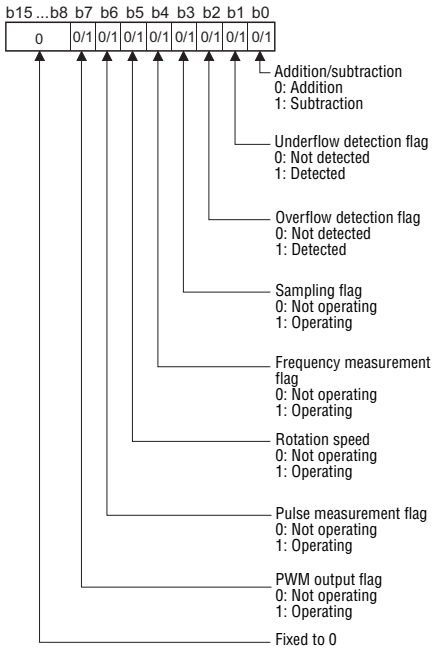
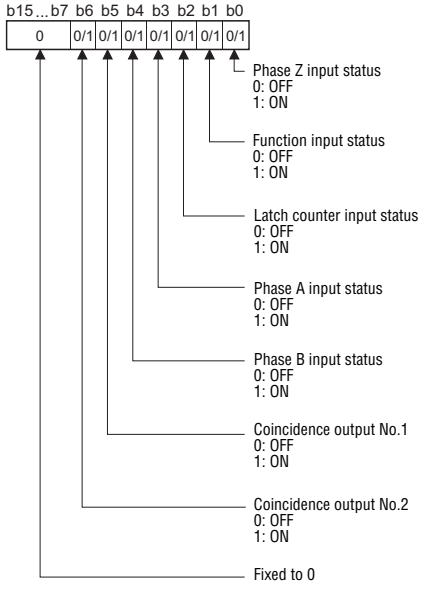
Tab. A-77: *Special registers (19): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1850	Axis 1 Data No. of positioning being executed		• This register stores the data No. of positioning being executed. (A storage value will be held until the next control starts.) • "0" is stored when JOG operation or machine OPR control is started. • "1" is stored when high-speed OPR control is started. • "1" is stored when positioning control is started by the IPDSTRT1 or IPDSTRT2 instruction. • If an error occurs at the start of positioning control, the previous value will be held.	S (Every END processing)	New	LCPU
SD1860 SD1861	Axis 2 current feed value		This register stores the current position value when the position where OPR control is completed is set as a base point. "0" is stored at power-on or reset of the CPU module. An OP address is stored at the completion of machine OPR control. This register is cleared to "0" when speed control in speed/position switching control is started. • When the current feed value is changed, the value after current value change is stored. • The current position read from a servo amplifier is stored at the completion of absolute position restoration. [1, 2)] [1] Range: -2147483648 to 2147483647 pulses. [2] Since the internal update cycle of the storage value is 1ms, the information of the current feed value may be older than the actual command position by 1ms at maximum depending on the refresh timing at END processing.	S (Every END processing)	New	LCPU
SD1862 SD1863	Axis 2 current speed		This register stores the current speed. (Fractions are not stored. If the current speed is slower than 1 pulse/s, "0" may be displayed.) [1, 2)] [1] Range: 0 to 200000 pulses [2] Since the internal update cycle of the storage value is 1ms, the information of the current speed value may be older than the actual command position by 1ms at maximum depending on the refresh timing at END processing.	S (Every END processing)	New	LCPU
SD1864	Axis 2 axis operation status		This register stores the axis operating status. • –1: Error occurring • 0: Standing by • 1: Stopped • 2: In JOG operation • 3: In OPR • 4: In position control • 5: In speed-position control (speed) (speed control in speed/position switching control) • 6: In speed-position control (position) (position control in speed/position switching control) • 7: Decelerating (axis stop ON) • 8: Decelerating (JOG start OFF) • 9: In high-speed OPR • 10: In speed control • 11: Analyzing	S (Every END processing)	New	LCPU
SD1865	Axis 2 error code		• This register stores the error code of the present axis error. • If another error occurs while an axis error occurs, the latest error code is not stored. • This register is cleared to "0" by turning on SM1870.	S (Every END processing)	New	LCPU
SD1866	Axis 2 warning code		• This register stores the warning code of the present axis warning. • If another axis warning occurs, the latest warning code is written to this register. • This register is cleared to "0" by turning on SM1870.	S (Every END processing)	New	LCPU

Tab. A-77: *Special registers (19): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1867	Axis 2 external I/O signals		• This register stores the on/off status of external I/O signals. • When an OPR method with the OPR retry function (Near-point dog method, Count 1, Count 2) is performed, a value indicating the status of the upper limit signal or the lower limit signal is stored to the external command signal.[1)]  1 When an OPR method with the OPR retry function is performed even once, a value indicating the status of the upper limit signal or the lower limit signal will be stored until when speed/position switching control is started.	S (Every END processing)	New	LCPU
SD1868 SD1869	Axis 2 movement amount after near-point dog ON		• This register stores "0" when machine OPR control is started. • After machine OPR control is started, this register stores a travel distance from the point where the near-point watchdog signal turns on (The point is set to "0".) to the point where machine OPR control is completed. Range: 0 to 2147483647 pulses • When an OPR method is set to Stopper 3, this register always stores "0".	S (Every END processing)	New	LCPU
SD1870	Axis 2 Data No. of positioning being executed		• This register stores the data No. of positioning being executed. (A storage value will be held until the next control starts.) • "0" is stored when JOG operation or machine OPR control is started. • "1" is stored when high-speed OPR control is started. • "1" is stored when positioning control is started by the IPDSTRT1 or IPDSTRT2 instruction. • If an error occurs at the start of positioning control, the previous value will be held.	S (Every END processing)	New	LCPU
SD1880 SD1881	CH1 current value		• These registers store the current counter value of CH1 at END processing. • When the ICCNTRD1 instruction is executed, this register is updated by the current value at that moment. The current value is updated at END processing and by the ICCNTRD1 instruction only when Normal Mode is set for Operation Mode Setting (high-speed counter function parameter). The range of a value that can be read is from -2147483648 to 2147483647.	S (Every END processing/ Instruction execution)	New	LCPU

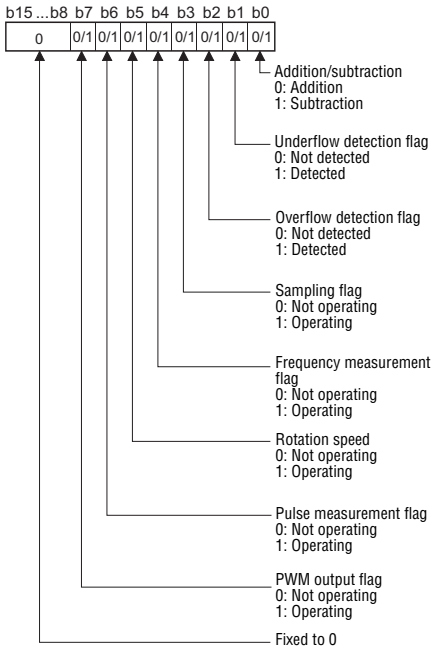
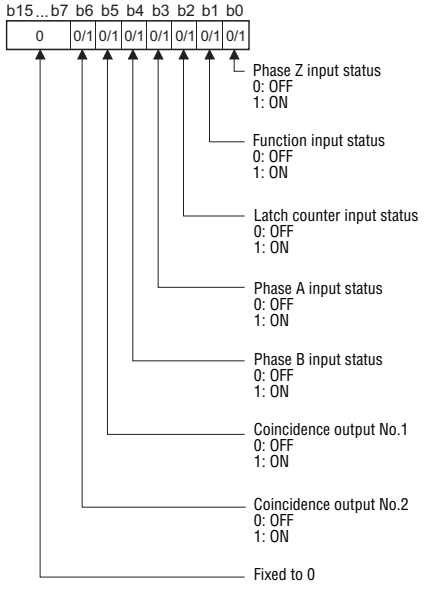
Tab. A-77: *Special registers (19): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1882	CH1 status monitor		This register stores each status of CH1. 	S (Every END processing)	New	LCPU
SD1883	CH1 external I/O status monitor		• This register stores a value indicating the external I/O signal status of CH1. • Unused signal status is fixed at off. • When Normal Mode is set for Operation Mode Setting (high-speed counter function parameter), a value according to the setting configured for Function Input Logic Setting (high-speed counter function parameter) is stored in the function input status. Therefore, when a voltage is applied to the function input terminal while Negative logic is set for Function input logic setting, this register turns off. • When other than A Phase/B Phase is selected for Count Source Selection (high-speed counter function parameter), the phase A input status and phase B input status are fixed at off. 	S (Every END processing)	New	LCPU

Tab. A-77: *Special registers (19): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1884	CH1 operation mode monitor		This register stores a value indicating the operation mode for high-speed counter of CH1 set by the parameter. 0: Not used 1: Normal mode 2: Frequency measurement mode 3: Rotation speed measurement mode 4: Pulse measurement mode 5: PWM output mode	S (Every END processing)	New	LCPU
SD1885	CH1 counter type monitor		• This register stores a value indicating the counter type for high-speed counter of CH1 set by the parameter. • Counter selection is disabled (fixed at "0") when a value stored to CH1 operation mode monitor (SD1884) is other than "1" (normal mode). 0: Linear counter 1: Ring counter	S (Every END processing)	New	LCPU
SD1886	CH1 selected counter function		• This register stores a value indicating the selected counter function for high-speed counter of CH1 set by the parameter. • Counter selection is disabled (fixed at "0") when a value stored to CH1 operation mode monitor (SD1884) is other than "1" (normal mode). 0: Count disabling function 1: Latch counter function 2: Sampling counter function 3: Count disabling/preset function 4: Latch counter/preset function	S (Every END processing)	New	LCPU
SD1887	CH1 error code		This register stores the error code of an error occurred in CH1.	S (Every END processing)	New	LCPU
SD1888	CH1 warning code		This register stores the warning code of a warning occurred in CH1.	S (Every END processing)	New	LCPU
SD1900	CH2 current value		• This register stores the current counter value of CH2 at END processing. • When the ICCNTRD2 instruction is executed, this register is updated by the current value at that moment. The current value is updated at END processing and by the ICCNTRD2 instruction only when Normal Mode is set for Operation Mode Setting (high-speed counter function parameter). The range of a value that can be read is from -2147483648 to 2147483647.	S (Every END processing/ Instruction execution)	New	LCPU
SD1901						

Tab. A-77: *Special registers (19): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1902	CH2 status monitor		This register stores each status of CH2. 	S (Every END processing)	New	LCPU
SD1903	CH2 external I/O status monitor		• This register stores a value indicating the external I/O signal status of CH2. • Unused signal status is fixed at off. • When Normal Mode is set for Operation Mode Setting (high-speed counter function parameter), a value according to the setting configured for Function Input Logic Setting (high-speed counter function parameter) is stored in the function input status. Therefore, when a voltage is applied to the function input terminal while Negative logic is set for Function input logic setting, this register turns off. • When other than A Phase/B Phase is selected for Count Source Selection (high-speed counter function parameter), the phase A input status and phase B input status are fixed at off. 	S (Every END processing)	New	LCPU

Tab. A-77: *Special registers (19): Built-in I/O function*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1904	CH2 operation mode monitor		This register stores a value indicating the operation mode for high-speed counter of CH2 set by the parameter. • 0: Unused • 1: Normal mode • 2: Frequency measurement mode • 3: Rotation speed measurement mode • 4: Pulse measurement mode • 5: PWM output mode	S (Every END processing)	New	LCPU
SD1905	CH2 counter type monitor		• This register stores a value indicating the counter type for high-speed counter of CH2 set by the parameter. • Counter selection is disabled (fixed at "0") when a value stored to CH2 operation mode monitor (SD1904) is other than "1" (normal mode). 0: Linear counter 1: Ring counter	S (Every END processing)	New	LCPU
SD1906	CH2 selected counter function		• This register stores a value indicating the selected counter function for high-speed counter of CH2 set by the parameter. • Counter selection is disabled (fixed at "0") when a value stored to CH2 operation mode monitor (SD1904) is other than "1" (normal mode). 0: Count disabling function 1: Latch counter function 2: Sampling counter function 3: Count disabling/preset function 4: Latch counter/preset function	S (Every END processing)	New	LCPU
SD1907	CH2 error code		This register stores the error code of an error occurred in CH2.	S (Every END processing)	New	LCPU
SD1908	CH2 warning code		This register stores the warning code of a warning occurred in CH2.	S (Every END processing)	New	LCPU

Tab. A-77: *Special registers (19): Built-in I/O function*

A.7.20 Data logging

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1940	Data logging setting No. 1	Latest file No.	This register stores the latest file number. This register is cleared to "0" by a stop command from LCPU Logging Configuration Tool.	S (Status change)	New	LCPU
SD1941						
SD1942		Oldest file No.	This register stores the oldest file number. This register is cleared to "0" by a stop command from LCPU Logging Configuration Tool.	S (Status change)	New	LCPU
SD1943						
SD1944		Free buffer space	This register stores free buffer space (unit: 1K byte). If the value is small, processing overflow may occur. For trigger logging, this register stores the buffer size until when data are collected by the number of records after trigger. This register is cleared to "0" by a stop command from LCPU Logging Configuration Tool.	S (Status change)	New	LCPU
SD1945		Number of times processing overflow occurred	This register stores the number of times that data logging processing overflow occurred. If an overflow occurs, some data may not be collected. When the storage value reaches to 65535, count is resumed from "0". If Stop is specified for Operation occurring when number of saved files is exceeded, processing overflow may occur from when data collection by the number of specified storage files is completed and until when data logging is stopped. This register is cleared to "0" by the registration of the setting or a stop command from LCPU Logging Configuration Tool.	S (Error)	New	LCPU

Tab. A-78: *Special registers (20): Data logging*

Number	Name	Meaning	Description	Set by (if set)	ACPU D9□□□	Valid for:
SD1946	Data logging setting No.1	Data logging error cause	This register stores the cause of an error that occurred during data logging. • 0: No error • Other than 0: For values stored at error occurrence, refer to the errors that occurs in data logging described in the following manual (MELSEC-L CPU Module User's Manual (Data Logging Function)). This register is cleared to "0" by the registration of the setting or a stop command from LCPU Logging Configuration Tool.	S (Error)	New	LCPU
SD1950 to SD1956	Data logging setting No.2	Same as in data logging setting No. 1	Same as in data logging setting No. 1 (SD1940 to SD1946)	Same as in data logging setting No. 1	New	LCPU
SD1960 to SD1966	Data logging setting No.3					
SD1970 to SD1976	Data logging setting No.4					
SD1980 to SD1986	Data logging setting No.5					
SD1990 to SD1996	Data logging setting No.6					
SD2000 to SD2006	Data logging setting No.7					
SD2010 to SD2016	Data logging setting No.8					
SD2020 to SD2026	Data logging setting No.9					
SD2030 to SD2036	Data logging setting No.10					

Tab. A-78: *Special registers (20): Data logging*

Index

A

ACOSD, ACOSDP 7-378
ACOS, ACOSP 7-375
Adding
clock data 7-494
expansion clock data 7-525
Addressing of
arrays 3-23
registers 3-22
ADRSET, ADRSETP 7-562
ANB 5-13
ANDFI 5-10
ANDPI 5-10
ANDP, ANDF 5-7
AND, ANI 5-4
ANY (data type) 3-9
Application instructions, Part 1 6-1
Application Instructions, Part 2 7-1
Arcus cosine calculation
from BCD data 7-448
of floating point values (double precision) 7-379
of floating point values (single precision) 7-376
Arcus sine calculation
from BCD data 7-445
from floating point values (double precision) 7-373
from floating point values (single precision) 7-370
Arcus tangent calculation
from BCD data 7-451
of floating point values (double precision) 7-385
of floating point values (single precision) 7-382
Arithmetic Operation instructions (overview) . . 2-14
Array address/ initial address conversion 3-24
Arrays (adressing) 3-23
ASC, ASCP 7-312
ASIND, ASINDP 7-372
ASIN, ASINP 7-369
ATAND, ATAND 7-384
ATAN, ATANP 7-381

B

BACOS, BACOSP 7-447
BAND, BANDP 7-458
BASIN, BASINP 7-444
BATAN, BATANP 7-450
Batch
save and recovery of index register contents 7-571
BBLKRD, BBLKRDP 12-27
BBLKWR, BBLKWRP 12-30
BCD data
arithmetic operation instructions 6-37
BCD 4-digit
addition and subtraction operations 6-53
multiplication and division operations 6-61
BCD 8-digit
addition and subtraction operations 6-57
multiplication and division operations 6-64
BCDDA, BCDDAP 7-258
BCD, BCDP 6-103
BCOS, BCOSP 7-438
BDSQR, BDSQRP 7-431
BIN block data
transfer 6-167
BIN data
arithmetic operation instructions 6-37
inversion 6-162
transfer 6-152
BIN data exchange 6-177
BIN 16-bit
addition and subtraction operations . 6-39, 6-84
increment and decrement operations 6-95
multiplication and division 6-46
BIN 16-bit and 32-bit
dead band control 7-459
zone control 7-463
BIN 16-bit block data
exchange 6-180
BIN 32-bit
addition and subtraction operations . 6-43, 6-88
increment and decrement operations 6-98
multiplication and division 6-50
BINDA, BINDAP 7-248
BINHA, BINHAP 7-253
BIN, BINP 6-106
Bit blocks 3-11
of double word data 3-14
of word data 3-13
Bit device output inversion 5-48
Bit devices
of double word data 3-14
of word data 3-12
usage in instructions 3-11
Bit processing instructions 7-67
Bit processing instructions (overview) 2-32
Bit test
16-bit 7-72
32-bit 7-72
BKAND, BKANDP 7-10
BKBCD, BKBCDP 6-140
BKBIN, BKBINP 6-143
BKCMP, BKCMPP 6-25
BKOR, BKORP 7-18
BKRST, BKRSTP 7-75
BKXNR, BKXNRP 7-35
BKXOR, BKXORP 7-26
BK+, BK+P, BK-, BK-P 6-83

BMOV, BMOVP 6-166
BREAK, BREAKP 7-137
BRST, BRSTP 7-68
BSET, BSETP 7-68
BSFL, BSFLP 7-55
BSFR, BSFRP 7-55
BSIN, BSINP 7-435
BSQR, BSQRP 7-431
BTAN, BTANP 7-441
BTOW, BTOWP 7-111
Buffer memory access instructions (overview) . 2-39
BUFRCV 12-34
BUFRCVS (ETHERNET interface module) . . 12-39
BUFRCVS (Serial communication modules) . . 12-3
BUFSND 12-42
Building an input matrix 6-233
Built-in Ethernet port
  Special registers A-207
  Special relays A-139
Built-in I/O function
  Special registers A-219
  Special relays A-149
BXCH, BXCHP 6-179
Bx, BxP, B/, B/P 6-60
Byte exchange 6-183
B+, B+P, B-, B-P 6-52

C

Calculating
  averages of 16-/32-bit BIN data 7-131
  totals of 16-bit BIN data blocks 7-127
  totals of 32-bit BIN data blocks 7-129
CALL, CALLP 7-140
Categories of instructions (overview) 2-1
CCOM, CCOMP 7-177
Changing
  clock data format 7-504
Character string
  extract from right or left 7-320
  insert 7-334
  linking operations 6-92
  search 7-330
  STRING instruction 3-25
Character string data
  transfer 6-159
Character string processing instructions 7-245
Character string processing instructions
(overview) 2-41
Character strings
  delete 7-337
  detect length 7-281
  storing and moving 7-324
Check data bits 7-88
CHKCIR, CHKEND 7-240
CHKST, CHK 7-232
CJ 6-186
Clock data
  adding of 7-494
  format changing 7-504
  reading of 7-488
  subtraction of 7-499
  writing of 7-491
Clock instructions 7-485
Clock instructions (overview) 2-51
CLOSE 12-56
CML, CMLP 6-161
COM (Refresh) 7-170
COM (Selective Refresh) 7-173
Common logarithm calculation from
floating point values (double precision) 7-427
Common logarithm calculation from
floating point values (single precision) 7-424
Comparision operation instructions (overview) 2-10
Comparison
  BIN block data 6-26
  BIN 16-bit data 6-7
  BIN 32-bit block data 6-31
  Character string data 6-21
  clock data 7-515
  Counters A-109
  Display instructions A-110
  Floating point data 6-13
  of CPU modules A-102
  Timer functions A-105
  32-bit data 6-10
Comparison operation instructions 6-2
  D=, D<>, D>, D<=, D<, D>= 6-10
  E=, E<>, E>, E<=, E<, E>= 6-13
  =, <>, >, <=, <, >= 6-7
  $=, $<>, $>, $<=, $<, $>= 6-21
COMRD, COMRDP 7-276
Connection instructions 5-13
Connection instructions (overview) 2-7
Connection (ETHERNET)
  close 12-58
  open of 12-50
Constants (use in an instruction) 3-1
Conversion
  from BCD block data into BIN block data 6-144
  from BCD data into BIN data 6-107
  from BIN block data into BCD block data 6-141
  from BIN data into BCD data 6-104
  from BIN data into floating point data
  (double precision) 6-114
  from BIN data into floating point data
  (single precision) 6-111
  from BIN data into Gray code data 6-128
  from BIN 16-bit data into BIN 32-bit data 6-124
  from BIN 32-bit data into BIN 16-bit data 6-126
  from degrees into radian as
  floating point value (double precision) . . 7-391
  from degrees into radian as
  floating point value (single precision) . . . 7-388
  from Double precision to Single precision 6-149

from floating point data into BIN data (double precision) 6-121
from floating point data into BIN data (single precision) 6-117
from Gray code data into BIN data 6-131
from radian in floating point format into degrees (double precision) 7-397
from radian in floating point format into degrees (single precision) 7-394
from Single precision to Double precision 6-147
of BIN 16-bit data into ASCII code 7-313
of BIN 16-/32-bit binary data into character strings 7-284
of character string data into decimal floating point data 7-307
of character strings into BIN 16-/32-bit binary data 7-291
of decimal ASCII data into BIN 16-/32-bit binary data 7-264
of decimal ASCII data into 4-/8-digit BCD data 7-273
of floating point data into character string data 7-297
of floating point data into the decimal format 7-344
of floating point number into the BCD format 7-340
of hexadecimal ASCII data into BIN 16-/32-bit binary data 7-269
of hexadecimal ASCII values into binary values 7-316
of 16-/32-bit binary data into decimal values in ASCII code 7-249
of 4-/ 8-digit BCD data into ASCII code . . 7-259
of16-/32-bit binary data into hexadecimal values in ASCII code 7-254
Conversion from A series to MELSEC System Q or L series
Special registers A-199
Special relays A-132
COSD, COSD 7-360
Cosine calculation
from BCD data 7-439
from floating point values (double precision) 7-361
from floating point values (single precision) 7-358
COS, COSP 7-357
Count-up/-down counter
1-phase 6-207
2-phase 6-210
CPU shared memory
reading from 9-15
writing to 9-5, 9-9
CPU table in instruction descriptions 4-2
CPU types 1-2

D

DABCD, DABCDP 7-272
DABIN, DABINP 7-263
DAND, DAND 7-4
Data
decoding 7-91
encoding 7-94
Data control instructions 7-453
Data control instructions (overview) 2-48
Data conversion instructions 6-101
Data conversion instructions (overview) 2-20
Data link instructions 8-1
Data logging
Special registers A-227
Special relays A-154
Data processing instructions 7-79
Data processing instructions (overview) 2-33
Data refresh instructions 8-1
Data refresh instructions (overview) 2-25
Data rotation
to the left (16-bit) 7-43
to the left (32-bit) 7-49
to the right (16-bit) 7-40
to the right (32-bit) 7-46
Data rotation instructions 7-38
Data rotation instructions (overview) 2-30
Data shift instructions 7-51
Data table instructions (overview) 2-38
Data table operation instructions 7-187
Data transfer instructions 6-150
Data transfer instructions (overview) 2-23
Data type
REAL 3-17
Data types 3-9
Date comparison 7-510
DATERD, DATERDP 7-487
DATEWR, DATEWRP 7-490
DATE-, DATE-P 7-498
DATE+, DATE+P 7-493
DBAND, DBANDP 7-458
DBCDDA, DBCDDAP 7-258
DBCD, DBCDP 6-103
DBINDA, DBINDAP 7-248
DBINHA, DBINHAP 7-253
DBIN, DBINP 6-106
DBKCMP, DBKCMPP 6-30
DBK+, DBK+P, DBK-, DBK-P 6-87
DBL, DBLP 6-123
DBx, DBxP, DB/, DB/P 6-63
DB+, DB+P, DB-, DB-P 6-56
DCML, DCMLP 6-161
DDABCD, DDABCDP 7-272
DDABIN, DDABINP 7-263
Dead band control 7-453
Debugging
Instructions 7-231
Special registers A-198

Special relays A-131
Debugging and failure diagnosis instructions (overview) 2-40
Decoding
from 8 to 256 bits 7-91
7-segment code 7-97
DECO, DECOP 7-90
Dedicated instructions 3-6
DEGD, DEGDP 7-396
DEG, DEGP 7-393
Delete
of character strings 7-337
program from memory 7-622
Deleting data blocks in a data table 7-201
DELTA 5-49
DELTAP 5-49
Destination of data (d) 3-2
Detecting the length of character strings 7-281
Devices
Multiple Instructions using the same device . 3-61
Devices in instruction descriptions 4-3
DFLT, DFLTP 6-110
DFRO, DFROP 7-208, 9-14
DGBIN, DGBINP 6-130
DGRY, DGRYP 6-127
DHABIN, DHABINP 7-268
DI 6-193
Diagnostic information
Special registers A-157
Special relays A-114
DINC, DINCP, DDEC, DDECP 6-97
DINT (data type) 3-22
DINT, DINTP 6-116
Direct read of one byte from a file register . . . 7-555
Direct write of one byte to a file register 7-559
Display instructions 7-217
Display instructions (overview) 2-39
Disuniting
data in byte units 7-112
data in random bit groupings 7-107
16-bit data 7-101
Disuniting and uniting data in byte units 7-112
Disuniting or uniting of data in random bit groupings 7-107
DIS, DISP 7-100
DLIMIT, DLIMITP 7-454
DMAX, DMAXP 7-116
DMEAN, DMEANP 7-130
DMOV, DMOVP 6-151
DNEG, DNEGP 6-133
DOR, DORP 7-13
Double precision floating-point data 3-19
DRCL, DRCLP 7-48
DRCR, DRCRP 7-45
Drive information
Special relays A-125
DROL, DROLP 7-48
DROR, DRORP 7-45
DSCL, DSCLP 7-466
DSCL2, DSCL2P 7-471
DSER, DSERP 7-81
DSFL, DSFLP 7-61
DSFR, DSFRP 7-61
DSTR, DSTRP 7-283
DSUM, DSUMP 7-87
DTEST, DTESTP 7-71
DTO, DTOP 7-212, 9-8
DT=, DT, DT>, DT= 7-509
DUTY 7-549
DVAL, DVALP 7-290
DWORD 3-22
DWSUM, DWSUMP 7-128
DXCH, DXCHP 6-176
DXNR, DXNRP 7-29
DXOR, DXORP 7-21
Dx, DxP, D/, D/P 6-49
DZONE, DZONEP 7-462
D+, D+P, D-, D-P 6-42
D=, D<>, D>, D<=, D<, D>= 6-9

E

ECALL, ECALLP 7-153
ECON, ECONP 6-146
EDMOV, EDMOVP 6-156
EDx, EDxP, ED/, ED/P 6-80
ED+, ED+P, ED-, ED-P 6-72
EFCALL, EFCALLP 7-159
EGF 5-22
EGP 5-22
EI 6-193
EMOD, EMODP 7-339
EMOV, EMOVP 6-154
EN input 3-53
Encoding
from 256 to 8 bits 7-94
ENCO, ENCOP 7-93
END 5-63
End of main routine program 5-61
End of sequence program 5-64
End of subroutine program 7-147
ENEG, ENEGP 6-136
ENO output 3-53
EREXP, EREXPP 7-343
ERRCLR 12-61
Error
clearing 13-2
Error code (ETHERNET module)
clearing of 12-63
reading of 12-69
Error codes
List 13-1
types 13-2
ERRRD 12-67
ESTR, ESTRP 7-296

ETHERNET
Deletion of "ERR"-LED 12-63
reading of an error code 12-69
Re-initialisation 12-74
EVAL, EVALP 7-306
Exchange
upper and lower byte 6-183
Exclusive NOR
operations with 16-bit data blocks 7-36
Exclusive OR
operations with 16-bit data blocks 7-27
Execution conditions
for comparison operation instructions 6-4
Representation 2-5
Execution conditions of the instructions 3-52
Expansion clock data
adding of 7-525
reading of 7-521
subtraction of 7-529
Expansion clock instructions 7-519
Expansion clock instructions (overview) 2-53
EXPD, EXPDP 7-414
Exponentiation
of floating point data (double precision) . 7-403
of floating point data (single precision) . . 7-400
EXP, EXPP 7-411
Extraction
of character string data 7-320
Ex, ExP, E/, E/P 6-77
E+, E+P, E-, E-P 6-67
E=, E<>, E>, E< =, E<, E>= 6-12

F

Failure check for bidirectional operations . . . 7-233
Failure diagnosis and debugging instructions 7-231
FCALL, FCALLP 7-148
FDEL, FDELP 7-200
FEND 5-60
FF 5-47
FIFR, FIFRP 7-192
FIFW, FIFWP 7-188
File register switching instructions 7-475
File register switching instructions (overview) . 2-50
File registers
Precautions 3-66
Finding an instruction 1-3
FINS, FINSP 7-200
Floating point data
addition and subtraction operations (double precision) 6-73
addition and subtraction operations (single precision) 6-68
multiplication and division operations (double precision) 6-81
multiplication and division operations (single precision) 6-78
transfer (double precision) 6-157
transfer (single precision) 6-155
Floating point values as exponent of the base e (double precision) 7-415
Floating point values as exponent of the base e (single precision) 7-412
Floating-point data
double precision 3-19
Monitoring with programming tool 3-20
single precision 3-17
FLT, FLTP 6-110
FMOV, FMOVP 6-170
FOR, NEXT 7-134
FOR/NEXT loop instruction 7-135
FPOP, FPOPP 7-196
FROM, FROMP 7-208, 9-14
Functions in instruction descriptions 4-5
Fuse blown module
Special registers A-210

G

GBIN, GBINP 6-130
Generating check circuits for the CHK instruction 7-241
Generating pulses at direct access outputs . . 5-50
GETE, GETEP 12-6
GOEND 6-190
GRY, GRYP 6-127

H

HABIN, HABINP 7-268
Header of the MELSEC-AWL 3-7
HEX, HEXP 7-315
Hierarchy of data types ANY 3-9
High-speed block transfer of file register . . . 7-628
HOUR, HOURP 7-503

I

IEC Commands for comparison 6-4
Iinstructions
for failure diagnosis and debugging 7-231
IMASK 6-193
INC, INCP, DEC, DECP 6-94
Index qualification
General 3-27
Index qualification of entire program parts . . 7-180
Indexed device numbers
storing in an index qualification table . . . 7-185
Indirect adress
storing 7-563
Indirect designation 3-39
Initial registers 3-22

Input instructions 5-4
Input instructions (overview) 2-6
Insert
of character strings 7-334
Inserting data blocks in a data table 7-201
Instruction for MELSECNET/H 2-63
Instruction names 2-4
Instruction related registers
Special registers A-194
Instruction related special relays
Special relays A-128
Instruction (structure) 3-1
Instructions
Applicable to subset processing 3-56
Buffer memory access 7-207
Display 7-217
Execution conditions 3-52
expansion clock data 7-519
for a multi-CPU system 9-1
for a redundant system 11-1
for character string processing 7-245
for clock data 7-485
for data control 7-453
for data refresh 8-1
for file register switching 7-475
for program control 7-532
for special function modules 12-1
Multiple CPU device write/read 10-1
operation errors 3-44
read from CPU shared memory 9-13
writing to the CPU shared memory 9-2
Instructions for CC-Link (overview) 2-64
Instructions for ETHERNET interface modules (overview) 2-63
Instructions for PROFIBUS/DP interface modules (overview) 2-62
Instructions for reading from the CPU shared memory of another CPU (overview) 2-59
Instructions for serial communication modules (overview) 2-62
Instructions for writing to the CPU shared memory of host CPU (overview) 2-59
Instructions (overview) 2-4
INSTR, INSTRP 7-329
Interrupt
Program execution 2-25
Interrupt control instructions 6-194
Interrupt control instructions (overview) 2-25
INT, INTP 6-116
INV 5-18
IRET 6-200
IXDEV, IXSET 7-184
IX, IXEND 7-179
I/O module verification
Special registers A-210
I/O partial refresh 6-204
I/O refresh
Special relays A-124

J

JMP 6-186
Jump instructions 6-187
Jump to the end of a program 6-191

K

KEY 7-564

L

L series CPUs 1-2
Ladder block parallel connection 5-14
Ladder block series connection 5-14
LDFI 5-10
LDPI 5-10
LDP, LDF 5-7
LD, LDI 5-4
Leading edge and trailing edge output 5-44
LEDR 7-227
LEFT, LEFTP 7-319
LEN, LENP 7-280
Limitation of output values 7-455
LIMIT, LIMITP 7-454
Link refresh instructions 6-202
Load inverse (normally closed contact) 5-5
Load (normally open contact) 5-5
Logarithm (ln) calculation from floating point values (double precision) 7-421
Logarithm (ln) calculation from floating point values (single precision) 7-418
LOGD, LOGDP 7-420
Logical
AND 7-5
exclusive NOR 7-30
exclusive OR 7-22
OR 7-14
product with 16-bit data blocks 7-11
sum with 16-bit data blocks 7-19
Logical operation instruction (overview) 2-27
Logical operation instructions 7-2
LOG, LOGP 7-417
LOG10D, LOG10DP 7-426
LOG10, LOG10P 7-423

M

Master Control Instructions 5-54
Master control instructions (overview) 2-9
MAX, MAXP 7-116
MC 5-54
MCR 5-54
MEAN, MEANP 7-130
MEF 5-20
MELSEC System Q CPUs 1-2
MELSECNET
  Pairing setting of stations 12-80
Memory cards
  Special registers A-185
MEP 5-20
MIDR, MIDRP 7-323
MIDW, MIDWP 7-323
Miscellaneous Instructions 5-66
Miscellaneous instructions (overview) 2-9
Moving parts of character strings 7-324
MOV, MOVP 6-151
MPP 5-15
MPS 5-15
MRD 5-15
MTR 6-232
Multiple CPU high-speed transmission dedicated instructions (overview) 2-60

N

NDIS, NDISP 7-106
NEG, NEGP 6-133
Network data refresh 8-3
Network refresh instructions (overview) 2-58
No operation program step 5-69
NOP 5-68
NOPLF 5-68
Notation of instructions 3-3
Notes in instruction descriptions 4-6
Number (n) 3-2
Numerical key input 7-565
NUNI, NUNIP 7-106

O

OPEN 12-47
Operation errors
  when an instruction is executed 3-44
Operation errors in instruction descriptions 4-6
Operation result
  conversion into pulse 5-21
  inversion 5-19
  processing 5-16
Operation start 5-5
ORB 5-13
ORFI 5-10
ORPI 5-10
ORP, ORF 5-7
OR, ORI 5-4
Other convenient instructions, Part 1 6-205
Other convenient instructions, Part 2 7-546
Other instructions, Part 2 (overview) 2-55
OUT 5-24
OUT C 5-30
OUT F 5-33
OUT T 5-26
OUTH T 5-26
Output
  of a comment 7-224
  of an ASCII character string 7-219
Output Instructions 5-24
Output instructions (overview) 2-8
Output of annunciators 5-34
Overview
  Arithmetic operation instructions 2-14
  available devices A-102
  Bit processing instructions 2-32
  Buffer memory access instructions 2-39
  Categories of instructions 2-1
  Character string processing instructions . 2-41
  Clock instructions 2-51
  Comparison operation instructions 2-10
  Connection instructions 2-7
  CPU types 1-2
  Data control instructions 2-48
  Data conversion instructions 2-20
  Data processing instructions 2-33
  Data refresh instructions 2-25
  Data table instructions 2-38
  Data transfer instructions 2-23
  Data types A-104
  Debugging and failure diagnosis instructions 2-40
  Display instructions 2-39
  Expansion clock instructions 2-53
  File register switching instructions 2-50
  Further manuals 1-1
  Input instructions 2-6
  Instruction Tables 2-1
  Instructions 2-4
  Instructions for CC-Link 2-64
  Instructions for ETHERNET interface modules 2-63
  Instructions for PROFIBUS/DP interface modules 2-62
  Instructions for reading from the CPU shared memory of another CPU . . . 2-59
  Instructions for serial communication modules 2-62
  Instructions for writing to the CPU shared memory of host CPU 2-59
  Interrupt control instructions 2-25
  I/O control modes A-104
  Logical operation instructions 2-27

Master control instructions 2-9
Miscellaneous instructions 2-9
Multiple CPU high-speed transmission dedicated instructions 2-60
Network refresh instructions. 2-58
Other convenient instructions. 2-26
Other instructions, Part 2 2-55
Output instructions 2-8
Processing times (definition) A-1
Program branch instructions 2-25
Program instructions 2-54
Program termination instructions 2-9
QCPU/LCPU instructions equivalent to A series instructions A-111
Read/write routing information 2-58
Rotation instructions 2-30
Shift instructions 2-9, 2-31
Special function instructions. 2-44
Special registers A-156
Special relays A-113
SPS types 1-2
Structured program instructions 2-36

P

PAGE n 5-68
PAIRSET 12-79
Parallel connection
of contacts 5-5
PCHK. 7-543
PLC parameters 1-4
PLF 5-43
PLOADP 7-617
PLOW, PLOWP 7-541
PLS 5-43
PLSY 6-228
POFF, POFFP 7-536
Positioning instruction for rotary tables 6-219
POWD, POWDP 7-402
POW, POWP 7-399
PR 7-218
PRC. 7-223
Presetting the number of execution scans of a device 7-550
Process control instruction
Special registers A-211
Special relays A-140
Processing
Data of data type REAL. 3-17
of bit data 3-11
of double word data (32 bits) 3-14
of word data (16 bits). 3-12
Processing Time
LCPU A-79
Processing times
tables for MELSEC System Q CPUs A-2
Program
Delete from program memory 7-622
loading from a memory card 7-618
Number of program steps 3-55
Unload and load. 7-625
Program branch instructions 6-185
Program branch instructions (overview) 2-25
Program control instructions 7-532
Program examples in instruction descriptions. . 4-7
Program execution control instructions 6-192
Program execution status check instruction . 7-544
Program instructions (overview) 2-54
Program jumps. 6-187
Program organisation unit (POU). 3-22
Program steps (number) 3-55
Program termination instructions (overview) . . . 2-9
Programmable (teaching) Timer 6-213
Programming
of dedicated instructions 3-6
of variables 3-7
PRR, PRRP 12-18
PSCAN, PSCANP 7-539
PSTOP, PSTOPP 7-534
PSWAPP 7-624
Pulse
NOT parallel connection 5-11
NOT series connection 5-11
operation start 5-8
parallel connection 5-8
series connection 5-8
Pulse density measurement 6-226
Pulse output with adjustable number of pulses 6-229
Pulse width modulation. 6-231
PUNLOADP. 7-621
PUTE, PUTEP 12-11
PWM 6-230

Q

QCDSET, QCDSETP 7-482
QDRSET, QDRSETP 7-479

R

RADD, RADDP 7-390
RAD, RADP 7-387
RAMP 6-222
Ramp signal. 6-223
Randomizing values and series update 7-430
RBMOV, RBMOVP. 7-627
RCL, RCLP 7-42
RCR, RCRP. 7-39
Read
data from Standard ROM 7-616
device comment data 7-277

Reading
clock data. 7-488
data from a designated file 7-602
expansion clock data 7-521
module information 7-575
of user registered frames. 12-8
routing information 8-9
Reading and writing routing information 8-7
Reading data
entered first from a data table 7-193
entered last from a data table 7-197
from a special function module 7-209
1-word data (16-bit) from a special function module. 7-209
2-word data (32-bit) from a special function module. 7-209
Reading devices from another CP 10-20
Reading of data
from a communication module 12-4
from a PROFIBUS interface module. . . . 12-28
from another PLC (CC-Link) 12-99
from automatic updating buffer 12-130
from fixed buffer 12-36
from intell. device station (CC-Link) . . . 12-114
in an interrupt program 12-40
Read/write routing information (overview). . . . 2-58
REAL (data type) 3-17
Redundant CPU information
Special registers. A-198
Redundant power supply module
Special registers. A-218
Special relays. A-148
Redundant system information (host CPU)
Special registers. A-212
Special relays. A-140
Redundant system information (other CPU)
Special registers. A-214
Special relays. A-144
Redundant system (tracking information)
Special registers. A-217
Special relays. A-145
Refresh
for link and interface data 7-171
Refresh instruction 7-171
Remote password count
Special registers. A-198
Reset
batch reset of bits 7-76
Resetting
annunciators and error displays 7-228
devices 5-38
outputs in subroutine programs in program files (EFCALL). 7-160
outputs in subroutine programs (FCALL) 7-149
watchdog timer. 7-548
RET. 7-146
Return from an interrupt program. 6-201
RFS, RFSP 6-203
RIFR. 12-130
RIGHT, RIGHTP 7-319
RIRCV 12-111
RIRD. 12-95
RISEND 12-119
RITO. 12-126
RIWT 12-103
RLPASET 12-83
RND, RNDP 7-429
ROL, ROLP. 7-42
ROR, RORP 7-39
Rotation instruction
left (16-bit), with carry flag. 7-43
left (16-bit), without carry flag 7-43
left (32-bit), with carry flag. 7-49
left (32-bit), without carry flag 7-49
right (16-bit), with carry flag 7-40
right (16-bit), without carry flag 7-40
right (32-bit), with carry flag 7-46
right (32-bit), without carry flag 7-46
Rotation instructions (overview) 2-30
ROTC 6-218
RSET, RSETP. 7-476
RST 5-37
RST F 5-40

S

Scaling
Point-by-point coordinate data 7-467
Scan information
Special registers A-182
Special relays A-124
SCJ. 6-186
SCL, SCLP 7-466
SCL2, SCL2P 7-471
Search
data 7-82
for character strings 7-330
maximum values in 16-/32-bit data 7-117
minimum values in 16-/32-bit data. 7-120
16-bit data. 7-82
32-bit data. 7-84
SECOND, SECONDP 7-503
SEG, SEGP. 7-96
Select refresh instruction 7-174, 7-178
Sequence instructions 5-1
Sequence program stop 5-67
Series connection of contacts 5-5
SER, SERP. 7-81
SET 5-35
SET F 5-40
Setting
comment files 7-483
file register blocks 7-477
file register files 7-480

Setting a program
into the low-speed execution mode 7-542
into the scan execution mode 7-540
into the stand-by mode 7-535
into the stand-by mode including reset of the outputs 7-537
Setting and resetting
single bits 7-69
master control 5-55
of annunciators 5-41
Setting counters 5-31
Setting of devices 5-36
Setting of edge relays 5-22
Setting output 5-25
Setting timers 5-27
SFL, SFLP 7-52
SFR, SFRP 7-52
SFT 5-51
SFTBL, SFTBLP 7-58
SFTBR, SFTBRP 7-58
SFTP 5-51
SFTWL, SFTWLP 7-64
SFTWR, SFTWRP 7-64
Shift instructions
Bit devices 5-51
Shift instructions (overview) 2-9
Shift instructions part 2 (overview) 2-31
Shifting
bit devices 5-52
n bit devices by n bits 7-59
n bit devices by 1 bit 7-56
n word devices by n words 7-65
n word devices by 1 address 7-62
16-bit data word by n bits 7-53
Sign reversal
for BIN data (complement of 2) 6-134
for floating point data (double precision) . 6-139
for floating point data (single precision) . . 6-137
SIND, SINDP 7-354
Sine calculation
from BCD data 7-436
from floating point values (double precision) 7-355
from floating point values (single precision) 7-352
Single precision floating-point data 3-17
SIN, SINP 7-351
Software 1-2
Sorting
16-/32-bit data 7-123
SORT, DSORT 7-122
Source of data (s) 3-1
SPD 6-225
Special function instructions (overview) 2-44
Special function timer 6-215
Special functions 7-347
Special registers
Built-in Ethernet port A-207
Built-in I/O function A-219
Conversion from A series to MELSEC System Q or L series A-199
Data logging A-227
Debugging A-198
Diagnostic information A-157
Fuse blown module A-210
Instruction related registers A-194
I/O module verification A-210
Memory cards A-185
Overview A-156
Process control instruction A-211
Redundant CPU information A-198
Redundant power supply module A-218
Redundant system information (host CPU) A-212
Redundant system information (other CPU) A-214
Redundant system (tracking information) A-217
Remote password count A-198
Scan information A-182
SM1592 13-61
System clocks/counters A-181
System information A-170
Special relay
SM1592 11-3
SM237 13-3
SM321 13-69
SM701 7-219
SM775 7-171
Special relays
Built-in Ethernet port A-139
Built-in I/O function A-149
Conversion from A series to MELSEC System Q or L series A-132
Data logging A-154
Debugging A-131
Diagnostic information A-114
Drive information A-125
Instruction related special relays A-128
I/O refresh A-124
Overview A-113
Process control instruction A-140
Redundant power supply module A-148
Redundant system information (host CPU) A-140
Redundant system information (other CPU) A-144
Redundant system (tracking information) A-145
Scan information A-124
System clocks A-122
System information A-117
SP.CONTSW 11-2
SP.DEVST 7-612
SP.FREAD 7-599
SP.FWRITE 7-588

SQRD, SQRDP 7-408
SQR, SQRP 7-405
Square root calculation
from 4-digit or 8-digit BCD data 7-432
of floating point values (double precision) 7-409
of floating point values (single precision) 7-406
SRND, SRNDP 7-429
Standard device registers (Z) 3-43
Start instruction for the CHK instruction 7-233
STMR 6-214
STOP 5-66
Storing parts of character strings 7-324
STRDEL, STRDELP 7-336
STRINS, STRINSP 7-333
Structured program instructions 7-133
Structured program instructions (overview) . . . 2-36
STR, STRP 7-283
Subroutine program call
CALL 7-141
in a program file (ECALL) 7-154
XCALL 7-165
Subset processing 3-42
Subtracting
clock data 7-499
expansion clock data 7-529
SUM, SUMP 7-87
SWAP, SWAPP 6-182
System clocks
Special relays A-122
System clocks/counters
Special registers A-181
System information
Special registers A-170
Special relays A-117
System switching instruction for
a redundant system 2-61
System switching instruction (redundant PLC) 11-3
S.DATERD, SP.DATERP 7-520
S.DATE-, SP.DATE- 7-528
S.DATE+, SP.DATE+ 7-524
S.DEVLD, SP.DEVLD 7-615
S.RTREAD, SP.RTREAD 8-8
S.RTWRITE, SP.RTWRITE 8-10
S.TO, SP.TO 9-4
S.ZCOM, SP.ZCOM 8-2

T

TAND, TANDP 7-366
Tangent calculation
from BCD data 7-442
from floating point values
(double precision) 7-367
from floating point values
(single precision) 7-364
TAN, TANP 7-363
Terminating a FOR/NEXT loop 7-138
Termination Instructions 5-60
Test of single bits in 16- / 32-bit data words . . 7-72
TEST, TESTP 7-71
TIMCHK 7-552
Time check instruction 7-553
TM=, TM<>, TM>, TM<=, TM<, TM>= 7-514
TO, TOP 7-212, 9-8
Transfer
BIN block data (16-bit) 6-171
BIN block data (32-bit) 6-174
TTMR 6-212

U

UDCNT1 6-206
UDCNT2 6-209
UINI 12-72
UMSG 7-634
UNIRD, UNIRDP 7-574
Uniting
data in byte units 7-113
data in random bit groupings 7-108
16-bit data 7-104
UNI, UNIP 7-103
Unloading and loading of a program 7-625
Upper and lower limit control 7-453
User frames
deletion 12-13
reading of 12-8
registration 12-13
transmission 12-20
User message 7-635

V

VAL, VALP 7-290
Variables in instruction descriptions 4-5
Variables (use in an instruction) 3-1
Verification
of device data 3-51
of the device range 3-45

W

WAND, WANDP 7-4
WDT, WDTP 7-547
Word devices
  specification of bits 3-11
  usage in an instruction 3-13
WORD, WORDP 6-125
WOR, WORP 7-13
Writing
  clock data 7-491
  data to a designated file 7-591
  data to Standard ROM 7-613
  devices to another CPU 10-15
  routing information 8-11
  to an index table 7-185
Writing data
  to a data table 7-189
  to the buffer memory of a special function module 7-213
Writing of data
  to a PROFIBUS interface module 12-31
  to automatic updating buffer 12-126
  to fixed buffer (ETHERNET) 12-44
  to intelligent device station (CC-Link) . . 12-122
  to station on CC-Link 12-107
WSUM, WSUMP 7-126
WTOB, WTOBP 7-111
WXNR, WXNRP 7-29
WXOR, WXORP 7-21

X

XCALL 7-164
XCH, XCHP 6-176
x, xP, /, /P 6-45

Z

Zone control 7-453
ZONE, ZONEP 7-462
ZPOP, ZPOPP 7-570
ZPUSH, ZPUSHP 7-570
ZRRDB, ZRRDBP 7-554
ZRWRB, ZRWRBP 7-558

Symbols

+, +P, -, -P 6-38
=, < >, >, < =, <, > = ... 6-6
$ =, $ < >, $ >, $ < =, $ <, $ > = ... 6-20
$MOV, $MOVP 6-158
$+, $+P 6-91

Numbers

16-bit data comparison 6-7
32-bit data comparison 6-10
7-segment data 7-98

MITSUBISHI ELECTRIC INDUSTRIAL AUTOMATION

Gothaer Straße D-40880 Ratingen	Telefon: 02102 486-0 Hotline: 01805 000-7650	Fax: 02102 486-7170 melfa-mail@meg.mee.com	www.mitsubishi-automation.de www.mitsubishi-automation.com