MITSUBISHI ELECTRIC

MELSEC FX series

Programmable Controllers

Structured Programming Manual [Basic & Applied Instruction]

FXCPU

01 07 2009
JY997D34701
Version B

MITSUBISHI ELECTRIC INDUSTRIAL AUTOMATION

FXCPU Structured Programming Manual

(Basic & Applied Instruction)

Manual number	JY997D34701
Manual revision	B
Date	7/2009

Foreword

This manual contains text, diagrams and explanations which will guide the reader through the safe and correct installation, use, and operation of the FX Series function for structured programs. It should be read and understood before attempting to install or use the unit.

Store this manual in a safe place so that you can take it out and read it whenever necessary. Always forward it to the end user.

Outline Precautions

- This manual provides information for the use of the FX Series Programmable Controllers. The manual has been written to be used by trained and competent personnel. The definition of such a person or persons is as follows;

1) Any engineer who is responsible for the planning, design and construction of automatic equipment using the product associated with this manual should be of a competent nature, trained and qualified to the local and national standards required to fulfill that role. These engineers should be fully aware of all aspects of safety with aspects regarding to automated equipment.
2) Any commissioning or maintenance engineer must be of a competent nature, trained and qualified to the local and national standards required to fulfill the job. These engineers should also be trained in the use and maintenance of the completed product. This includes being familiar with all associated manuals and documentation for the product. All maintenance should be carried out in accordance with established safety practices.
3) All operators of the completed equipment should be trained to use that product in a safe and coordinated manner in compliance with established safety practices. The operators should also be familiar with documentation that is connected with the actual operation of the completed equipment.

Note: the term 'completed equipment' refers to a third party constructed device that contains or uses the product associated with this manual.

- This product has been manufactured as a general-purpose part for general industries, and has not been designed or manufactured to be incorporated in a device or system used in purposes related to human life.
- Before using the product for special purposes such as nuclear power, electric power, aerospace, medicine or passenger movement vehicles, consult with Mitsubishi Electric.
- This product has been manufactured under strict quality control. However when installing the product where major accidents or losses could occur if the product fails, install appropriate backup or failsafe functions into the system.
- When combining this product with other products, please confirm the standards and codes of regulation to which the user should follow. Moreover, please confirm the compatibility of this product with the system, machines, and apparatuses to be used.
- If there is doubt at any stage during installation of the product, always consult a professional electrical engineer who is qualified and trained in the local and national standards. If there is doubt about the operation or use, please consult the nearest Mitsubishi Electric distributor.
- Since the examples within this manual, technical bulletin, catalog, etc. are used as reference; please use it after confirming the function and safety of the equipment and system. Mitsubishi Electric will not accept responsibility for actual use of the product based on these illustrative examples.
- The content, specification etc. of this manual may be changed for improvement without notice.
- The information in this manual has been carefully checked and is believed to be accurate; however, if you notice any doubtful point, error, etc., please contact the nearest Mitsubishi Electric distributor.

Registration

- Microsoft® and Windows® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
- CompactFlash is a trademark of SanDisk Corporation in the United States and other countries.
- The company name and the product name to be described in this manual are the registered trademarks or trademarks of each company.

Table of Contents

Positioning of This Manual ... 9
Related Manuals ... 12
Generic Names and Abbreviations Used in Manuals ... 15

1. Outline 16

1.1 Outline of Structured Programs and Programming languages ... 16
1.1.1 Outline of Structured Programs ... 16
1.1.2 Programming languages ... 17
1.2 PLC Series and Programming Software Version ... 17
1.3 Cautions on Creation of Fundamental Programs ... 18
1.3.1 I/O PROCESSING AND RESPONSE DELAY ... 18
1.3.2 Double output (double coil) operation and countermeasures ... 19
1.3.3 Circuits which cannot be created by structured ladder programs and countermeasures ... 20
1.3.4 Handling of general flags ... 20
1.3.5 Handling of operation error flag ... 23
1.3.6 Handling of function extension flag ... 24
1.3.7 Limitation in number of sequence instructions and number of simultaneous instances of instructions ... 24

2. Instruction List 27

2.1 Basic Instructions ... 27
2.2 Step Ladder Instructions ... 28
2.3 Applied Instructions ... 28

3. Configuration of Instruction 47

3.1 Expression and Operation Form of Sequence Instructions ... 47
3.2 Labels ... 49
3.3 Devices and Addresses ... 52
3.4 EN and ENO ... 53

4. How to Read Explanation of Instructions 54

5. Basic Instruction 56

5.1 LD, LDI, AND, ANI, OR, OR ... 56
5.2 LDP, LDF, ANDP, ANDF, ORP, ORF ... 61
5.3 OUT (Excluding timers and counters) ... 67
5.4 Operating Timer ... 70
5.4.1 OUT_T ... 70
5.5 Operating Counters ... 74
5.5.1 OUT_C, OUT_C_32 ... 74
5.6 AND(...), OR(...) ... 77
5.7 MPS, MRD, MPP ... 79
5.8 INV ... 84
5.9 MEP, MEF ... 86
5.10 SET, RST ... 88
5.11 PLS, PLF ... 92
5.12 MC, MCR ... 95
5.13 END ... 99

5.14 NOP (for simple project only) ... 99

6. Step Ladder Instructions 100

6.1 Step Ladder ... 100
6.1.1 Outline ... 100
6.1.2 Function and operation explanation ... 100
6.1.3 Program examples ... 108
6.2 STL ... 109
6.3 RET ... 110

7. Applied Instructions 112

7.1 Program Flow ... 112
7.1.1 CJ ... 112
7.1.2 CALL ... 120
7.1.3 SRET ... 126
7.1.4 IRET ... 127
7.1.5 DI ... 130
7.1.6 EI ... 131
7.1.7 FEND ... 133
7.1.8 WDT ... 135
7.1.9 FOR ... 138
7.1.10 NEXT ... 139
7.2 Move and Compare ... 142
7.2.1 CMP ... 142
7.2.2 ZCP ... 146
7.2.3 MOV ... 149
7.2.4 SMOV ... 154
7.2.5 CML ... 157
7.2.6 BMOV ... 160
7.2.7 FMOV ... 165
7.2.8 XCH ... 168
7.2.9 BCD ... 170
7.2.10 BIN ... 174
7.3 Arithmetic and Logical Operation ... 178
7.3.1 ADD ... 178
7.3.2 SUB ... 181
7.3.3 MUL ... 185
7.3.4 DIV ... 189
7.3.5 INC ... 192
7.3.6 DEC ... 194
7.3.7 WAND ... 196
7.3.8 WOR ... 198
7.3.9 WXOR ... 200
7.3.10 NEG ... 203
7.4 Rotation and Shift Operation ... 207
7.4.1 ROR ... 207
7.4.2 ROL ... 210
7.4.3 RCR ... 213
7.4.4 RCL ... 216
7.4.5 SFTR ... 219
7.4.6 SFTL ... 221
7.4.7 WSFR ... 224
7.4.8 WSFL ... 227
7.4.9 SFWR ... 230
7.4.10 SFRD ... 233
7.5 Data Operation ... 235
7.5.1 ZRST ... 235
7.5.2 DECO ... 239
7.5.3 ENCO ... 243

7.5.4 SUM ... 246
7.5.5 BON ... 249
7.5.6 MEAN ... 252
7.5.7 ANS ... 254
7.5.8 ANR ... 256
7.5.9 SQR ... 258
7.5.10 FLT ... 260
7.6 High Speed Processing ... 264
7.6.1 REF ... 264
7.6.2 REFF ... 268
7.6.3 MTR ... 272
7.6.4 DHSCS ... 276
7.6.5 DHSCR ... 284
7.6.6 DHSZ ... 288
7.6.7 SPD ... 303
7.6.8 PLSY ... 307
7.6.9 PWM ... 314
7.6.10 PLSR ... 317
7.7 Handy Instruction ... 322
7.7.1 IST ... 322
7.7.2 SER ... 334
7.7.3 ABSD ... 338
7.7.4 INCD ... 342
7.7.5 TTMR ... 345
7.7.6 STMR ... 348
7.7.7 ALT ... 351
7.7.8 RAMP ... 354
7.7.9 ROTC ... 357
7.7.10 SORT ... 360
7.8 External FX I/O Device ... 363
7.8.1 TKY ... 363
7.8.2 HKY ... 367
7.8.3 DSW ... 372
7.8.4 SEGD ... 376
7.8.5 SEGL ... 378
7.8.6 ARWS ... 383
7.8.7 ASC ... 388
7.8.8 PR ... 390
7.8.9 FROM ... 393
7.8.10 TO ... 399
7.9 External Device (optional device) ... 402
7.9.1 RS ... 402
7.9.2 PRUN ... 405
7.9.3 ASCI ... 407
7.9.4 HEX ... 411
7.9.5 CCD ... 415
7.9.6 VRRD ... 419
7.9.7 VRSC ... 421
7.9.8 RS2 ... 423
7.9.9 PID ... 426
7.10 External Device ... 430
7.10.1 MNET ... 430
7.10.2 ANRD ... 432
7.10.3 ANWR ... 434
7.10.4 RMST ... 435
7.10.5 RMWR ... 436
7.10.6 RMRD ... 438
7.10.7 RMMN ... 440
7.10.8 BLK ... 441
7.10.9 MCDE ... 443
7.11 Data Transfer 2 ... 444
7.11.1 ZPUSH ... 444
7.11.2 ZPOP ... 447

7.12 Floating Point 449
7.12.1 DECMP 449
7.12.2 DEZCP 451
7.12.3 DEMOV 453
7.12.4 DESTR 455
7.12.5 DEVAL 462
7.12.6 DEBCD 468
7.12.7 DEBIN 470
7.12.8 DEADD 473
7.12.9 DESUB 475
7.12.10 DEMUL 477
7.12.11 DEDIV 479
7.12.12 DEXP 481
7.12.13 DLOGE 483
7.12.14 DLOG10 485
7.12.15 DESQR 487
7.12.16 DENEG 489
7.12.17 INT 491
7.12.18 DSIN 493
7.12.19 DCOS 495
7.12.20 DTAN 497
7.12.21 DASIN 499
7.12.22 DACOS 502
7.12.23 DATAN 505
7.12.24 DRAD 508
7.12.25 DDEG 511
7.13 Data Operation 2 513
7.13.1 WSUM 513
7.13.2 WTOB 516
7.13.3 BTOW 519
7.13.4 UNI 522
7.13.5 DIS 525
7.13.6 SWAP 527
7.13.7 SORT2 529
7.14 Positioning Control 534
7.14.1 DSZR 534
7.14.2 DVIT 536
7.14.3 DTBL 539
7.14.4 DABS 541
7.14.5 ZRN 543
7.14.6 PLSV 547
7.14.7 DRVI 551
7.14.8 DRVA 554
7.15 Real Time Clock Control 557
7.15.1 TCMP 557
7.15.2 TZCP 560
7.15.3 TADD 563
7.15.4 TSUB 565
7.15.5 HTOS 567
7.15.6 STOH 570
7.15.7 TRD 573
7.15.8 TWR 575
7.15.9 HOUR 579
7.16 External Device 582
7.16.1 GRY 582
7.16.2 GBIN 584
7.16.3 RD3A 586
7.16.4 WR3A 588
7.17 Extension Function 590
7.17.1 EXTR_IN 590
7.17.2 EXTR_OUT 593
7.18 Others 597
7.18.1 COMRD 597
7.18.2 RND 600

7.18.3 DUTY ... 602
7.18.4 CRC ... 605
7.18.5 DHCMOV ... 609
7.19 Block Data Operation ... 614
7.19.1 BK+ ... 614
7.19.2 BK- ... 618
7.19.3 BKCMP=, BKCMP>, BKCMP<, BKCMP<>, BKCMP<=, BKCMP>= ... 622
7.20 Character String Control ... 629
7.20.1 STR ... 629
7.20.2 VAL ... 636
7.20.3 $+ ... 642
7.20.4 LEN ... 645
7.20.5 RIGHT ... 648
7.20.6 LEFT ... 651
7.20.7 MIDR ... 654
7.20.8 MIDW ... 658
7.20.9 INSTR ... 662
7.20.10 $MOV ... 665
7.21 Data Operation 3 ... 668
7.21.1 FDEL ... 668
7.21.2 FINS ... 671
7.21.3 POP ... 674
7.21.4 SFR ... 678
7.21.5 SFL ... 681
7.22 Data Comparison ... 684
7.22.1 LD=, LD>, LD<, LD<>, LD<=, LD>= ... 684
7.22.2 AND=, AND>, AND<, AND<>, AND<=, AND>= ... 688
7.22.3 OR=, OR>, OR<, OR<>, OR<=, OR>= ... 692
7.23 Data Table Operation ... 695
7.23.1 LIMIT ... 695
7.23.2 BAND ... 700
7.23.3 ZONE ... 706
7.23.4 SCL ... 712
7.23.5 DABIN ... 717
7.23.6 BINDA ... 721
7.23.7 SCL2 ... 726
7.24 External Device Communication (Inverter Communication) ... 732
7.24.1 IVCK ... 732
7.24.2 IVDR ... 735
7.24.3 IVRD ... 738
7.24.4 IVWR ... 740
7.24.5 IVBWR ... 742
7.25 Data Transfer 3 ... 745
7.25.1 RBFM ... 745
7.25.2 WBFM ... 751
7.26 High Speed Processing 2 ... 753
7.26.1 DHSCT ... 753
7.27 Extension File Register Control ... 759
7.27.1 LOADR ... 759
7.27.2 SAVER ... 763
7.27.3 INITR ... 772
7.27.4 LOGR ... 776
7.27.5 RWER ... 780
7.27.6 INITER ... 785
7.28 FX3U-CF-ADP ... 789
7.28.1 FLCRT ... 789
7.28.2 FLDEL ... 793
7.28.3 FLWR ... 795
7.28.4 FLRD ... 798
7.28.5 FLCMD ... 800
7.28.6 FLSTRD ... 802

8. Interrupt Function and Pulse Catch Function 805

8.1 Outline ... 805
8.2 Common items ... 806
8.2.1 Interrupt function ... 806
8.2.2 How to disable interrupt function and pulse catch function ... 807
8.2.3 Related items ... 808
8.2.4 Cautions on use (common) ... 809
8.3 Input Interrupt (Interrupt Triggered by External Signal) [Without Delay Function] ... 811
8.3.1 Input Interrupt (Interrupt Triggered by External Signal) [Without Delay Function] ... 811
8.3.2 Examples of practical programs (programs to measure short pulse width) ... 816
8.4 Input Interrupt (Interrupt by External Signal) [With Delay function] ... 818
8.5 Timer Interrupt (Interrupt in Constant Cycle) ... 819
8.5.1 Timer Interrupt (Interrupt in Constant Cycle) ... 819
8.5.2 Example of practical program (timer interrupt program using instruction) ... 820
8.6 Counter Interrupt - Interrupt Triggered by Counting Up of High Speed Counter ... 822
8.7 Pulse Catch Function[M8170 to M8177] ... 823
8.8 Pulse width/Pulse period measurement function [M8075 to M8083, D8074 to D8097] ... 825

Appendix A: Relationships between devices and addresses 830

Warranty ... 832
Revised History ... 833

Positioning of This Manual

This manual explains sequence instructions for structured programs provided by GX Works2. Refer to other manuals for devices, parameters and application functions.
Refer to each corresponding manual for analog, communication, positioning control and special units and blocks.

1. When using FX3U/FX3UC/FX3G PLCs

Q/FX Structured

QCPU/FXCPU Structured Programming Manual (Fundamentals) (Additional Manual)

This manual explains programming methods, specifications, functions, etc. required to create structured programs.

FX Structured

FXCPU Structured Programming Manual (Device & Common) (Additional Manual)

This manual explains devices and parameters for structured programs provided by GX Works2.

(This manual)

FX Structured

FXCPU Structured Programming Manual (Basic & Applied Instruction) (Additional Manual)

This manual explains sequence instructions for structured programs provided by GX Works2.

FX Structured

FXCPU Structured Programming Manual (Application Functions) (Additional Manual)

This manual explains application functions for structured programs provided by GX Works2.

FX3U FX3UC FX3G

FX3G/FX3U/FX3UC User's Manual- Analog Control Edition (Additional Manual)

This manual explains details of analog special function blocks and analog special adapters for FX3U/FX3UC/FX3G PLCs and PID instruction.
Explanation of instructions and instructions used in program examples are expressed for GX Developer.

FX

FX Series User's Manual -Data Communication Edition (Additional Manual)

This manual explains details of simple N:N link, parallel link, computer link, no-protocol communication (RS and RS2 instructions), programming communication and inverter communication for FX PLCs.
Explanation of instructions and instructions used in program examples are expressed for GX Developer.

FX3U FX3UC FX3G

FX3G/FX3U/FX3UC Series User's Manual -Positioning Edition (Additional Manual)

This manual explains details of wiring, instructions and operations of the positioning function built in FX3U/FX3UC/FX3G PLC main units.
Explanation of instructions and instructions used in program examples are expressed for GX Developer.

Special unit/block

Individual manuals (Manual supplied with product or additional Manual*1)

This manual explains details of each special unit/block.
Explanation of instructions and instructions used in program examples are expressed for GX Developer.

*1. Detailed explanation may be provided by a separate manual in some products.

2. When using FX1S/FX1N/FXU/FX1NC/FX2NC PLCs

Q/FX
Structured

QCPU/FXCPU Structured Programming Manual (Fundamentals) (Additional Manual)

This manual explains programming methods, specifications, functions, etc. required to create structured programs.

FX
Structured

FXCPU Structured Programming Manual (Device & Common) (Additional Manual)

This manual explains devices and parameters for structured programs provided by GX Works2.

(This manual)

FX
Structured

FXCPU Structured Programming Manual (Basic & Applied Instruction) (Additional Manual)

This manual explains sequence instructions for structured programs provided by GX Works2.

FX
Structured

FXCPU Structured Programming Manual (Application Functions) (Additional Manual)

This manual explains application functions for structured programs provided by GX Works2.

FX

FX Series User's Manual -Data Communication Edition (Additional Manual)

This manual explains details of simple N:N link, parallel link, computer link, no-protocol communication (RS instruction), programming communication and inverter communication for FX PLCs.
Explanation of instructions and instructions used in program examples are expressed for GX Developer and FX-PCS/WIN.

Special
unit/block

Individual manuals (Manual supplied with product or additional Manual*1)

This manual explains details of each special unit/block.
Explanation of instructions and instructions used in program examples are expressed for GX Developer and FX-PCS/WIN.

*1. Detailed explanation may be provided by a separate manual in some products.

3. When using FX0/FX0S/FX0N/FXU/FX2C PLCs

Q/FX Structured

QCPU/FXCPU Structured Programming Manual (Fundamentals) (Additional Manual)

This manual explains programming methods, specifications, functions, etc. required to create structured programs.

FX Structured

FXCPU Structured Programming Manual (Device & Common) (Additional Manual)

This manual explains devices and parameters for structured programs provided by GX Works2.

(This manual)

FX Structured

FXCPU Structured Programming Manual (Basic & Applied Instruction) (Additional Manual)

This manual explains sequence instructions for structured programs provided by GX Works2.

FX Structured

FXCPU Structured Programming Manual (Application Functions) (Additional Manual)

This manual explains application functions for structured programs provided by GX Works2.

FX

FX Series User's Manual -Data Communication Edition (Additional Manual)

This manual explains details of parallel link, computer link, no-protocol communication (RS instruction) and programming communication for FX PLCs.
Explanation of instructions and instructions used in program examples are expressed for GX Developer and FX-PCS/WIN.

Special unit/block

Individual manuals (Manual supplied with product or additional Manual *1)

This manual explains details of each special unit/block.
Explanation of instructions and instructions used in program examples are expressed for GX Developer and FX-PCS/WIN.

*1. Detailed explanation may be provided by a separate manual in some products.

Related Manuals

This manual explains devices and parameters for structured programs provided by GX Works2.
Refer to other manuals for sequence instructions and applied functions.
This chapter introduces only reference manuals for this manual and manuals which describe the hardware information of PLC main units.
Manuals not introduced here may be required in some applications.
Refer to the manual of the used PLC main unit and manuals supplied together with used products.
Contact the distributor for acquiring required manuals.

Common among FX PLCs [structured]

Manual name	Manual number	Supplied with product or Additional Manual	Contents	Model name code
QCPU/FXCPU Structured Programming Manual (Fundamentals)	SH-080782	Additional Manual	Programming methods, specifications, functions, etc. required to create structured programs	13JW06
FXCPU Structured Programming Manual (Device & Common)	JY997D26001	Additional Manual	Devices, parameters, etc. provided in structured projects of GX Works2	09R920
FXCPU Structured Programming Manual (Basic & Applied Instruction)	JY997D34701	Additional Manual	Sequence instructions provided in structured projects of GX Works2	09R921
FXCPU Structured Programming Manual (Application Functions)	JY997D34801	Additional Manual	Application functions provided in structured projects of GX Works2	09R922

FX3U/FX3UC/FX3G PLCs

Manual name	Manual number	Supplied with product or Additional Manual	Contents	Model name code
PLC main unit				
FX3U Series Hardware Manual	JY997D18801	Supplied with product	I/O specifications, wiring and installation of the PLC main unit FX3U extracted from the FX3U Series User's Manual - Hardware Edition. For detailed explanation, refer to the FX3U Series User's Manual - Hardware Edition.	-
FX3U Series User's Manual- Hardware Edition	JY997D16501	Additional Manual	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX3U PLC main unit.	09R516
FX3UC (D, DSS) Series Hardware Manual	JY997D28601	Supplied with product	I/O specifications, wiring and installation of the PLC main unit FX3UC (D, DSS) extracted from the FX3UC Series User's Manual - Hardware Edition. For detailed explanation, refer to the FX3UC Series User's Manual - Hardware Edition.	-
FX3UC-32MT-LT-2 Hardware Manual	JY997D31601	Supplied with product	I/O specifications, wiring and installation of the PLC main unit FX3UC-32MT-LT-2 extracted from the FX3UC Series User's Manual - Hardware Edition. For detailed explanation, refer to the FX3UC Series User's Manual - Hardware Edition.	-
FX3UC Series User's Manual - Hardware Edition	JY997D28701	Additional Manual	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX3UC PLC main unit.	09R519
FX3G Series Hardware Manual	JY997D33401	Supplied with product	I/O specifications, wiring and installation of the PLC main unit FX3G extracted from the FX3G Series User's Manual - Hardware Edition. For detailed explanation, refer to the FX3G Series User's Manual - Hardware Edition.	-
FX3G Series User's Manual- Hardware Edition	JY997D31301	Additional Manual	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX3G PLC main unit.	09R521

Manual name	Manual number	Supplied with product or Additional Manual	Contents	Model name code
Programming				
FX3G/FX3U/FX3UC User's Manual-Analog Control Edition	JY997D16701	Additional Manual	Detaileds about the analog special function block (FX3U-4AD, FX3U-4DA, FX3UC-4AD) and analog special adapter (FX3U-****-ADP).	09R619
FX Series User's Manual -Data Communication Edition	JY997D16901	Additional Manual	Details about simple N : N link, parallel link, computer link and no-protocol communication (RS instruction and FX2N-232IF).	09R715
FX3G/FX3U/FX3UC Series User's Manual -Positioning Edition	JY997D16801	Additional Manual	Details about the positioning function built in the FX3G/FX3U/FX3UC Series.	09R620
FX3U-CF-ADP User's Manual	JY997D35401	Additional Manual	Describes details of the FX3U-CF-ADP CF card special adapter.	09R720

FX1S/FX1N/FX2N/FX1NC/FX2NC PLCs

Manual name	Manual number	Supplied with product or Additional Manual	Contents	Model name code
PLC main unit				
FX1S HARDWARE MANUAL	JY992D83901	Additional Manual	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX1S PLC main unit.	-
FX1N HARDWARE MANUAL	JY992D89301	Additional Manual	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX1N PLC main unit.	-
FX2N HARDWARE MANUAL	JY992D66301	Additional Manual	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX2N PLC main unit.	09R508
FX1NC HARDWARE MANUAL	JY992D92101	Additional Manual	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX1NC PLC main unit. (Japanese only)	09R505
FX2NC HARDWARE MANUAL	JY992D76401	Additional Manual	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX2NC PLC main unit.	09R509
Programming				
FX Series User's Manual -Data Communication Edition	JY997D16901	Additional Manual	Details about simple N : N link, parallel link, computer link and no-protocol communication (RS instruction and FX2N-232IF).	09R715

FX0/FX0S/FX0N/FXU/FX2C PLCs [whose production is finished]

Manual name	Manual number	Supplied with product or Additional Manual	Contents	Model name code
PLC main unit				
FX0/FX0N HARDWARE MANUAL	JY992D47501	Supplied with product	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX0/FX0N PLC main unit.	-
FX0S HARDWARE MANUAL	JY992D55301	Supplied with product	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FX0S PLC main unit.	-
FX/FX2C HARDWARE MANUAL	JY992D47401	Supplied with product	Details about the hardware including I/O specifications, wiring, installation and maintenance of the FXU/FX2C PLC main unit.	-
Programming				
FX Series User's Manual -Data Communication Edition	JY997D16901	Additional Manual	Details about simple N : N link, parallel link, computer link and no-protocol communication (RS instruction and FX2N-232IF).	09R715

Manuals of models whose production is finished

Production is finished for FX0/FX0S/FX0N/FXU/FX2C PLCs.

Generic Names and Abbreviations Used in Manuals

Abbreviation/generic name	Name
PLCs	
FX3U Series or FX3U PLC	Generic name of FX3U Series PLCs
FX3UC Series or FX3UC PLC	Generic name of FX3UC Series PLCs
FX3G Series or FX3G PLC	Generic name of FX3G Series PLCs
FX2N Series or FX2N PLC	Generic name of FX2N Series PLCs
FX2NC Series or FX2NC PLC	Generic name of FX2NC Series PLCs
FX1N Series or FX1N PLC	Generic name of FX1N Series PLCs
FX1NC Series or FX1NC PLC	Generic name of FX1NC Series PLCs These products can only used in Japan.
FX1S Series or FX1S PLC	Generic name of FX1S Series PLCs
FXU Series or FXU PLC	Generic name of FXU(FX,FX2) Series PLCs
FX2C Series or FX2C PLC	Generic name of FX2C Series PLCs
FX0N Series or FX0N PLC	Generic name of FX0N Series PLCs
FX0S Series or FX0S PLC	Generic name of FX0S Series PLCs
FX0 Series or FX0 PLC	Generic name of FX0 Series PLCs
Special adapters	
CF card special adapter	Generic name of CF card special adapters
CF-ADP	FX3U-CF-ADP
Programming language	
ST	Abbreviation of structured text language
Structured ladder	Abbreviation of ladder diagram language
Manuals	
Q/FX Structured Programming Manual (Fundamentals)	Abbreviation of QCPU/FXCPU Structured Programming Manual (Fundamentals)
FX Structured Programming Manual (Device & Common)	Abbreviation of FXCPU Structured Programming Manual (Device & Common)
FX Structured Programming Manual (Basic & Applied Instruction)	Abbreviation of FXCPU Structured Programming Manual (Basic & Applied Instruction)
FX Structured Programming Manual (Application Functions)	Abbreviation of FXCPU Structured Programming Manual (Application Functions)
COMMUNICATION CONTROL EDITION	Abbreviation of FX Series User's Manual-DATA COMMUNICATION CONTROL EDITION
ANALOG CONTROL EDITION	Abbreviation of FX3G/FX3U/FX3UC Series User's Manual-ANALOG CONTROL EDITION
POSITIONING CONTROL EDITION	Abbreviation of FX3G/FX3U/FX3UC Series User's Manual-POSITIONING CONTROL EDITION

1. Outline

This manual explains setting of sequence instructions for structured programs provided by GX Works2.
Refer to another manuals for device, parameter, and application functions for structured programs.
Refer to the following manual for label, data types and programming languages for structured programs.
→ **Q/FX Structured Programming Manual (Fundamentals)**

1.1 Outline of Structured Programs and Programming languages

1.1.1 Outline of Structured Programs

You can construct two or more programs (program blocks) into one program.
Because you can divide the entire machine processing into small sub processes and create a program for each sub process, you can create a program for a large system efficiently.

1. **Structured program**
Program structuring is a technique to divide the contents of control executed by the PLC CPU into hierarchical small units (blocks) of processing, and then construct a program. By using this technique, you can design a program while recognizing structuring of a sequence program.

Advantages of hierarchical program

- You can examine the outline of a program at first, and then design its details gradually.
- Program blocks located at the lowest level in the hierarchy are extremely simple and highly dependent.

Advantages of program consisting of program blocks

- Because the processing of each block is clear, the entire program is easy to understand.
- The entire program can be divided into several blocks that are created by several people.
- The program reusability is improved, and the development efficiency is improved accordingly.

2. **Improved reusability of programs**
You can save program blocks in a library. Program resources in the library can be shared, and often used again.

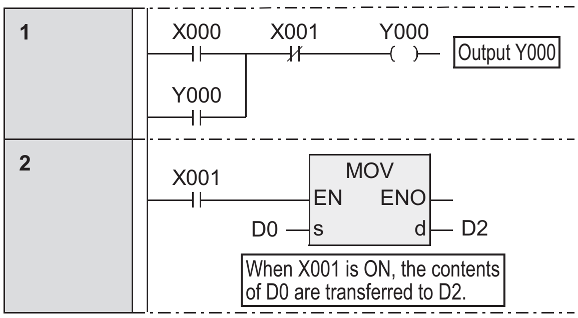
1.1.2 Programming languages

The following programming languages can be used in each program block.

Graphic languages

1. Structured ladder language

This graphic language is created based on the relay circuit design technology.
Any circuit always starts from the bus line located on the leftmost.
The structured ladder language consists of contacts, coils, functions and function blocks. These components are connected with vertical lines and horizontal lines.



Text language

1. ST ("Structured text language")

The ST language can describe control achieved by syntax using selective branches with conditional statements and repetition by repetitive statements in the same way as high-level languages such as C language. By using the ST language, you can create simple programs easy to understand.

```
Y000:=(X000 OR Y000) AND NOT X001;
IF X001 THEN
    D2:=D0;  (* When X001 is ON, the contents of D0 are transferred to D2.*)
END_IF;
IF X002 THEN
    D4:=D4+1;  (* When X002 is ON, the contents of D4 are added by "1". *)
ELSE
    D6:=D6+1;  (* When X002 is OFF, the contents of D6 are added by "1". *)
END_IF;
```

1.2 PLC Series and Programming Software Version

PLC series	Software package name (model name)	GX Works2 version
FX3U•FX3UC	GX Works2 (SW1DNC-GXW2-E)	Ver. 1.08J or later
FX3G		
FX2N•FX2NC		
FX1N•FX1NC		
FX1S		
FXU•FX2C		
FX0N		
FX0•FX0S		

1.3 Cautions on Creation of Fundamental Programs

This section explains cautions on programming.
Refer to the following manual for cautions on structured programs and programming languages:

→ **Q/FX Structured Programming Manual (Fundamentals)**

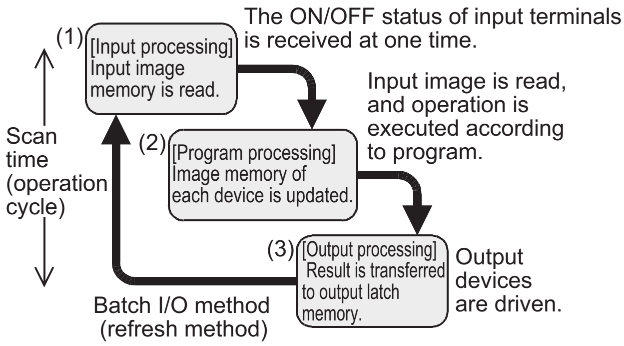
Refer to the following programming manual for detailed operations of and cautions on devices and parameters:

→ **FX Structured Programming Manual (Device & Common)**

1.3.1 I/O PROCESSING AND RESPONSE DELAY

1. Operation timing of I/O relays and resonse delay

FX PLCs execute the I/O processing by repeating the processing (1) to processing (3).
Accordingly, the control executed by PLCs contains not only the drive time of input filters and output devices but also the response delay caused by the operation cycle.



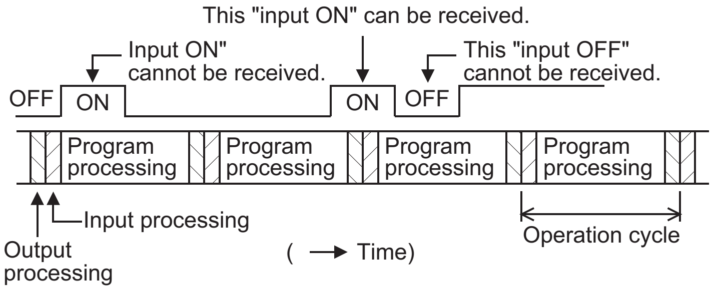
Acquiring the latest I/O information
For acquiring the latest input information or immediately outputting the operation result in the middle of the operation cycle shown above, the I/O refresh instruction "REF" is available.

2. Short input pulses cannot be received.

The ON duration and OFF duration of inputs in PLCs require longer time than "PLC cycle time + Input filter response delay."
When the response delay "10 ms" of the input filter is considered and the cycle time is supposed as "10 ms", the ON duration and OFF duration should be at least 20 ms respectively.
Accordingly, PLCs cannot handle input pulses at 25 Hz (1000 / (20+20) = 25) or more. However, the situation can be improved by PLC special functions and instructions.

Convenient functions for improvement
By using the following functions, PLCs can receive pulses shorter than the operation cycle.

- High speed counter function
- Input interrupt function
- Pulse catch function
- Input filter value adjustment function



1.3.2 Double output (double coil) operation and countermeasures

This subsection explains the double output (double coil) operation and countermeasures.

1. Operation of double output

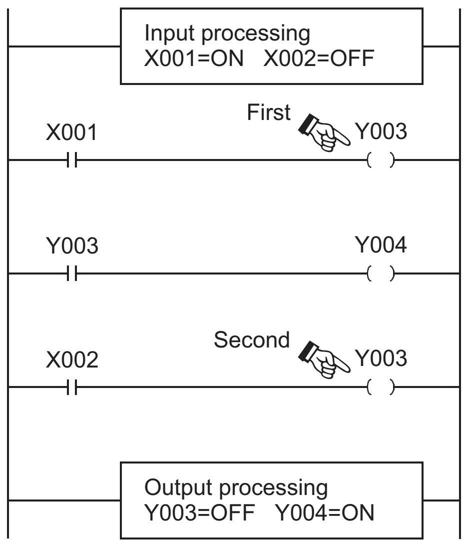
When a coil (output variable) is used twice (double coil) in another program block to be executed or in the same program block, the PLC gives priority to the last coil.

Suppose that the same coil Y003 is used in two positions as shown in the figure on the right.
For example, suppose that X001 is ON and X002 is OFF.

In the first coil Y003, the image memory turns ON and the output Y004 turns ON also because the input X001 is ON.

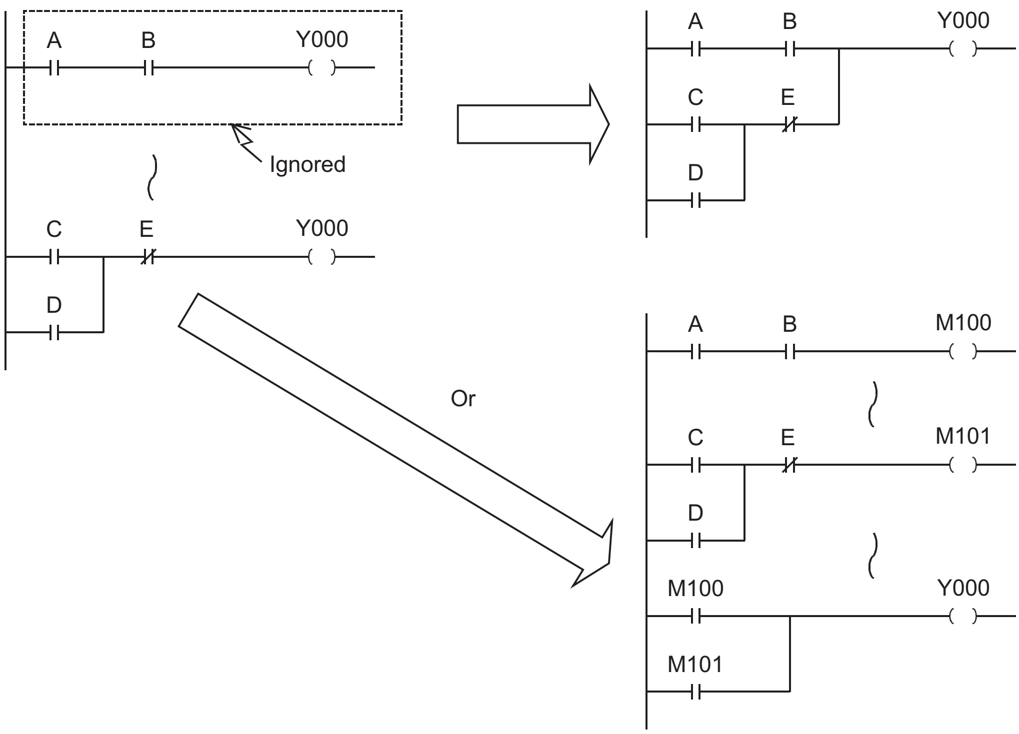
In the second coil Y003, however, the image memory is set to OFF because the input X002 is OFF.

Accordingly, the actual output to the outside is "Y003 = OFF, Y004 = ON".



2. Countermeasures against double output

Double output (double coil) does not cause an illegal input (program error), but the operation is disrupted as described above. Change the program as shown in the example below.

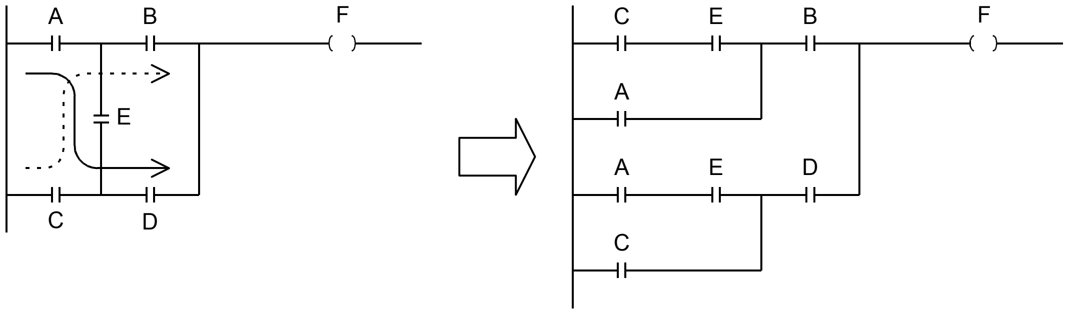


SET, RST or jump instruction can be used instead, or a same output coil can be programmed at each state by using step ladder instructions STL or RET.
When you use the step ladder instruction STL or RET, note that the PLC regards it as double coils if you program, inside the state, an output coil located outside the RET from another program block or the STL instruction.

1.3.3 Circuits which cannot be created by structured ladder programs and countermeasures

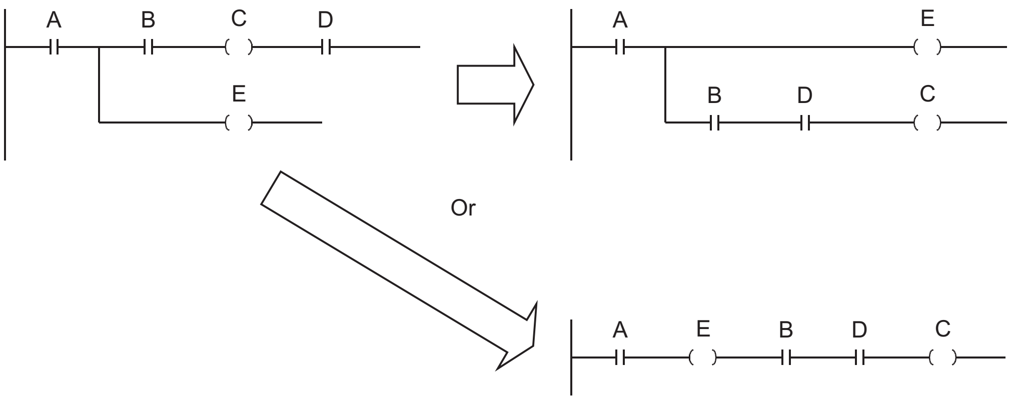
1. Bridge circuit

A circuit in which the current flows in both directions should be changed as shown in the figure on the right (so that a circuit without D and a circuit without B are connected in parallel).



2. Coil connection position

- You can program a contact on the right side of a coil. In this case, be sure to program a coil (including a function or a function block) at the end of the circuit.



1.3.4 Handling of general flags

In some types of sequence instructions, the following flags operate:

<Examples> M8020: Zero flag — M8021: Borrow flag
M8022: Carry flag — M8029: Instruction execution complete flag
M8090: Block comparison signal[*1] — M8328: Instruction non-execution flag[*1]
M8329: Instruction execution abnormal complete flag [*2]
M8304: Zero flag[*1] — M8306: Carry flag[*1]

*1. Supported only by FX3U and FX3UC PCLs.

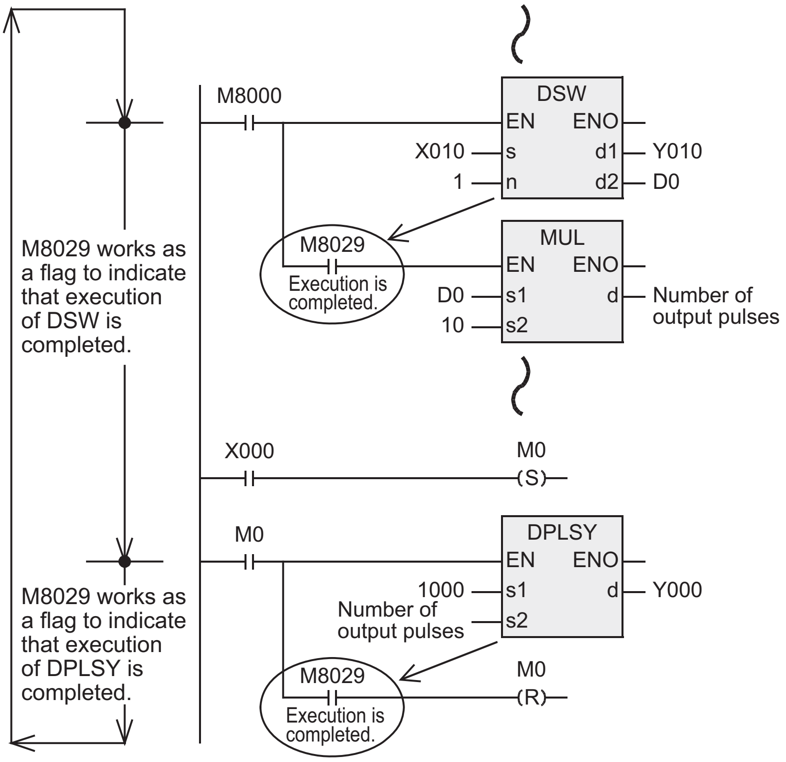
*2. Supported only by FX3U, FX3UC and FX3G PLCs.

Each of these flags turns ON or OFF every time the PLC executes a corresponding function. These flags do not turn ON or OFF when the PLC does not execute a corresponding function or when an error occurs.
Because these flags turn ON or OFF in many sequence instructions, the ON/OFF status of flags changes every time such instructions are executed.
Refer to the examples in the next page, and program a flag contact just under the target sequence instruction.

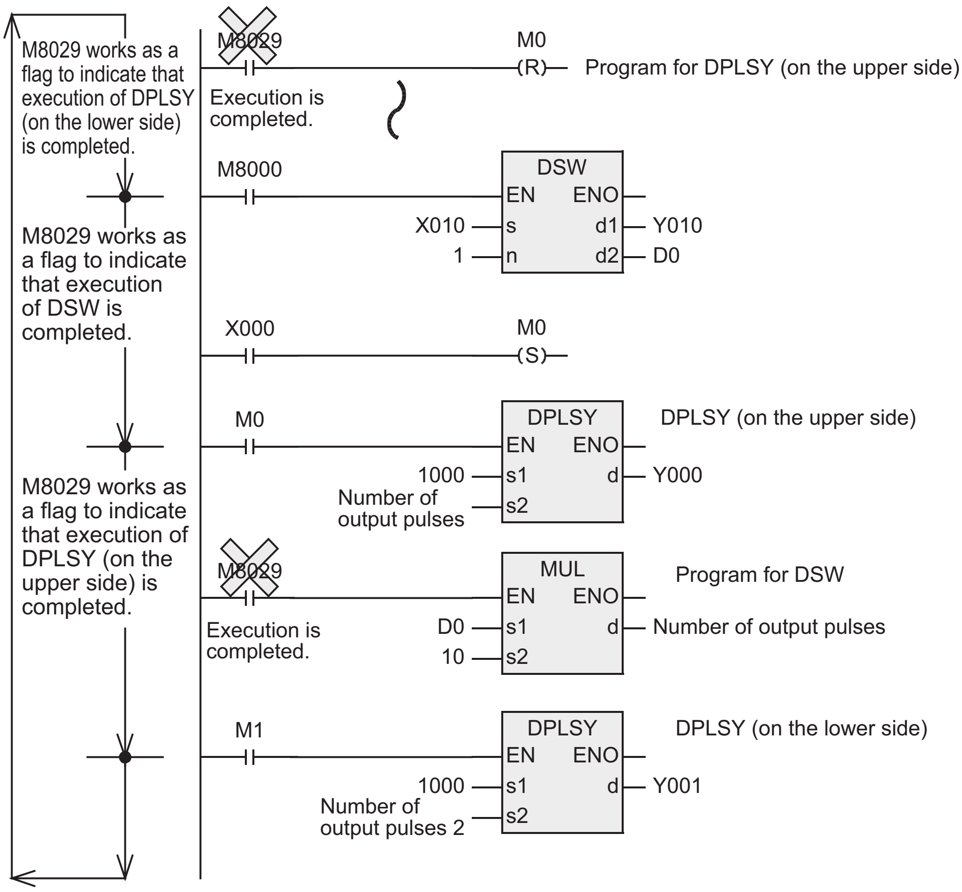
1. Program containing many flags (example of instruction execution complete flag M8029)

If you program the instruction execution completion flag M8029 for two or more sequence instructions which actuate the flag M8029, you cannot judge easily by which sequence instruction the flag M8029 is controlled. In addition, the flag M8029 does not turn ON or OFF correctly for each corresponding sequence instruction. Refer to the next page when you would like to use the flag M8029 in any position other than the position just under the corresponding sequence instruction.

Good example



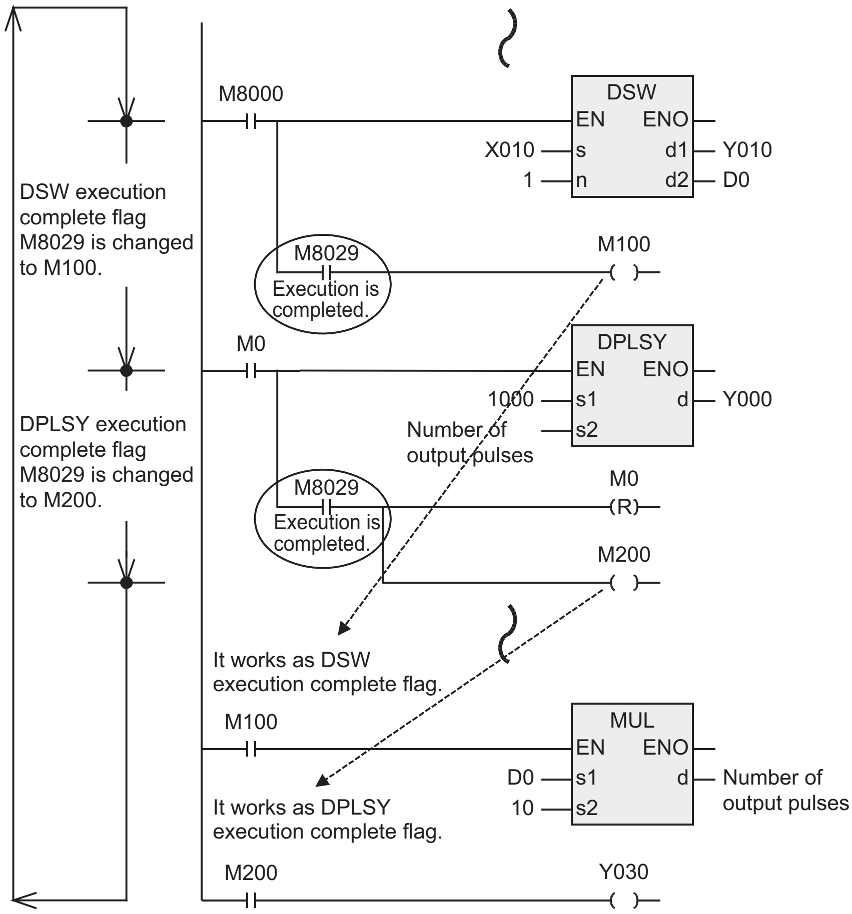
Bad example



2. Introduction of method for using flags in any positions other than directly under sequence instructions.

When two or more sequence instructions are programmed, general flags turn ON or OFF when each sequence instruction turns ON.

Accordingly, when using a flag in any position other than directly under a sequence instruction, set to ON or OFF another bit device (variable), and then use the contact (variable) of the device as the command contact.



1.3.5 Handling of operation error flag

When there is an error in the sequence instruction configuration, target device or target device number range and an error occurs while operation is executed, the following flag turns ON and the error information is store.

1. Operation error

Error flag	Device storing error code	Device storing error occurrence step	
		FX0/FX0S/FX0N/FXU/FX2C/FX1S /FX1N/FX2N/FX1NC/FX2NC/FX3G	FX3U/FX3UC
M8067	D8067	D8069*1	D8315, D8314

*1. When the error occurrence step is up to the 32767th step in FX3U and FX3UC PLCs, the error occurrence step can be checked in D8069 (16 bits).

- When an operation error has occurred, M8067 is set, D8067 stores the operation error code number, and the device storing error occurrence step (see the table above) stores the error occurrence step number.
- If another error occurs in another step, the stored data is updated in turn to the error code and step number of the new error. (These devices are set to OFF when errors are cleared.)
- When the PLC mode switches from STOP to RUN, these devices are cleared instantaneously, and then set to ON again if errors have not been cleared.

2. Operation error latch

Error flag	Device storing error code	Device storing error occurrence step	
		FX0/FX0S/FX0N/FXU/FX2C/FX1S /FX1N/FX2N/FX1NC/FX2NC/FX3G	FX3U/FX3UC
M8068	-	D8068*2	D8313, D8312

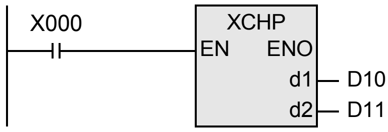
*2. When the error occurrence step is up to the 32767th step in FX3U and FX3UC PLCs, the error occurrence step can be checked in D8068 (16 bits).

- When an operation error has occurred, M8068 is set, and the device storing error occurrence step (see the table above) stores the error occurrence step number.
- Even if another error has occurred in another step, the stored data is not updated, and remains held until these devices are forcibly reset or until the power turns OFF.

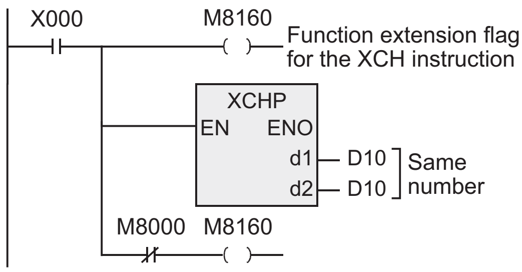
1.3.6 Handling of function extension flag

In some sequence instructions, the function can be extended by combining a specific special auxiliary relay determined for each sequence instruction. An example is explained below using a structured ladder program.

- When X000 turns ON, this instruction exchanges the contents of D10 and D11 with each other.



- If M8160 has been driven before the XCH function and the source and destination of the XCH instruction are specified to the same device, high-order 8 bits and low-order 8 bits are exchanged with each other inside the device.
- For returning this XCH to the normal XCH function, it is necessary to set M8160 to OFF.



When using an instruction requiring the function extension flag in an interrupt program, program DI function (for disabling interrupt) before driving the function extension flag, and program EI function (for enabling interrupt) after turning OFF the function extension flag.

1.3.7 Limitation in number of sequence instructions and number of simultaneous instances of instructions

Each sequence instruction has a limitation in the number of using the instruction and the number of simultaneous instances of instructions. The limitation, however, differs from one PLC to another.

Limitations in the number of instructions

Some instructions can be used only up to the specified number of times.
As for the instructions having a limited number of times of use and whose operands allow indexing, device numbers and numeric values in such instructions can be changed by index registers. By indexing, when driving multiple instances simultaneously is required, such instruction can be used as if they were used beyond the allowable number of times.

→ **FX Structured Programming Manual (Device & Common)**

Note that some PLCs do not provide some instructions.

→ **2. Instruction List**

FX3U, FX3UC and FX3G PLCs

Instruction name	Allowable number of times of use	Remarks
MTR	1	-
DHSCS	FX3U, FX3UC PLC is 32. FX3G PLC is 6.	FX3U, FX3UC PLC is the allowable number of times of sum total use of DHSCS, DHSCR, DHSZ, DHSCT. FX3G PLC is the allowable number of times of sum total use of DHSCS, DHSCR, DHSZ.
DHSCR		
DHSZ		
SPD	8 (1 instruction / 1 input or fewer)	Pay attention so that this instruction does not overlap the input numbers in interrupt input in DVIT instruction, DOG inputs in ZRN instruction, zero point signal in DSZR instruction, input interrupt numbers and high speed counter input numbers.
IST	1	-
SORT	1	FX3G PLC is not provided.
TKY	1	FX3G PLC is not provided.
HKY	1	FX3G PLC is not provided.
ARWS	1	FX3G PLC is not provided.
PR	2	FX3G PLC is not provided.
SORT2	2	FX3G PLC is not provided.
DUTY	5 (1 instruction / 1 output or fewer)	FX3G PLC is not provided.
DHSCT	1	FX3G PLC is not provided.

FX1S, FX1N, FX1NC, FX2N and FX2NC PLCs

Instruction name	Allowable number of times of use	
	FX1S, FX1N, FX1NC	FX2N, FX2NC
MTR	1	1
SPD	1	1
PWM	1	1
IST	1	1
ABSD	1	1
INCD	1	1
ROTC	FX1S, FX1N or FX1NC PLCs are not provided.	1
SORT	FX1S, FX1N or FX1NC PLCs are not provided.	1
TKY	FX1S, FX1N or FX1NC PLCs are not provided.	1
HKY	FX1S, FX1N or FX1NC PLCs are not provided.	1
DSW	No limit	2
SEGL	No limit	2
ARWS	FX1S, FX1N or FX1NC PLCs are not provided.	1
PR	FX1S, FX1N or FX1NC PLCs are not provided.	2

FX0, FX0S, FX0N, FXU and FX2C PLCs

Instruction name	Allowable number of times of use	Remarks
MTR	1	FX0, FX0S or FX0N PLCs are not provided.
PLSY	1	-
PWM	1	
IST	1	-
ABSD	1	FX0, FX0S or FX0N PLCs are not provided.
INCD	1	
ROTC	1	
SORT	1	
TKY	1	
HKY	1	
DSW	2	
SEGL	2	
ARWS	1	
PR	2	

Limitation in simultaneous instances of instructions

Some instructions can be programmed two or more times, but the number of simultaneous instances is limited. Even in instructions not shown below, if two or more instructions are driven at the same time for a same I/O number, it is regarded as double outputs. In some combinations of instructions, the operation may be disrupted, or the instructions cannot be executed.
For details, refer to the caution described in each instruction page.

- **FX3U, FX3UC and FX3G PLCs**
  PLSY, PWM, PLSR, DSZR, DVIT*1, ZRN, PLSV, DRVI, DRVA
  DHSCS, DHSCR, DHSZ, DHSCT*1
  RS, RS2, IVCK, IVDR, IVRD, IVWR, IVBWR*1

  *1. FX3G PLC is not compatible.

- **FX1S, FX1N, FX1NC, FX2N and FX2NC PLCs**
  DHSCS, DHSCR, DHSZ(FX1S, FX1N, FX1NC, FX2N and FX2NC PLCs)
  RS (FX2N and FX2NC PLCs)
  PLSY, PLSR, RS, ZRN, PLSV, DRVI, DRVA(FX1S, FX1N and FX1NC PLCs)
- **FX0, FX0S, FX0N, FXU and FX2C PLCs**
  DHSCS, DHSCR, DHSZ(FX0, FX0S, FX0N, FXU and FX2C PLCs)
  RS (FX0N, FXU and FX2C PLCs)

2. Instruction List

This chapter introduces a list of instructions available in programming.

2.1 Basic Instructions

Instruction name	Function	Applicable PLCs: FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	Reference
LD	Initial logical operation contact type NO (normally open)	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.1
LDI	Initial logical operation contact type NC (normally closed)									
AND	Serial connection of NO contacts									
ANI	Serial connection of NC contacts									
OR	Parallel connection of NO contacts									
ORI	Parallel connection of NC contacts									
LDP	Initial logical operation of rising edge pulse	✓	✓	✓	✓	✓				Section 5.2
LDF	Initial logical operation of falling edge pulse									
ANDP	Serial connection of rising edge pulse									
ANDF	Serial connection of falling edge pulse									
ORP	Parallel connection of rising edge pulse									
ORF	Parallel connection of falling edge pulse									
OUT	Coil drive	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.3
OUT_T	Timer drive	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.4.1
OUT_C	Counter drive	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.5.1
OUT_C_32										
AND(•••)	Serial connection of circuit block	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.6
OR(•••)	Parallel connection of circuit block									
MPS	Stack pushdown	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.7
MRD	Stack read									
MPP	Stack popup									
INV	Invert the current result of the internal PLC operations	✓	✓	✓	✓	✓				Section 5.8
MEP	Conversion of operation result to leading edge pulse	*1	✓							Section 5.9
MEF	Conversion of operation result to trailing edge pulse									
SET	Set bit device latch ON	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.10
RST	Reset bit device OFF and clear current value and resister									
PLS	Rising edge pulse differential output	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.11
PLF	Falling edge pulse differential output									
MC	Connection to common contact	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.12
MCR	Clear connection to common contact									
END	Program END, I/O refresh and return to step 0	✓	✓	✓	✓	✓	✓	✓	✓	Section 5.13
NOP	No operation or null step									Section 5.14

*1. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.30 or later.

2.2 Step Ladder Instructions

Instruction name	Function	Applicable PLCs								Reference
		FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
STL	Starts step ladder	✓	✓	✓	✓	✓	✓	✓	✓	Section 6.2
RET	Completes step ladder	✓	✓	✓	✓	✓	✓	✓	✓	Section 6.3

2.3 Applied Instructions

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Program Flow											
CJ	Continuous	Conditional jump	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.1.1
CJP	Pulse		✓	✓	✓	✓	✓	✓			
CALL	Continuous	Call subroutine	✓	✓	✓	✓	✓	✓			Section 7.1.2
CALLP	Pulse		✓	✓	✓	✓	✓	✓			
SRET	Continuous	Subroutine return	✓	✓	✓	✓	✓	✓			Section 7.1.3
IRET	Continuous	Interrupt return	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.1.4
DI	Continuous	Disable interrupt	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.1.5
EI	Continuous	Enable interrupt	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.1.6
FEND	Continuous	Main routine program end	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.1.7
WDT	Continuous	Watchdog timer refresh	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.1.8
WDTP	Pulse		✓	✓	✓	✓	✓	✓			
FOR	Continuous	Start a FOR/NEXT loop	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.1.9
NEXT	Continuous	End a FOR/NEXT loop	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.1.10

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Move and Compare											
CMP	Continuous	Compare	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.2.1
CMPP	Pulse		✓	✓	✓	✓	✓	✓			
DCMP	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DCMPP	Pulse		✓	✓	✓	✓	✓	✓			
ZCP	Continuous	Zone compare	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.2.2
ZCPP	Pulse		✓	✓	✓	✓	✓	✓			
DZCP	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DZCPP	Pulse		✓	✓	✓	✓	✓	✓			
MOV	Continuous	Move	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.2.3
MOVP	Pulse		✓	✓	✓	✓	✓	✓			
DMOV	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DMOVP	Pulse		✓	✓	✓	✓	✓	✓			
SMOV	Continuous	Shift move	✓	✓	✓			✓			Section 7.2.4
SMOVP	Pulse		✓	✓	✓			✓			
CML	Continuous	Inversion move	✓	✓	✓			✓			Section 7.2.5
CMLP	Pulse		✓	✓	✓			✓			
DCML	Continuous		✓	✓	✓			✓			
DCMLP	Pulse		✓	✓	✓			✓			
BMOV	Continuous	Block move	✓	✓	✓	✓	✓	✓	✓		Section 7.2.6
BMOVP	Pulse		✓	✓	✓	✓	✓	✓			
FMOV	Continuous	Fill move	✓	✓	✓			✓			Section 7.2.7
FMOVP	Pulse		✓	✓	✓			✓			
DFMOV	Continuous		✓	✓	✓			*1			
DFMOVP	Pulse		✓	✓	✓			*1			
XCH	Continuous	Exchange	✓		✓			✓			Section 7.2.8
XCHP	Pulse		✓		✓			✓			
DXCH	Continuous		✓		✓			✓			
DXCHP	Pulse		✓		✓			✓			
BCD	Continuous	Conversion to binary coded decimal	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.2.9
BCDP	Pulse		✓	✓	✓	✓	✓	✓			
DBCD	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DBCDP	Pulse		✓	✓	✓	✓	✓	✓			

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.
*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.
*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.
*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.
*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.
*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.
*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.
*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.
*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.
*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.
*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Move and Compare											
BIN	Continuous	Conversion to binary	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.2.10
BINP	Pulse		✓	✓	✓	✓	✓	✓			
DBIN	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DBINP	Pulse		✓	✓	✓	✓	✓	✓			
Arithmetic and Logical Operation											
ADD	Continuous	Addition	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.3.1
ADDP	Pulse		✓	✓	✓	✓	✓	✓			
DADD	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DADDP	Pulse		✓	✓	✓	✓	✓	✓			
SUB	Continuous	Subtraction	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.3.2
SUBP	Pulse		✓	✓	✓	✓	✓	✓			
DSUB	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DSUBP	Pulse		✓	✓	✓	✓	✓	✓			
MUL	Continuous	Multiplication	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.3.3
MULP	Pulse		✓	✓	✓	✓	✓	✓			
DMUL	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DMULP	Pulse		✓	✓	✓	✓	✓	✓			
DIV	Continuous	Division	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.3.4
DIVP	Pulse		✓	✓	✓	✓	✓	✓			
DDIV	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DDIVP	Pulse		✓	✓	✓	✓	✓	✓			
INC	Continuous	Increment	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.3.5
INCP	Pulse		✓	✓	✓	✓	✓	✓			
DINC	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DINCP	Pulse		✓	✓	✓	✓	✓	✓			
DEC	Continuous	Decrement	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.3.6
DECP	Pulse		✓	✓	✓	✓	✓	✓			
DDEC	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DDECP	Pulse		✓	✓	✓	✓	✓	✓			

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Arithmetic and Logical Operation											
WAND	Continuous	Logical word AND	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.3.7
WANDP	Pulse		✓	✓	✓	✓	✓	✓			
DAND	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DANDP	Pulse		✓	✓	✓	✓	✓	✓			
WOR	Continuous	Logical word OR	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.3.8
WORP	Pulse		✓	✓	✓	✓	✓	✓			
DOR	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DORP	Pulse		✓	✓	✓	✓	✓	✓			
WXOR	Continuous	Logical exclusive OR	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.3.9
WXORP	Pulse		✓	✓	✓	✓	✓	✓			
DXOR	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
DXORP	Pulse		✓	✓	✓	✓	✓	✓			
NEG	Continuous	Negation	✓		✓			✓			Section 7.3.10
NEGP	Pulse		✓		✓			✓			
DNEG	Continuous		✓		✓			✓			
DNEGP	Pulse		✓		✓			✓			
Rotation and Shift Operation											
ROR	Continuous	Rotation right	✓	✓	✓			✓			Section 7.4.1
RORP	Pulse		✓	✓	✓			✓			
DROR	Continuous		✓	✓	✓			✓			
DRORP	Pulse		✓	✓	✓			✓			
ROL	Continuous	Rotation left	✓	✓	✓			✓			Section 7.4.2
ROLP	Pulse		✓	✓	✓			✓			
DROL	Continuous		✓	✓	✓			✓			
DROLP	Pulse		✓	✓	✓			✓			
RCR	Continuous	Rotation right with carry	✓		✓			✓			Section 7.4.3
RCRP	Pulse		✓		✓			✓			
DRCR	Continuous		✓		✓			✓			
DRCRP	Pulse		✓		✓			✓			
RCL	Continuous	Rotation left with carry	✓		✓			✓			Section 7.4.4
RCLP	Pulse		✓		✓			✓			
DRCL	Continuous		✓		✓			✓			
DRCLP	Pulse		✓		✓			✓			

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.
*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.
*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.
*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.
*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.
*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.
*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.
*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.
*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.
*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.
*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Rotation and Shift Operation											
SFTR	Continuous	Bit shift right	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.4.5
SFTRP	Pulse		✓	✓	✓	✓	✓	✓			
SFTL	Continuous	Bit shift left	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.4.6
SFTLP	Pulse		✓	✓	✓	✓	✓	✓			
WSFR	Continuous	Word shift right	✓	✓	✓			✓			Section 7.4.7
WSFRP	Pulse		✓	✓	✓			✓			
WSFL	Continuous	Word shift left	✓	✓	✓			✓			Section 7.4.8
WSFLP	Pulse		✓	✓	✓			✓			
SFWR	Continuous	Shift write [FIFO/FILO control]	✓	✓	✓	✓	✓	✓			Section 7.4.9
SFWRP	Pulse		✓	✓	✓	✓	✓	✓			
SFRD	Continuous	Shift read [FIFO control]	✓	✓	✓	✓	✓	✓			Section 7.4.10
SFRDP	Pulse		✓	✓	✓	✓	✓	✓			
Data Operation											
ZRST	Continuous	Zone reset	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.5.1
ZRSTP	Pulse		✓	✓	✓	✓	✓	✓			
DECO	Continuous	Decode	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.5.2
DECOP	Pulse		✓	✓	✓	✓	✓	✓			
ENCO	Continuous	Encode	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.5.3
ENCOP	Pulse		✓	✓	✓	✓	✓	✓			
SUM	Continuous	Sum of active bits	✓	✓	✓			✓			Section 7.5.4
SUMP	Pulse		✓	✓	✓			✓			
DSUM	Continuous		✓	✓	✓			✓			
DSUMP	Pulse		✓	✓	✓			✓			
BON	Continuous	Check specified bit status	✓	✓	✓			✓			Section 7.5.5
BONP	Pulse		✓	✓	✓			✓			
DBON	Continuous		✓	✓	✓			✓			
DBONP	Pulse		✓	✓	✓			✓			
MEAN	Continuous	Mean	✓	✓	✓			✓			Section 7.5.6
MEANP	Pulse		✓	✓	✓			✓			
DMEAN	Continuous		✓	✓	✓			*1			
DMEANP	Pulse		✓	✓	✓			*1			
ANS	Continuous	Timed annunciator set	✓	✓	✓			✓			Section 7.5.7

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Data Operation											
ANR	Continuous	Annunciator reset	✓	✓	✓			✓			Section 7.5.8
ANRP	Pulse		✓	✓	✓			✓			
SQR	Continuous	Square root	✓		✓			✓			Section 7.5.9
SQRP	Pulse		✓		✓			✓			
DSQR	Continuous		✓		✓			✓			
DSQRP	Pulse		✓		✓			✓			
FLT	Continuous	Conversion to floating point	✓	*12	✓			*2			Section 7.5.10
FLTP	Pulse		✓	*12	✓			*2			
DFLT	Continuous		✓	*12	✓			*2			
DFLTP	Pulse		✓	*12	✓			*2			
High Speed Processing											
REF	Continuous	Refresh	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.6.1
REFP	Pulse		✓	✓	✓	✓	✓	✓			
REFF	Continuous	Refresh and filter adjust	✓		✓			✓			Section 7.6.2
REFFP	Pulse		✓		✓			✓			
MTR	Continuous	Input matrix	✓	✓	✓	✓	✓	✓			Section 7.6.3
DHSCS	Continuous	High speed counter set	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.6.4
DHSCR	Continuous	High speed counter reset	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.6.5
DHSZ	Continuous	High speed counter zone compare	✓	✓	✓			✓			Section 7.6.6
SPD	Continuous	Speed detection	✓	✓	✓	✓	✓	✓			Section 7.6.7
DSPD	Continuous		*3	✓							
PLSY	Continuous	Pulse Y output	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.6.8
DPLSY	Continuous		✓	✓	✓	✓	✓	✓	✓	✓	
PWM	Continuous	Pulse width modulation	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.6.9
PLSR	Continuous	Acceleration/deceleration setup	✓	✓	✓	✓	✓				Section 7.6.10
DPLSR	Continuous		✓	✓	✓	✓	✓				
Handy Instruction											
IST	Continuous	Initial state	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.7.1
SER	Continuous	Search a data stack	✓	✓	✓			*2			Section 7.7.2
SERP	Pulse		✓	✓	✓			*2			
DSER	Continuous		✓	✓	✓			*2			
DSERP	Pulse		✓	✓	✓			*2			

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs: FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	Reference
Handy Instruction											
ABSD	Continuous	Absolute drum sequencer	✓	✓	✓	✓	✓	✓			Section 7.7.3
DABSD	Continuous		✓	✓	✓	✓	✓	*1			
INCD	Continuous	Incremental drum sequencer	✓	✓	✓	✓	✓	✓			Section 7.7.4
TTMR	Continuous	Teaching timer	✓		✓			✓			Section 7.7.5
STMR	Continuous	Special timer	✓		✓			✓			Section 7.7.6
ALT	Continuous	Alternate state	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.7.7
ALTP	Pulse		✓	✓	✓	✓	✓	✓			
RAMP	Pulse	Ramp variable value	✓	✓	✓	✓	✓	✓	✓	✓	Section 7.7.8
ROTC	Continuous	Rotary table control	✓		✓			✓			Section 7.7.9
SORT	Continuous	SORT tabulated data	✓		✓			*2			Section 7.7.10
External FX I/O Device											
TKY	Continuous	Ten key input	✓		✓			✓			Section 7.8.1
DTKY	Continuous		✓		✓			✓			
HKY	Continuous	Hexadecimal input	✓		✓			✓			Section 7.8.2
DHKY	Continuous		✓		✓			✓			
DSW	Continuous	Digital switch	✓	✓	✓	✓	✓	✓			Section 7.8.3
SEGD	Continuous	Seven segment decoder	✓		✓			✓			Section 7.8.4
SEGDP	Pulse		✓		✓			✓			
SEGL	Continuous	Seven segment with latch	✓	✓	✓	✓	✓	✓			Section 7.8.5
ARWS	Continuous	Arrow switch	✓		✓			✓			Section 7.8.6
ASC	Continuous	ASCII code data input	✓		✓			✓			Section 7.8.7
PR	Continuous	Print (ASCII code)	✓		✓			✓			Section 7.8.8
FROM	Continuous	Read from a special function block	✓	✓	✓	✓		*4	✓		Section 7.8.9
FROMP	Pulse		✓	✓	✓	✓		*4			
DFROM	Continuous		✓	✓	✓	✓		*4	✓		
DFROMP	Pulse		✓	✓	✓	✓		*4			
TO	Continuous	Write to a special function block	✓	✓	✓	✓		*4	✓		Section 7.8.10
TOP	Pulse		✓	✓	✓	✓		*4			
DTO	Continuous		✓	✓	✓	✓		*4	✓		
DTOP	Pulse		✓	✓	✓	✓		*4			

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
External Device (optional devices)											
RS	Continuous	Serial Communication	✓	✓	✓	✓	✓	*2	*5		Section 7.9.1
PRUN	Continuous	Parallel run (octal mode)	✓	✓	✓	✓	✓	✓			Section 7.9.2
PRUNP	Pulse		✓	✓	✓	✓	✓	✓			
DPRUN	Continuous		✓	✓	✓	✓	✓	✓			
DPRUNP	Pulse		✓	✓	✓	✓	✓	✓			
ASCI	Continuous	Hexadecimal to ASCII conversion	✓	✓	✓	✓	✓	*2	*5		Section 7.9.3
ASCIP	Pulse		✓	✓	✓	✓	✓	*2			
HEX	Continuous	ASCII to hexadecimal conversion	✓	✓	✓	✓	✓	*2	*5		Section 7.9.4
HEXP	Pulse		✓	✓	✓	✓	✓	*2			
CCD	Continuous	Check code	✓	✓	✓	✓	✓	*2	*5		Section 7.9.5
CCDP	Pulse		✓	✓	✓	✓	✓	*2			
VRRD	Continuous	Volume read		*12	*6	*6	✓	✓			Section 7.9.6
VRRDP	Pulse			*12	*6	*6	✓	✓			
VRSC	Continuous	Volume scale		*12	*6	*6	✓	✓			Section 7.9.7
VRSCP	Pulse			*12	*6	*6	✓	✓			
RS2	Continuous	Serial data communication	✓	✓							Section 7.9.8
PID	Continuous	PID control loop	✓	✓	✓	✓	✓	*7			Section 7.9.9
External Device											
MNET	Continuous	F-16NP/NT communication						*8			Section 7.10.1
MNETP	Pulse							*8			
ANRD	Continuous	Read from F1-6A						*8			Section 7.10.2
ANRDP	Pulse							*8			
ANWR	Continuous	Write to F2-6A						*8			Section 7.10.3
ANWRP	Pulse							*8			
RMST	Continuous	F2-32RM start						✓			Section 7.10.4
RMWR	Continuous	Write to F2-32RM						✓			Section 7.10.5
RMWRP	Pulse							✓			
DRMWR	Continuous							✓			
DRMWRP	Pulse							✓			

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
External Device											
RMRD	Continuous	Read from F2-32RM						✓			Section 7.10.6
RMRDP	Pulse							✓			
DRMRD	Continuous							✓			
DRMRDP	Pulse							✓			
RMMN	Continuous	F2-32RM monitor						✓			Section 7.10.7
RMMNP	Pulse							✓			
BLK	Continuous	Specify F2-30GM						*8			Section 7.10.8
BLKP	Pulse							*8			
MCDE	Continuous	2-30GM code						*8			Section 7.10.9
MCDEP	Pulse							*8			
Data Transfer 2											
ZPUSH	Continuous	Batch store of index register	✓								Section 7.11.1
ZPUSHP	Pulse		✓								
ZPOP	Continuous	Batch POP of index register	✓								Section 7.11.2
ZPOPP	Pulse		✓								
Floating Point											
DECMP	Continuous	Floating point compare	✓	*12	✓						Section 7.12.1
DECMPP	Pulse		✓	*12	✓						
DEZCP	Continuous	Floating point zone compare	✓		✓						Section 7.12.2
DEZCPP	Pulse		✓		✓						
DEMOV	Continuous	Floating point move	✓	*12							Section 7.12.3
DEMOVP	Pulse		✓	*12							
DESTR	Continuous	Floating point to character string conversion	✓								Section 7.12.4
DESTRP	Pulse		✓								
DEVAL	Continuous	Character string to floating point conversion	✓								Section 7.12.5
DEVALP	Pulse		✓								
DEBCD	Continuous	Floating point to scientific notation conversion	✓		✓						Section 7.12.6
DEBCDP	Pulse		✓		✓						
DEBIN	Continuous	Scientific notation to floating point conversion	✓		✓						Section 7.12.7
DEBINP	Pulse		✓		✓						
DEADD	Continuous	Floating point addition	✓	*12	✓						Section 7.12.8
DEADDP	Pulse		✓	*12	✓						

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.
*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.
*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.
*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.
*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.
*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.
*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.
*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.
*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.
*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.
*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Floating Point											
DESUB	Continuous	Floating point subtraction	✓	*12	✓						Section 7.12.9
DESUBP	Pulse		✓	*12	✓						
DEMUL	Continuous	Floating point multiplication	✓	*12	✓						Section 7.12.10
DEMULP	Pulse		✓	*12	✓						
DEDIV	Continuous	Floating point division	✓	*12	✓						Section 7.12.11
DEDIVP	Pulse		✓	*12	✓						
DEXP	Continuous	Floating point exponent	✓								Section 7.12.12
DEXPP	Pulse		✓								
DLOGE	Continuous	Floating point natural logarithm	✓								Section 7.12.13
DLOGEP	Pulse		✓								
DLOG10	Continuous	Floating point common logarithm	✓								Section 7.12.14
DLOG10P	Pulse		✓								
DESQR	Continuous	Floating point square root	✓	*12	✓						Section 7.12.15
DESQRP	Pulse		✓	*12	✓						
DENEG	Continuous	Floating point negation	✓								Section 7.12.16
DENEGP	Pulse		✓								
INT	Continuous	Floating point to integer conversion	✓	*12	✓						Section 7.12.17
INTP	Pulse		✓	*12	✓						
DINT	Continuous		✓	*12	✓						
DINTP	Pulse		✓	*12	✓						
DSIN	Continuous	Floating point sine	✓		✓						Section 7.12.18
DSINP	Pulse		✓		✓						
DCOS	Continuous	Floating point cosine	✓		✓						Section 7.12.19
DCOSP	Pulse		✓		✓						
DTAN	Continuous	Floating point tangent	✓		✓						Section 7.12.20
DTANP	Pulse		✓		✓						
DASIN	Continuous	Floating point arc sine	✓								Section 7.12.21
DASINP	Pulse		✓								
DACOS	Continuous	Floating point arc cosine	✓								Section 7.12.22
DACOSP	Pulse		✓								
DATAN	Continuous	Floating point arc tangent	✓								Section 7.12.23
DATANP	Pulse		✓								

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.
*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.
*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.
*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.
*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.
*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.
*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.
*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.
*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.
*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.
*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	Reference
Floating Point											
DRAD	Continuous	Floating point degrees to radians conversion	✓								Section 7.12.24
DRADP	Pulse		✓								
DDEG	Continuous	Floating point radians to degrees conversion	✓								Section 7.12.25
DDEGP	Pulse		✓								
Data Operation 2											
WSUM	Continuous	Sum of word data	*9								Section 7.13.1
WSUMP	Pulse		*9								
DWSUM	Continuous		*9								
DWSUMP	Pulse		*9								
WTOB	Continuous	WORD to BYTE	*9								Section 7.13.2
WTOBP	Pulse		*9								
BTOW	Continuous	BYTE to WORD	*9								Section 7.13.3
BTOWP	Pulse		*9								
UNI	Continuous	4-bit linking of word data	*9								Section 7.13.4
UNIP	Pulse		*9								
DIS	Continuous	4-bit grouping of word data	*9								Section 7.13.5
DISP	Pulse		*9								
SWAP	Continuous	Byte swap	✓		✓						Section 7.13.6
SWAPP	Pulse		✓		✓						
DSWAP	Continuous		✓		✓						
DSWAPP	Pulse		✓		✓						
SORT2	Continuous	Sort tabulated data 2	*9								Section 7.13.7
DSORT2	Continuous		*9								
Positioning Control											
DSZR	Continuous	Dog search zero return	✓	✓							Section 7.14.1
DVIT	Continuous	Interrupt positioning	✓								Section 7.14.2
DDVIT	Continuous		✓								
DTBL	Continuous	Batch data positioning mode	*9	✓							Section 7.14.3
DABS	Continuous	Absolute current value read	✓	✓	*10	✓	✓				Section 7.14.4
ZRN	Continuous	Zero return	✓	✓		✓	✓				Section 7.14.5
DZRN	Continuous		✓	✓		✓	✓				
PLSV	Continuous	Variable speed pulse output	✓	✓		✓	✓				Section 7.14.6
DPLSV	Continuous		✓	✓		✓	✓				

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.
*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.
*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.
*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.
*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.
*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.
*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.
*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.
*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.
*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.
*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Positioning Control											
DRVI	Continuous	Drive to increment	✓	✓		✓	✓				Section 7.14.7
DDRVI	Continuous		✓	✓		✓	✓				
DRVA	Continuous	Drive to absolute	✓	✓		✓	✓				Section 7.14.8
DDRVA	Continuous		✓	✓		✓	✓				
Real Time Clock Control											
TCMP	Continuous	RTC data compare	✓	✓	✓	✓	✓				Section 7.15.1
TCMPP	Pulse		✓	✓	✓	✓	✓				
TZCP	Continuous	RTC data zone compare	✓	✓	✓	✓	✓				Section 7.15.2
TZCPP	Pulse		✓	✓	✓	✓	✓				
TADD	Continuous	RTC data addition	✓	✓	✓	✓	✓				Section 7.15.3
TADDP	Pulse		✓	✓	✓	✓	✓				
TSUB	Continuous	RTC data subtraction	✓	✓	✓	✓	✓				Section 7.15.4
TSUBP	Pulse		✓	✓	✓	✓	✓				
HTOS	Continuous	Hour to second conversion	✓								Section 7.15.5
HTOSP	Pulse		✓								
DHTOS	Continuous		✓								
DHTOSP	Pulse		✓								
STOH	Continuous	Second to hour conversion	✓								Section 7.15.6
STOHP	Pulse		✓								
DSTOH	Continuous		✓								
DSTOHP	Pulse		✓								
TRD	Continuous	Read RTC data	✓	✓	✓	✓	✓				Section 7.15.7
TRDP	Pulse		✓	✓	✓	✓	✓				
TWR	Continuous	Set RTC data	✓	✓	✓	✓	✓				Section 7.15.8
TWRP	Pulse		✓	✓	✓	✓	✓				
HOUR	Continuous	Hour meter	✓	✓	*10	✓	✓				Section 7.15.9
DHOUR	Continuous		✓	✓	*10	✓	✓				
External Device											
GRY	Continuous	Decimal to gray code conversion	✓	✓	✓						Section 7.16.1
GRYP	Pulse		✓	✓	✓						
DGRY	Continuous		✓	✓	✓						
DGRYP	Pulse		✓	✓	✓						

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.
*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.
*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.
*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.
*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.
*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.
*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.
*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.
*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.
*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.
*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs: FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	Reference
External Device											
GBIN	Continuous	Gray code to decimal conversion	✓	✓	✓						Section 7.16.2
GBINP	Pulse		✓	✓	✓						
DGBIN	Continuous		✓	✓	✓						
DGBINP	Pulse		✓	✓	✓						
RD3A	Continuous	Read from dedicated analog block	✓	✓	*10	✓					Section 7.16.3
RD3AP	Pulse		✓	✓	*10	✓					
WR3A	Continuous	Write to dedicated analog block	✓	✓	*10	✓					Section 7.16.4
WR3AP	Pulse		✓	✓	*10	✓					
Extension Function											
EXTR_IN	Continuous	External ROM function			*10						Section 7.17.1
EXTRP_IN	Pulse				*10						
EXTR_OUT	Continuous				*10						Section 7.17.2
EXTRP_OUT	Pulse				*10						
Others											
COMRD	Continuous	Read device comment data	*9								Section 7.18.1
COMRDP	Pulse		*9								
RND	Continuous	Random number generation	✓								Section 7.18.2
RNDP	Pulse		✓								
DUTY	Continuous	Timing pulse generation	*9								Section 7.18.3
CRC	Continuous	Cyclic redundancy check	✓								Section 7.18.4
CRCP	Pulse		✓								
DHCMOV	Continuous	High speed counter move	✓								Section 7.18.5
Block Data Operation											
BK+	Continuous	Block data addition	*9								Section 7.19.1
BK+P	Pulse		*9								
DBK+	Continuous		*9								
DBK+P	Pulse		*9								
BK-	Continuous	Block data subtraction	*9								Section 7.19.2
BK-P	Pulse		*9								
DBK-	Continuous		*9								
DBK-P	Pulse		*9								

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Block Data Operation											
BKCMP=	Continuous	Block data compare	*9								Section 7.19.3
BKCMP>	Continuous		*9								
BKCMP<	Continuous		*9								
BKCMP<>	Continuous		*9								
BKCMP<=	Continuous		*9								
BKCMP>=	Continuous		*9								
BKCMP=P	Pulse		*9								
BKCMP>P	Pulse		*9								
BKCMP<P	Pulse		*9								
BKCMP<>P	Pulse		*9								
BKCMP<=P	Pulse		*9								
BKCMP>=P	Pulse		*9								
DBKCMP=	Continuous		*9								
DBKCMP>	Continuous		*9								
DBKCMP<	Continuous		*9								
DBKCMP<>	Continuous		*9								
DBKCMP<=	Continuous		*9								
DBKCMP>=	Continuous		*9								
DBKCMP=P	Pulse		*9								
DBKCMP>P	Pulse		*9								
DBKCMP<P	Pulse		*9								
DBKCMP<>P	Pulse		*9								
DBKCMP<=P	Pulse		*9								
DBKCMP>=P	Pulse		*9								
Character String Control											
STR	Continuous	BIN to character string conversion	*9								Section 7.20.1
STRP	Pulse		*9								
DSTR	Continuous		*9								
DSTRP	Pulse		*9								
VAL	Continuous	Character string to BIN conversion	*9								Section 7.20.2
VALP	Pulse		*9								
DVAL	Continuous		*9								
DVALP	Pulse		*9								

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.
*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.
*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.
*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.
*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.
*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.
*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.
*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.
*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.
*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.
*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.
*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Character String Control											
$+	Continuous	Link character strings	✓								Section 7.20.3
$+P	Pulse		✓								
LEN	Continuous	Character string length detection	✓								Section 7.20.4
LENP	Pulse		✓								
RIGHT	Continuous	Extracting character string data from the right	✓								Section 7.20.5
RIGHTP	Pulse		✓								
LEFT	Continuous	Extracting character string data from the left	✓								Section 7.20.6
LEFTP	Pulse		✓								
MIDR	Continuous	Random selection of character strings	✓								Section 7.20.7
MIDRP	Pulse		✓								
MIDW	Continuous	Random replacement of character strings	✓								Section 7.20.8
MIDWP	Pulse		✓								
INSTR	Continuous	Character string search	*9								Section 7.20.9
INSTRP	Pulse		*9								
$MOV	Continuous	Character string transfer	✓								Section 7.20.10
$MOVP	Pulse		✓								
Data Operation 3											
FDEL	Continuous	Deleting data from tables	*9								Section 7.21.1
FDELP	Pulse		*9								
FINS	Continuous	Inserting data to tables	*9								Section 7.21.2
FINSP	Pulse		*9								
POP	Continuous	Shift last data read [FILO control]	✓								Section 7.21.3
POPP	Pulse		✓								
SFR	Continuous	Bit shift right with carry	✓								Section 7.21.4
SFRP	Pulse		✓								
SFL	Continuous	Bit shift left with carry	✓								Section 7.21.5
SFLP	Pulse		✓								

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Data Comparison											
LD=	Continuous	Load compare	✓	✓	✓	✓	✓				Section 7.22.1
LD>	Continuous		✓	✓	✓	✓	✓				
LD<	Continuous		✓	✓	✓	✓	✓				
LD<>	Continuous		✓	✓	✓	✓	✓				
LD<=	Continuous		✓	✓	✓	✓	✓				
LD>=	Continuous		✓	✓	✓	✓	✓				
LDD=	Continuous		✓	✓	✓	✓	✓				
LDD>	Continuous		✓	✓	✓	✓	✓				
LDD<	Continuous		✓	✓	✓	✓	✓				
LDD<>	Continuous		✓	✓	✓	✓	✓				
LDD<=	Continuous		✓	✓	✓	✓	✓				
LD>=	Continuous		✓	✓	✓	✓	✓				
AND=	Continuous	AND compare	✓	✓	✓	✓	✓				Section 7.22.2
AND>	Continuous		✓	✓	✓	✓	✓				
AND<	Continuous		✓	✓	✓	✓	✓				
AND<>	Continuous		✓	✓	✓	✓	✓				
AND<=	Continuous		✓	✓	✓	✓	✓				
AND>=	Continuous		✓	✓	✓	✓	✓				
ANDD=	Continuous		✓	✓	✓	✓	✓				
ANDD>	Continuous		✓	✓	✓	✓	✓				
ANDD<	Continuous		✓	✓	✓	✓	✓				
ANDD<>	Continuous		✓	✓	✓	✓	✓				
ANDD<=	Continuous		✓	✓	✓	✓	✓				
ANDD>=	Continuous		✓	✓	✓	✓	✓				

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Data Comparison											
OR=	Continuous	OR compare	✓	✓	✓	✓	✓				Section 7.22.3
OR>	Continuous		✓	✓	✓	✓	✓				
OR<	Continuous		✓	✓	✓	✓	✓				
OR<>	Continuous		✓	✓	✓	✓	✓				
OR<=	Continuous		✓	✓	✓	✓	✓				
OR>=	Continuous		✓	✓	✓	✓	✓				
ORD=	Continuous		✓	✓	✓	✓	✓				
ORD>	Continuous		✓	✓	✓	✓	✓				
ORD<	Continuous		✓	✓	✓	✓	✓				
ORD<>	Continuous		✓	✓	✓	✓	✓				
ORD<=	Continuous		✓	✓	✓	✓	✓				
ORD>=	Continuous		✓	✓	✓	✓	✓				
Data Table Operation											
LIMIT	Continuous	Limit control	✓								Section 7.23.1
LIMITP	Pulse		✓								
DLIMIT	Continuous		✓								
DLIMITP	Pulse		✓								
BAND	Continuous	Dead band control	✓								Section 7.23.2
BANDP	Pulse		✓								
DBAND	Continuous		✓								
DBANDP	Pulse		✓								
ZONE	Continuous	Zone control	✓								Section 7.23.3
ZONEP	Pulse		✓								
DZONE	Continuous		✓								
DZONEP	Pulse		✓								
SCL	Continuous	Scaling (coordinate by point data)	✓								Section 7.23.4
SCLP	Pulse		✓								
DSCL	Continuous		✓								
DSCLP	Pulse		✓								

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Data Table Operation											
DABIN	Continuous	Decimal ASCII to BIN conversion	*9								Section 7.23.5
DABINP	Pulse		*9								
DDABIN	Continuous		*9								
DDABINP	Pulse		*9								
BINDA	Continuous	BIN to decimal ASCII conversion	*9								Section 7.23.6
BINDAP	Pulse		*9								
DBINDA	Continuous		*9								
DBINDAP	Pulse		*9								
SCL2	Continuous	Scaling 2 (coordinate by X/Y data)	✓								Section 7.23.7
SCL2P	Pulse		✓								
DSCL2	Continuous		✓								
DSCL2P	Pulse		✓								
External Device Communication (Inverter Communication)											
IVCK	Continuous	Inverter status check	✓	*12							Section 7.24.1
IVDR	Continuous	Inverter drive	✓	*12							Section 7.24.2
IVRD	Continuous	Inverter parameter read	✓	*12							Section 7.24.3
IVWR	Continuous	Inverter parameter write	✓	*12							Section 7.24.4
IVBWR	Continuous	Inverter parameter block write	✓								Section 7.24.5
Data Transfer 3											
RBFM	Continuous	Divided BFM read	*9								Section 7.25.1
WBFM	Continuous	Divided BFM write	*9								Section 7.25.2
High Speed Processing 2											
DHSCT	Continuous	High speed counter compare with data table	✓								Section 7.26.1

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

Instruction name	Execution condition	Function	Applicable PLCs								Reference
			FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)	
Extension File Register Control											
LOADR	Continuous	Load from ER	✓	✓							Section 7.27.1
LOADRP	Pulse	Load from ER	✓	✓							Section 7.27.1
SAVER	Continuous	Save to ER	✓								Section 7.27.2
INITR	Continuous	Initialize R and ER	✓								Section 7.27.3
INITRP	Pulse	Initialize R and ER	✓								Section 7.27.3
LOGR	Continuous	Logging R and ER	✓								Section 7.27.4
LOGRP	Pulse	Logging R and ER	✓								Section 7.27.4
RWER	Continuous	Rewrite to ER	*11	✓							Section 7.27.5
RWERP	Pulse	Rewrite to ER	*11	✓							Section 7.27.5
INITER	Continuous	Initialize ER	*11								Section 7.27.6
INITERP	Pulse	Initialize ER	*11								Section 7.27.6
FX3U-CF-ADP											
FLCRT	Continuous	File create / check	*13								Section 7.28.1
FLDEL	Continuous	File delete / CF card format	*13								Section 7.28.2
FLWR	Continuous	Data write	*13								Section 7.28.3
FLRD	Continuous	Data read	*13								Section 7.28.4
FLCMD	Continuous	CF-ADP command	*13								Section 7.28.5
FLSTRD	Continuous	CF-ADP status read	*13								Section 7.28.6

*1. The instruction is provided in the FXU PLC Ver. 2.30 or later.

*2. The instruction is provided in the FXU PLC Ver. 3.07 or later.

*3. The 32-bit operations are provided in the FX3U and FX3UC PLCs Ver. 2.20 or later.

*4. The instruction is provided in the FXU PLC Ver. 2.10 or later.

*5. The instruction is provided in the FX0N PLC Ver. 1.20 or later.

*6. Though programmed, this instruction is not valid because the FX1NC or FX2NC PLC does not have a volume to read out under this instruction.

*7. The instruction is provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*8. The instruction is not provided in the FXU and FX2C PLCs Ver. 3.30 or later.

*9. The instruction is provided in the FX3UC PLC Ver. 2.20 or later.

*10. The instruction is provided in the FX2N and FX2NC PLCs Ver. 3.00 or later.

*11. The instruction is provided in the FX3UC PLC Ver. 1.30 or later.

*12. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

*13. The instruction is provided in the FX3U and FX3UC PLCs Ver. 2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

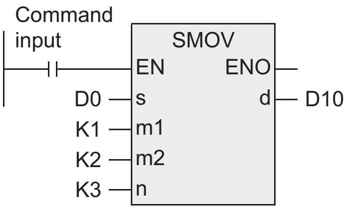
3. Configuration of Instruction

This chapter explains the configuration of sequence instructions.

3.1 Expression and Operation Form of Sequence Instructions

Instructions and arguments

- Each instruction is given a specific name that indicates its contents.
  "SMOV" (shift move) is one of such examples.
- Each instruction consists of the arguments that indicate input and output data used in that particular instruction.



(s)	:	This symbol indicates an argument called "source" that does not change its contents by the execution of an instruction.
(d)	:	This symbol indicates an argument called "destination" that changes its contents by the execution of an instruction.
m, n	:	Symbols "m" and "n" indicate an argument that belongs to neither the source nor the destination.

Applicable devices of arguments

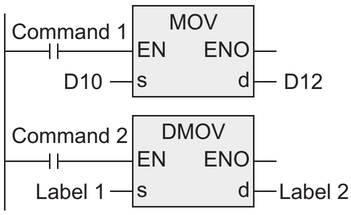
- An input variable (label or device) specifies the applicable device of an argument.
- Bit devices such as X, Y, M and S may be handled.
- These bit devices may be combined to form KnX, KnY, KnM and KnS to be handled as numerical data.
  → **FX Structured Programming Manual (Device & Common)**
- The current value register of data register D, timer T and counter C may be handled.
- When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data. A 32-bit counter can be specified directly as it is a 32-bit long device. Use a global label to specify a device.
  When handling 32-bit data, a 16-bit data register D is a combination of data registers of two consecutive points.
  For example, where data register D0 is defined by a label as the argument of a 32-bit instruction, the 32-bit data of (D1, D0) is handled. (D1 is high order 16 bits and D2 is low order 16 bits.)
  Where the current value registers of T and C are used as general data registers, they are handled in the same manner.

Instruction mode and Operation form

Instructions are divided into "16-bit instructions" and "32-bit instructions" depending on the size of values they handle. The instructions also have characteristics of either a "continuous execution" or "pulse execution" depending on the form of execution.
Some of the instructions have all these combinations.

1. 16-bit and 32-bit instructions

- An applied instruction that handles a numeric value is either 16 bits or 32 bits depending on the bit length of the numeric value data.



Instruction that transfers the D10 contents to D12

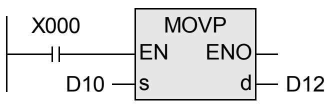
Instruction that transfers the contents of (D21, D20) to (D23, D22).
Label 1 and label 2 define D20 and D22, respectively.

- Where it is a 32-bit instruction, "D" is added to express as "DMOV".
- The specified device can be an even number or an odd number and is used in combination with the device of the next higher number. (In the case of word devices such as T, C and D)
  To avoid confusion, it is suggested to give an even number to the low order device specified by the argument of a 32-bit instruction.
- A 32-bit counter is of the size of 32 bits with this device alone, enabling to directly specify as an argument.

2. Pulse execution and continuous execution instructions

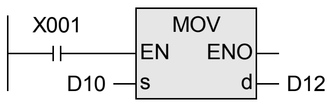
Pulse operation

In an example shown on the right, when X000 changes from OFF to ON, the instruction is executed only once. No other execution takes place.
It is therefore suggested that the instructions of pulse operation be used if not executed all the time.
Symbol "P" indicates that the instruction is of pulse operation.
The same is applied to DMOVP.



Continuous operation

The instruction in the figure on the right is of continuous operation.
It is executed in each cycle of operation while X001 is ON.



Where continuous execution instructions such as INC and DEC are used, some instructions have the destination contents be changed in each cycle of operation.

In either cases, the instruction is not executed if the drive input X000 or X001 is OFF. The destination does not change either if the instruction is not specified otherwise.

3.2 Labels

Types of labels

Labels are either global labels or local labels.

- Global labels are available for use in program blocks and function blocks.
- Local labels are available for use only in a declared program part.

Label classes

The label classes indicate how they are used in which program parts.
The table below shows the label classes.

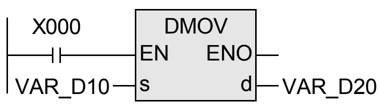
Label class	Descriptions	Program parts available for use		
		Program block	Function	Function block
VAR_GLOBAL	A common label that can be used in all program parts.	✓		✓
VAR_GROBAL_CONSTANT	A common constant that can be used in all program parts.	✓		✓
VAR	A label used within declared program parts. It cannot be used in other program parts.	✓	✓	✓
VAR_CONSTANT	A constant used within declared program parts. It cannot be used in other program parts.	✓	✓	✓
VAR_INPUT	A label that receives values. It cannot be changed within program parts.		✓	✓
VAR_OUTPUT	A label for output from a function block.			✓
VAR_IN_OUT	A local label that receives values and outputs from a program part.			✓

Definition of labels

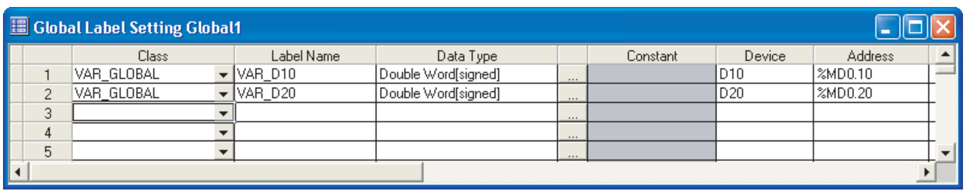
Before using a label, the label needs to be defined. An error is generated if attempting to convert (compile) a program where the label is not defined.

- Where defining a global label, the label name, class, data type and device are interrelated.
- Where defining a local label, the label name, class and data type are set.
  The user does not have to specify a device when using a local label. A device is allocated automatically during the compilation.

The following is an example of setting the label VAR_D10 and VAR_D20 of a DMOV instruction.



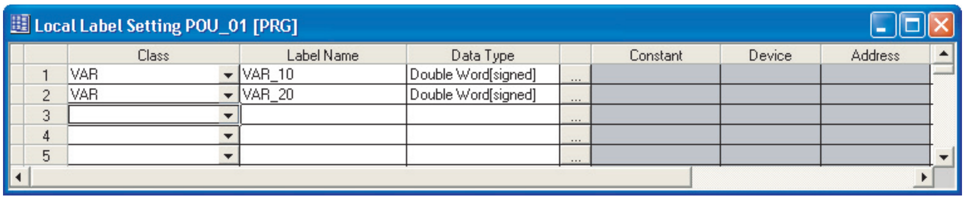
- When using as a global label:
  Set the class, label name and data type and device or address.



Global Label Setting Global1

	Class	Label Name	Data Type		Constant	Device	Address
1	VAR_GLOBAL	VAR_D10	Double Word[signed]	...		D10	%MD0.10
2	VAR_GLOBAL	VAR_D20	Double Word[signed]	...		D20	%MD0.20
3				...			
4				...			
5				...			

- When using as a local label:
  Set the class, label name and data type.



Expressing constants

The following describes the method of expression when setting constant to a label.

Type of constant	Method of expression	Example
Bit	Enter either "FALSE" or "TRUE", or either "0" or "1".	TRUE, FALSE
Binary number	Add "2#" before the binary number.	2#0010, 2#01101010
Octal number	Add "8#" before the octal number.	8#0, 8#337
Decimal number	Enter the decimal number directly. Or, add "K" before the decimal number	123, K123
Hexadecimal number	Add "16#" or "H" before the hexadecimal number.	16#FF, HFF
Real number	Enter the real number directly. Or, add "E" before the real number.	2.34, E2.34
Character string	Put the character string between single quotations (") or double quotations ("").	'ABC', "ABC"

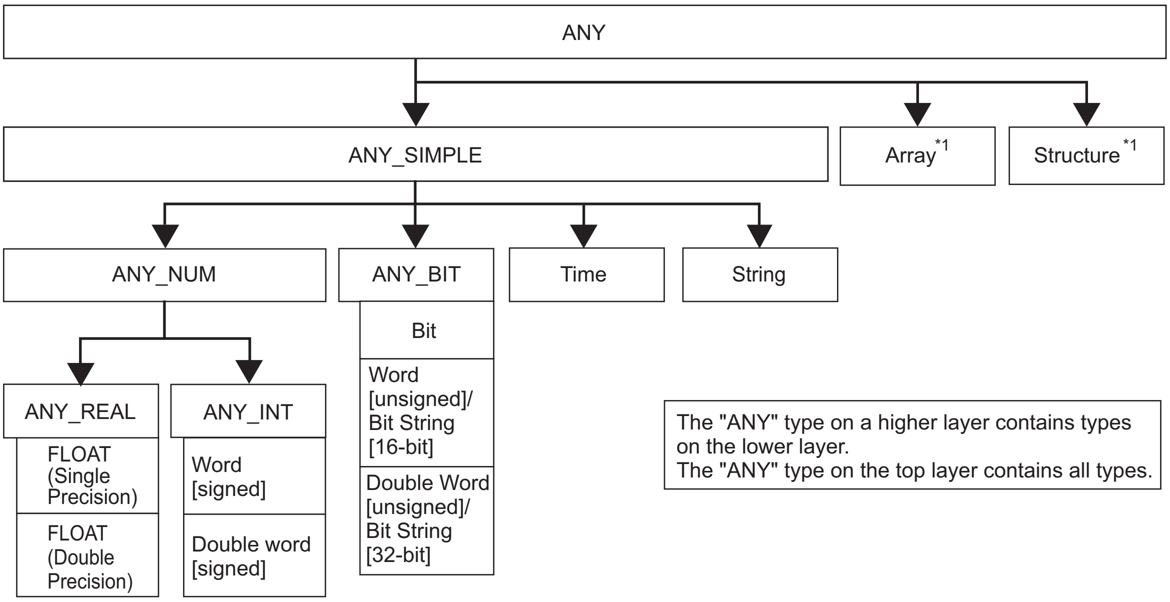
Data type

The data type of label is either basic data type or universal data type.

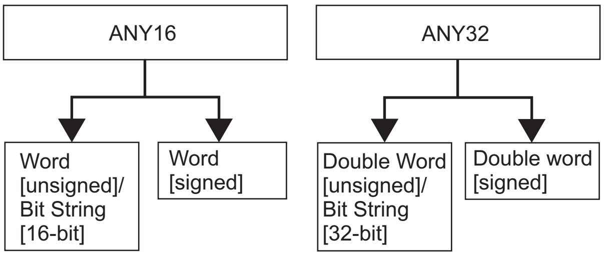
- The table below lists the basic data types.

Data type	Description	Value range	Bit length
Bit	Boolean data	0(FALSE), 1(TRUE)	1 bit
Word [signed]	Integer	-32768 to 32767	16 bits
Double Word [signed]	Double precision integer	-2147483648 to 2147483647	32 bits
Word [unsigned]/Bit String [16-bit]	16-bit data	0 to 65535	16 bits
Double Word [unsigned]/Bit String [32-bit]	32-bit data	0 to 4294967295	32 bits
FLOAT (Single Precision)	Real number	E $\pm 1.175495^{-38}$ to E $\pm 3.402823^{+38}$ (Number of significant figures: 6)	32 bits
String	Character string	(50 characters maximum)	Variable
Time	Time value	T#-24d-0h31m23s648.00ms to T#24d20h31m23s647.00ms	32 bits

- The universal data type is the data type of a label that puts together several basic data types.
  The data type name starts with "ANY".



The "ANY" type on a higher layer contains types on the lower layer.
The "ANY" type on the top layer contains all types.



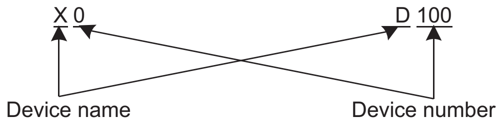
*1 Refer to the following manual for details.
→ Q/FX Structured Programming Manual (Fundamentals)

3.3 Devices and Addresses

A device is expressed by a device or an address.

Device

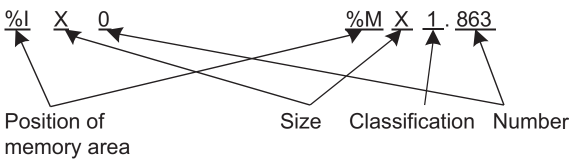
The device is expressed by a device name and a device number.



Address

An address is expressed by a method defined by IEC61131-3.
It is expressed as follows according to IEC61131-3.

Top	1st character: position		2nd character: size		3rd character and onwards: classification	Number
%	I	Input	(Omitted)	Bit	These are the numbers for detailed classification. This number is separated by "." (period) from subsequent numbers. This number may be omitted.	The number that indicates a device number (decimal number).
	Q	Output	X	Bit		
	M	Internal	W	Word (16 bits)		
			D	Double word (32 bits)		



- Position of memory area
  This is the first classification to identify the position of memory area either by input, output and internal where data is allocated.
  X (X device) :I (Input)
  Y (Y device) :Q (Output)
  Device other than above :M (Internal)
- Size
  The principle of expression method corresponding to device (method of expression for MELSEC) is as follows.
  Bit device :X (Bit)
  Word device :W (Word), D (Double word)
- Classification
  This is the second classification to identify the types of device that cannot be classified only by the above position and size.
  X or Y of a device does not classify.

Refer below for the expression corresponding to the device expression.

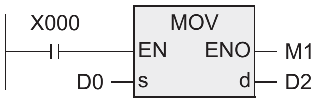
→ **7.3, Appendix A**

3.4 EN and ENO

The execution control is available for an instruction with "EN".

- EN is for entering an execution condition of instruction.
- ENO is for outputting the state of execution of instruction.
- The table below shows the relationships between the EN and ENO and the contents of operation results.

EN	ENO	Operation results
TRUE(Executing operation)	TRUE(Without operation error)	Operation output value
	FALSE(With operation error)	Undefined value
FALSE(Stopping operation)	FALSE	Undefined value



In the instruction above,
instruction MOV is executed
only when X000 is TRUE.

4. How to Read Explanation of Instructions

The following shows one of the pages that explains the instructions.

1) →

7.2.3 MOV

2) →

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction transfers (copies) the contents of a device to another device.

3) →

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
MOV	16 bits	Continuous	MOV (EN ENO / s d)	MOV(EN,s,d); Or an assignment statement
MOVP	16 bits	Pulse	MOVP (EN ENO / s d)	MOVP(EN,s,d); Or an assignment statement
DMOV	32 bits	Continuous	DMOV (EN ENO / s d)	DMOV(EN,s,d); Or an assignment statement
DMOVP	32 bits	Pulse	DMOVP (EN ENO / s d)	DMOVP(EN,s,d); Or an assignment statement

4) →

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Data or device of transfer source	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Transfer destination device	ANY16	ANY32

5) →

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation

6) →

1. 16-bit operation(MOV, MOVP)

The contents of the transfer source specified by (s) are transferred to the transfer destination specified by (d).

- While the command input is OFF, the transfer destination specified by (d) does not change.
- When a constant (K) is specified as the transfer source specified by (s), it is automatically converted into binary.

Command input — MOV (EN ENO); Transfer source data — s; d — Transfer destination data

(s) → (d)

Cautions

7) →

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the pulse operation type instructions.
To execute pulse operation, make the instruction execution condition pulse type.
3) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

8) →

1. When reading the current value of a timer and counter

[Structured ladder]

X001 — MOV (EN ENO); TN0 — s; d — D20

[ST]

MOV(X001,TN0,D20);

(Current value of T0) → (D20)
The operation is the same as a counter.

* The above is different from the actual page, as it is provided for explanation only.

1) Indicates the corresponding chapter, section, subsection, number and instruction name.
2) Indicates the PLCs that support the instruction.

Item	Descriptions
○	Supported by PLCs from the first release.
△	The support conditions depend on the versions. "Cautions" explains the applicable versions.
×	This particular series PLCs do not support the instruction.

3) Indicates the data length, operation form and expression of each instruction.

Item	Descriptions
16 bits	An instruction of 16-bit data length
32 bits	An instruction of 32-bit data length
Continuous	This is a continuous execution instruction that is executed in each cycle of operation while the execution condition (EN) is being satisfied.
Pulse	This is a pulse execution instruction that is executed only when the execution condition (EN) changes from the state of not established to the state of established.
Structured ladder	Indicates a structured ladder language instruction.
ST	Indicates a ST language instruction.

Some PLCs do not support "16 bits / 32 bits" or "continuous / pulse" depending on their versions.
Refer to "Cautions".

4) Indicates the names of the input and output variables of the instruction and the contents and data type of each variable.
Refer to the following manual for details of data type.
→ **Q/FX Structured Programming Manual (Fundamentals)**

5) Applicable devices
"●" indicates the devices that can be used in an instruction.
Devices marked "▲" have restrictions in use.
Refer to "Cautions".

6) Function and operation explanation
Explains the functions that the instruction is responsible for.
This explanation uses an example of structured ladder language.

7) Summarizes the notes before using the instruction.

8) Program example
Explains a program example in each language.

5. Basic Instruction

This chapter introduces the instructions and operators for the structured project corresponding to the basic instructions for the simple project.
Refer to the following manual for variable, instruction and data type.

→ **Q/FX Structured Programming Manual (Fundamentals)**

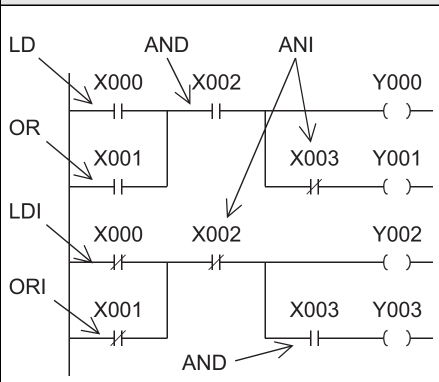
5.1 LD, LDI, AND, ANI, OR, OR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

The LD and LDI instructions are contacts connected to bus lines.
The AND and ANI instructions connect one contact in series.
The OR and ORI instructions connect one contact in parallel.

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
LD	Continuous		To become an assignment statement, operator, control syntax and so forth. The ST language may not have the instructions (symbols) directly corresponding to the contacts of LD, AND and OR that are programmed in a simple project. The structured ladder shown on the left can be expressed as shown below. (When configuring with assignment statements) Y000:=(X000 OR X001) AND X002; Y001:=(X000 OR X001) AND X002 AND NOT X003; Y002:=(NOT X000 OR NOT X001) AND NOT X002; Y003:=(NOT X000 OR NOT X001) AND NOT X002 AND X003;
LDI	Continuous		
AND	Continuous		
ANI	Continuous		
OR	Continuous		
ORI	Continuous		

2. Set data

Variable		Description	Data type
Input variable	-	Variable that are applicable to AND and OR input.	Bit
Output variable	-	Result of operation of AND and OR.	Bit

3. Applicable devices

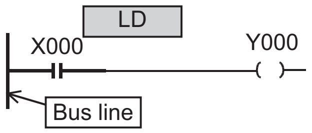
Instruction	Bit Devices: System User							Word Devices: Digit Specification				System User				Special Unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
LD	●	●	●	●	●	●	▲1												▲2					
LDI	●	●	●	●	●	●	▲1												▲2					
AND	●	●	●	●	●	●	▲1												▲2					
ANI	●	●	●	●	●	●	▲1												▲2					
OR	●	●	●	●	●	●	▲1												▲2					
ORI	●	●	●	●	●	●	▲1												▲2					

▲: Refer to "Cautions".

Function and operation explanation

1. LD (Initial logical operation of NO (normally open) contacts)

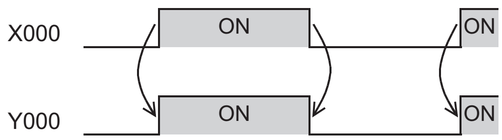
[Structured ladder]



[ST]

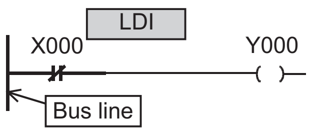
Y000:= X000;

timing chart



2. LDI (Initial logical operation of NC (normally closed) contact type)

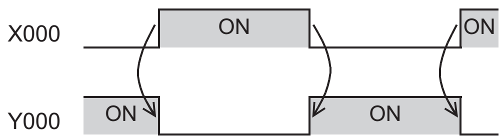
[Structured ladder]



[ST]

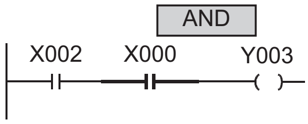
Y000:= NOT X000;

timing chart



3. AND (Serial connection of NO (normally open) contacts)

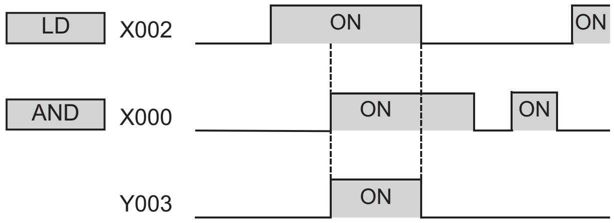
[Structured ladder]



[ST]

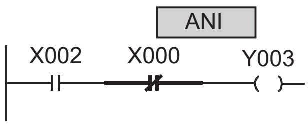
Y003:= X002 AND X000;

timing chart



4. ANI (Serial connection of NC (normally closed) contacts)

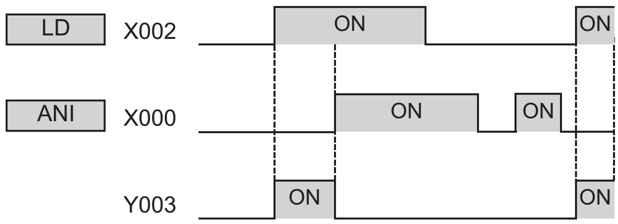
[Structured ladder]



[ST]

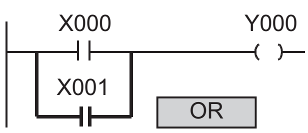
Y003:= X002 AND NOT X000;

timing chart



5. OR (Parallel connection of NO (normally open) contacts)

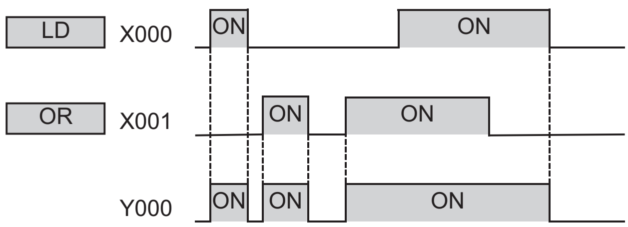
[Structured ladder]



[ST]

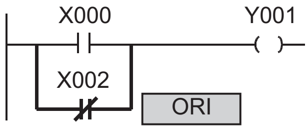
Y000:= X000 OR X001;

timing chart



6. ORI (Parallel connection of NC (normally closed) contacts)

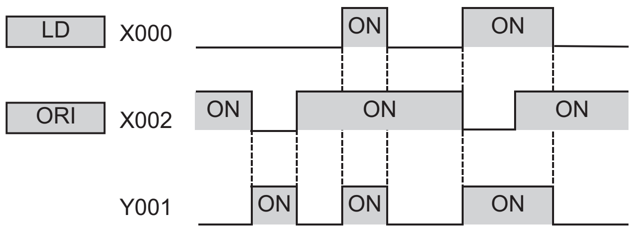
[Structured ladder]



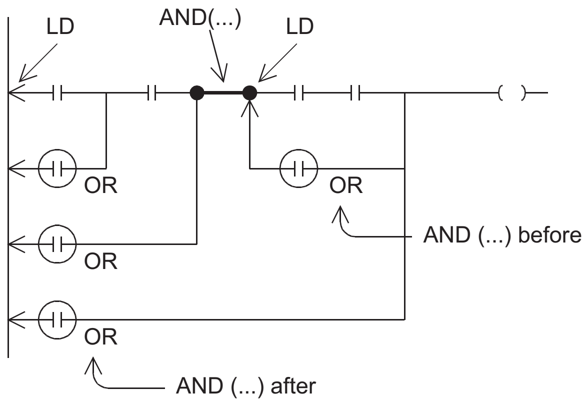
[ST]

Y001:= X000 OR NOT X002;

timing chart



7. Relationship with AND (...)

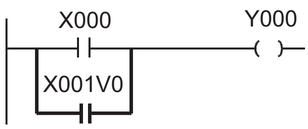


The parallel connection by OR or ORI instruction is connected to the preceding LD or LDI instruction in principle. The "AND (...) after" instruction, however, the parallel connection by OR or ORI instruction is connected to the second preceding LD or LDI instruction.

8. Indexing

Devices used in LD, LDI, AND, ANI, OR and ORI can be indexed with index registers (V, Z).
(State relays (S), special auxiliary relays (M), 32-bit counters (C) or "D□.b" cannot be indexed.)
Applicable only to the FX3U and FX3UC PLCs.

[Structured ladder]



[ST]

Y000:= X000 OR X001V0;

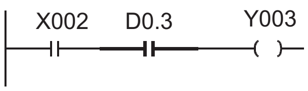
When a used devices is an input (X) or output (Y), the value of an index register (V or Z) is converted into an octal number, and then added.

Example: When the value of V0 is "10", the LD contact is set to ON (becomes conductive) or OFF (becomes nonconductive) by X013.

9. Bit specification of data register (D)

A bit in data register (D) can be specified as a device used in LD, LDI, AND, ANI, OR and ORI.
Applicable only to the FX3U and FX3UC PLCs.

[Structured ladder]



[ST]

Y003:= X002 AND D0.3;

When specifying a bit in data register, input "." after a data register (D) number, and then input a bit number (0 to F) consecutively.
Only 16-bit data resister is applicable.
Specify a bit number as "0 1, 2, ..., 9, A, B, ..., F" from the least significant bit.

Example: In the example shown on the left, LD contact is set to ON (becomes conductive) or OFF (becomes nonconductive) by the bit 3 of D0.

Cautions

1) Some restrictions to applicable devices
   ▲1: The FX3U and FX3UC PLCs only are applicable.
   ▲2: Only the FX3U and FX3UC PLCs are capable of indexing applicable devices.
   The following devices cannot be indexed.
   - Special auxiliary relays (M)
   - 32-bit counters (C)
   - State (S)
   - Word bit specification "D□.b"

Errors

1) When an I/O number used in LD, LDI, AND, ANI, OR or ORI instruction does not exist due to indexing, M8316 (Non-existing I/O specification error) turns ON. (Applicable to the FX3U and FX3UC PLCs only)
2) When the device number of a device (M, T or C) other than I/O does not exist due to indexing, an operation error (error code: 6706) occurs. (Applicable to the FX3U and FX3UC PLCs only)

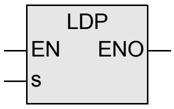
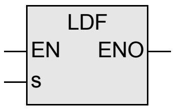
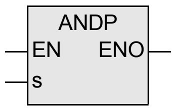
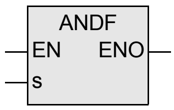
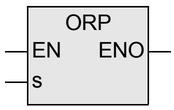
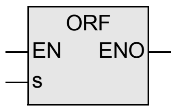
5.2 LDP, LDF, ANDP, ANDF, ORP, ORF

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

Contact instructions LDP, ANDP, and ORP detect the rising edge, and become active during one operation cycle only at the rising edge of a specified bit device (that is, when the bit device turns ON from OFF).
Contact instructions LDF, ANDF and ORF detect the falling edge, and become active during one operation cycle only at the falling edge of a specified bit device (that is, when the bit device turns OFF from ON).

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language	
		Structured ladder	ST
LDP	Pulse (detecting rising pulse)		LDP(EN,s);
LDF	Pulse (detecting falling pulse)		LDF(EN,s);
ANDP	Pulse (detecting rising pulse)		ANDP(EN,s);
ANDF	Pulse (detecting falling pulse)		ANDF(EN,s);
ORP	Pulse (detecting rising pulse)		ORP(EN,s);
ORF	Pulse (detecting falling pulse)		ORF(EN,s);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	LDP,LDF: Always TRUE Except LDP, LDF : BOOL
	(s)	Applicable devices	Bit
Output variable	ENO	Execution state	Bit

3. Applicable devices

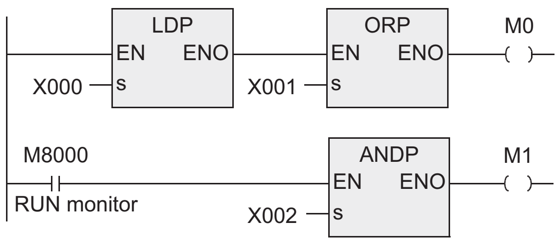
Instruction	Bit Devices							Word Devices													Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
LDP	●	●	●	●	●	●	▲1																	
LDF	●	●	●	●	●	●	▲1																	
ANDP	●	●	●	●	●	●	▲1																	
ANDF	●	●	●	●	●	●	▲1																	
ORP	●	●	●	●	●	●	▲1																	
ORF	●	●	●	●	●	●	▲1																	

▲: Refer to "Cautions".

Function and operation explanation

1. **LDP, ANDP, ORP** (Initial logical operation of rising edge pulse, serial connection and parallel connection)

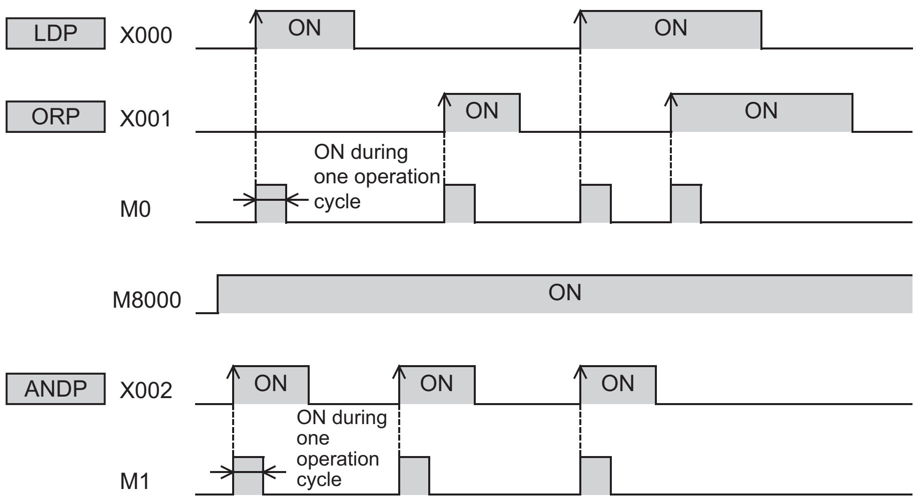
[Structured ladder]



[ST]

```
IF (LDP(TRUE,X0)) OR (LDP(TRUE,X001)) THEN
   M0:= TRUE;
END_IF;
M1:= ANDP(M8000,X002);
```

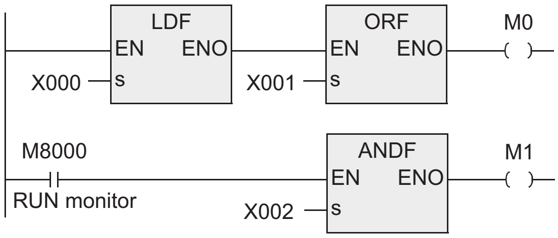
timing chart



In the example shown above, M0 or M1 is ON during only one operation cycle when X000 to X002 turn ON from OFF.

2. LDF, ANDF, ORF (Initial logical operation of falling/trailing edge pulse, serial connection and parallel connection)

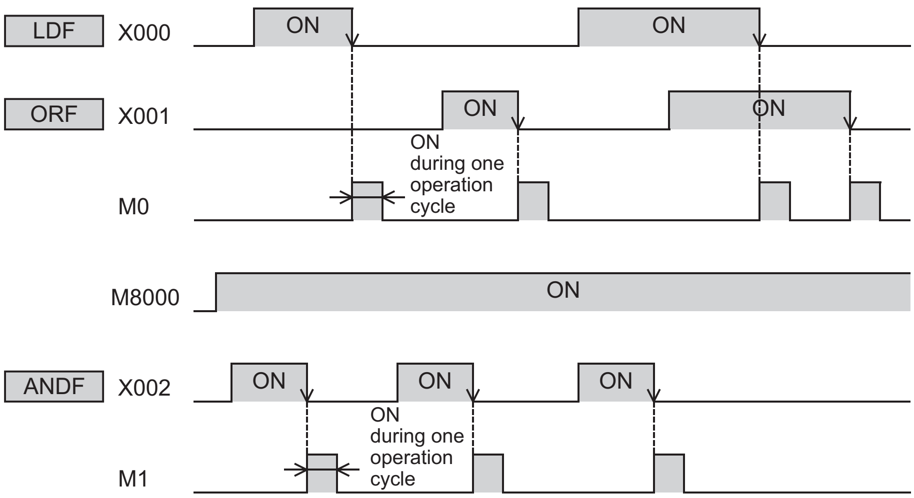
[Structured ladder]



[ST]

```
IF (LDF(TRUE,X0)) OR (LDF(TRUE,X001)) THEN
   M0:= TRUE;
END_IF;
M1:= ANDF(M8000,X002);
```

timing chart

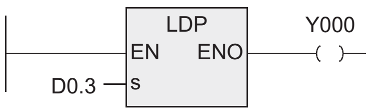


In the example shown above, M0 or M1 is ON during only one operation cycle when X000 to X002 turn OFF from ON.

3. Bit specification of data register (D)

A bit data register (D) can be specified as a device used in LDP, LDF, ANDP, ANDF, ORP and ORF instructions.

[Structured ladder]



[ST]

```
Y000:= LDP(TRUE,D0.3);
```

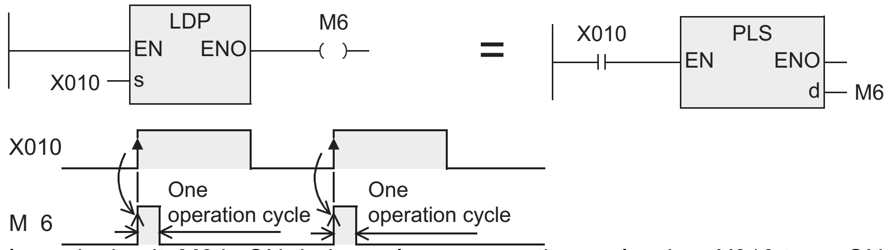
When specifying a bit in data register, input "." after a data register (D) number, and then input a bit number (0 to F) consecutively.
Only 16-bit data resister is applicable.
Specify a bit number as "0 1, 2, ..., 9, A, B, ..., F" from the least significant bit.
Example: In the example shown on the left, LDP contact turns ON (becomes conductive) or OFF (becomes nonconductive) when the bit 3 of D0 turns ON or OFF.

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

4. Output drive side

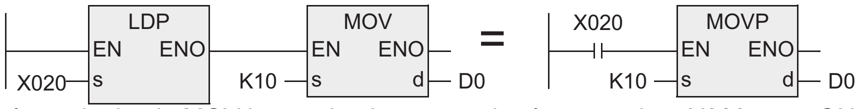
The following two circuits offer the same operation.

<OUT instruction> <Pulse instruction>



In each circuit, M6 is ON during only one operation cycle when X010 turns ON from OFF.

<Rising edge detection> <Pulse instruction (applied instruction)>

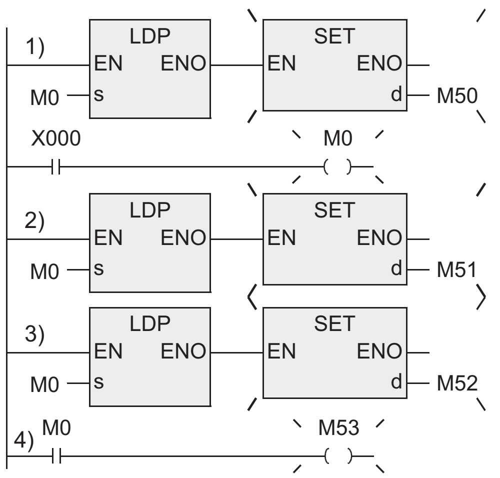


In each circuit, MOV instruction is executed only once when X020 turns ON from OFF.

5. Differences in the operation caused by auxiliary relay (M) numbers

Not supported by the FX1S, FX1N or FX1NC PLC.
When an auxiliary relay (M) is specified as a device in LDP, LDF, ANDP, ANDF, ORP and ORF instructions, the operation varies depending on the device number range as shown in the figure below.

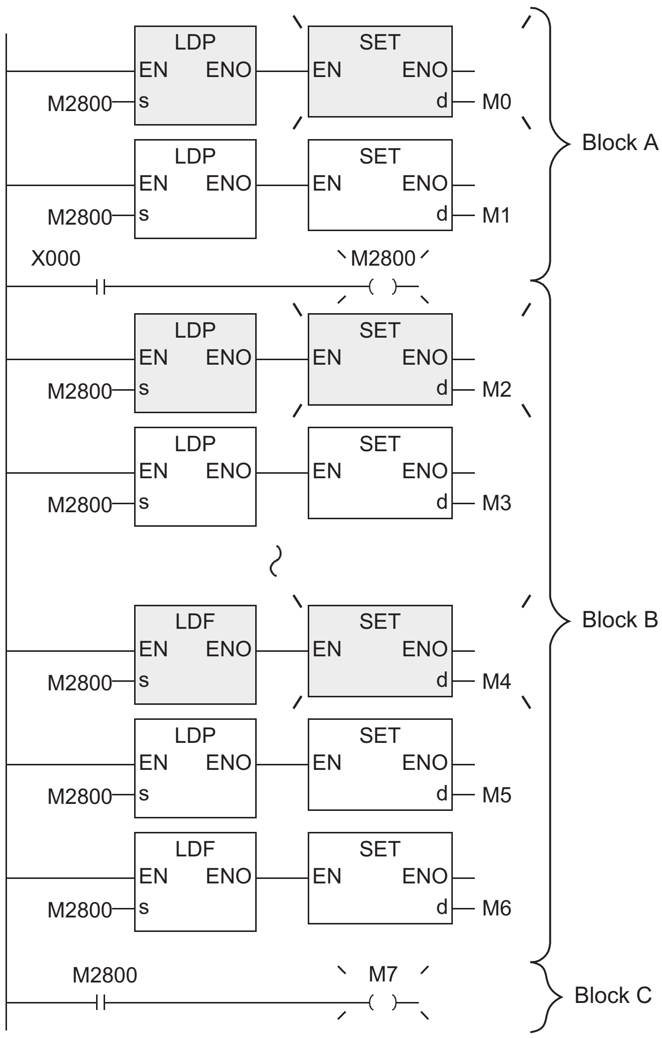
<M0 to M2799, M3072 to M7679> (M0 to M2799 for the FX2N and FX2NC PLCs)



After M0 is driven by X000, all contacts 1) to 4) corresponding to M0 are activated.

- The contacts 1) to 3) detect the rising edge of M0.
- Because of LD instruction, the contact 4) is conductive while M0 is ON.

<M2800 to M3071>

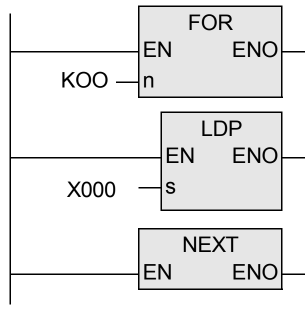


From M2800 driven by X000, the program is divided into the upper block (block A) and the lower block (block B). In each of the blocks A and B, only the first contact which detects the rising or falling edge is activated.
Because of LD instruction, the contact in the block C is conductive while M2800 is ON.
By utilizing these characteristics, "transition of state by same signal" in a step ladder circuit can be efficiently programmed.

Cautions

1) When LDP, LDF, ANDP, ANDF, ORP or ORF instruction programmed in a same step is executed two or more times within one operation cycle, the operation is as follows.

Programs executed two or more times
- Program between FOR and NEXT instructions
- Program which executes a same subroutine program from two or more CALL instructions during one operation cycle.
- Program which jumps to a label (P) in a smaller step number by CJ instruction.



Operation
- When a device turns ON from OFF
  1st time :LDP, ANDP or ORP instruction turns ON.
  2nd time and later :When the device status is same as the time when the instruction was executed last, the instruction turns OFF.
- When a device turns OFF from ON
  1st time :LDF, ANDF or ORF instruction turns ON.
  2nd time and later :When the device status is same as the time when the instruction was executed last, the instruction turns OFF.

2) When write during RUN is completed for a circuit including an instruction for falling edge pulse (LDF, ANDF, or ORF instruction), the instruction for falling edge pulse is not executed without regard to the ON/OFF status of the target device of the instruction for falling edge pulse.
When write during RUN is completed for a circuit including an instruction for falling edge pulse (PLF instruction), the instruction for falling edge pulse is not executed without regard to the ON/OFF status of the operation condition device.
It is necessary to set to ON the target device or operation condition device once and then set it to OFF for executing the instruction for falling edge pulse.

3) When write during RUN is completed for a circuit including an instruction for rising edge pulse, the instruction for rising edge pulse is executed if a target device of the instruction for rising edge pulse or the operation condition device is ON.
Target instructions for rising edge pulse: LDP, ANDP, ORP and pulse operation type applied instructions (such as MOVP)

Contact ON/OFF status (while write during RUN is executed)	Instruction for rising edge pulse	Instruction for falling edge pulse
OFF	Not executed	Not executed
ON	Executed*1	Not executed

*1. PLS instruction is not executed.

4) Some restrictions to applicable devices
▲1: The FX3U and FX3UC PLCs only are applicable.

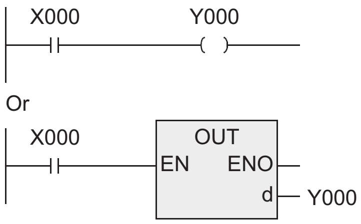
5.3 OUT (Excluding timers and counters)

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction outputs the operation result up to the execution of the OUT instruction to the specified device.

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
OUT	Continuous		OUT(EN,d); Or an assignment statement **Example:** OUT(X000,Y000); **When using an assignment statement.** Y000:=X000;

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	(d)	Target variable	ANY_SIMPLE

3. Applicable devices

Instruction	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
OUT		●	●			●	▲1												▲2					

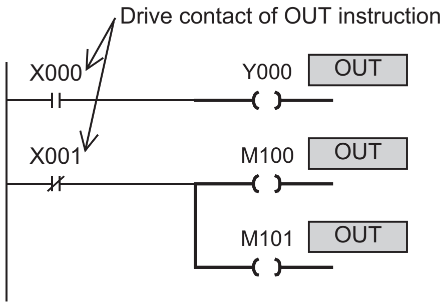
▲: Refer to "Cautions".

Function and operation explanation

1. When a bit device is used

A device described in OUT instruction turns ON or OFF according to the driven contact status.
Parallel OUT instructions can be used consecutively as many times as necessary.
In the program example shown below, OUT M100 and OUT M101 are parallel.
If two or more OUT instructions are executed for a same device number, however, the double output (double coil) operation is resulted.

[Structured ladder]



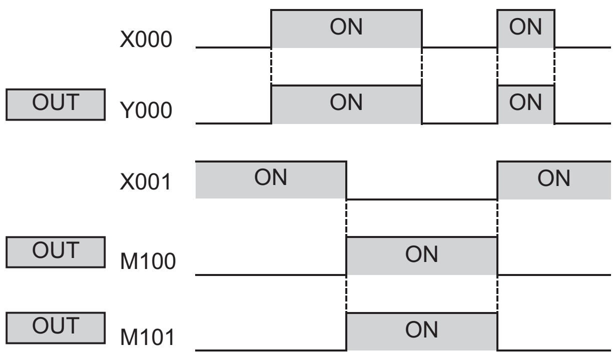
[ST]

```
OUT(X000,Y000);
IF NOT X001 THEN
   OUT(TRUE,M100);
   OUT(TRUE,M101);
END_IF;
```

For assignment statement

```
Y000:= X000;
M100:= NOT X001;
M101:= NOT X001;
```

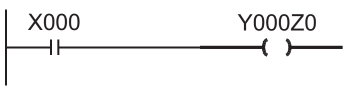
timing chart



2. Indexing

Devices used in OUT instruction can be indexed with index registers (V and Z).
(State relays (S), special auxiliary relays (M), or "D□.b" cannot be indexed.)
Applicable only to the FX3U and FX3UC PLCs.

[Structured ladder]



[ST]

```
OUT(X000,Y000Z0);
```

For assignment statement

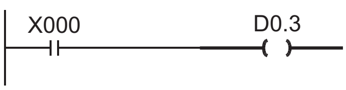
```
Y000Z0:= X000;
```

When a used devices is an input (X) or output (Y), the value of an index register (V or Z) is converted into an octal number, and then added.
Example: When the value of Z0 is "20", Y024 turns ON or OFF.

3. Bit specification of data register (D)

A bit in data register (D) can be specified as a device used in OUT instruction.
Applicable only to the FX3U and FX3UC PLCs.

[Structured ladder]



[ST]

```
OUT(X000,D0.3);
```

For assignment statement

```
D0.3:= X000;
```

When specifying a bit in data register, input "." after a data register (D) number, and then input a bit number (0 to F) consecutively.
Only 16-bit data resister is applicable.
Specify a bit number as "0 1, 2, ..., 9, A, B, ..., F" from the least significant bit.
Example: In the example shown on the left, the bit 3 of D0 turns ON or OFF when X000 turns ON or OFF.

Cautions

1) Some restrictions to applicable devices
   ▲1: The FX3U and FX3UC PLCs only are applicable.
   ▲2: Only the FX3U and FX3UC PLCs are capable of indexing applicable devices.
   The following devices cannot be indexed.
   - Special auxiliary relays (M)
   - State (S)
   - Word bit specification "D□.b"

2) The following instructions are used to operate the timer and counter in a structured program. Note that they are not operable in the OUT instruction.

Instruction name	Reference
OUT_T	Section 5.4.1
OUT_C	Section 5.5.1
OUT_C_32	Section 5.5.1

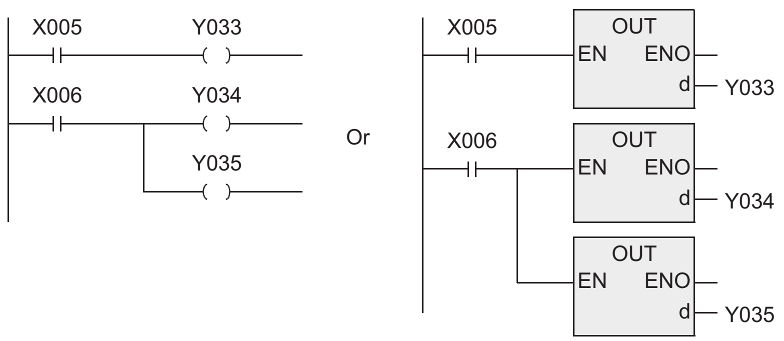
Errors

1) When a Y number used in OUT instruction does not exist due to indexing, M8316 (Non-existing I/O specification error) turns ON. (Applicable to the FX3U and FX3UC PLCs only)
2) When the device number of a device (M,T,C)other than I/O does not exist due to indexing, an operation error (error code: 6706) occurs. (Applicable to the FX3U and FX3UC PLCs only.)

Program example

1. When using bit device

[Structured ladder]

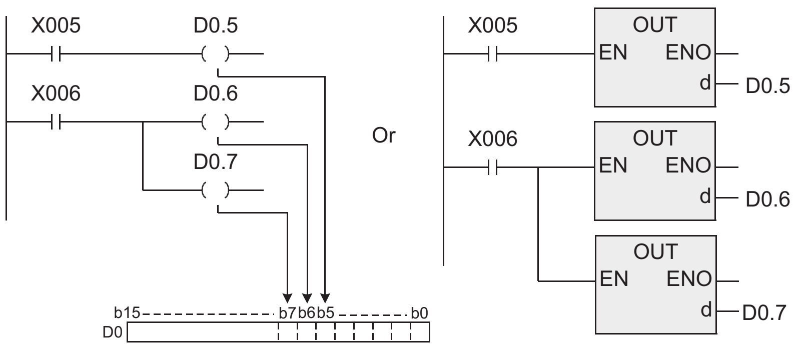


[ST]

```
OUT(X5,Y33);
IF X6 THEN
   OUT(TRUE,Y34);
   OUT(TRUE,Y35);
END_IF;
```

2. When specifying bit of word device

[Structured ladder]



[ST]

```
OUT(X5,D0.5);
IF X6 THEN
   OUT(TRUE,D0.6);
   OUT(TRUE,D0.7);
END_IF;
```

5.4 Operating Timer

5.4.1 OUT_T

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

An output is generated when a set time expires.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
OUT_T	16 bits	Continuous	OUT_T EN ENO TCoil TValue	OUT_T(EN, TCoil, TValue);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	TCoil	Target timer	Bit
	TValue	Timer set value	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
TCoil				●																				
TValue														●	▲1				▲2	●				

▲: Refer to "Cautions".

Function and operation explanation

1. OUT_T operation

1) When the operation result up to the OUT_T operation is ON, the timer coils is ON and counts until the set value is reached. When the set time expires (or reaches the set count), the contacts become as follows:

NO (normally open) contact	Timer is conductive.
NC (normally closed) contact	Timer is not conductive.

2) When the operation result up to the OUT_T operation turns OFF from ON, the timer parameters become as follows.

Timer type	Timer coil	Current timer value	Before time-up		After time-up	
			NO (normally open) contact	NC (normally closed) contact	NO (normally open) contact	NC (normally closed) contact
100 ms timer 0.1 to 3276.7 seconds	OFF	0	Nonconductive	Conductive	Nonconductive	Conductive
10 ms timer 0.01 to 327.67 seconds						
1 ms timer*1 0.001 to 32.767 seconds						
100 ms integrating timer*2 0.1 to 3276.7 seconds	OFF	Holding current value	Nonconductive	Conductive	Nonconductive	Conductive
1 ms integrating timer*2 0.001 to 32.767 seconds						

*1. Not supported by the FX2N, FX2NC, FX1N, FX1NC, FXU, FX2C, FX0 or FX0S PLC.

*2. Not supported by the FX1S, FX0N, FX0 or FX0S PLC.

2. Clearing integrating timer

After the set time expires, the current value of the integrating timer is cleared and the contacts are turned OFF by the RST.

3. Timer set value

The set value can be specified directly by a decimal number (K) or indirectly using a data register (D) or extension register (R).
Indirect setting by the extension register (R) is applicable only to the FX3U and FX3UC PLCs.
No negative numbers (-32768 to -1) can be set.
If the timer value is set to "0", the time expires at the same time as the OUT_T activates.

4. OUT_T operation

The following processes take place when the OUT_T activates.

1) The OUT_T TC coil turns ON or OFF.
2) The OUT_T TS contacts turn ON or OFF.
3) The OUT_T TN current value is changed.

If the OUT_T is skipped by an instruction such as JMP while the OUT_T is ON, neither the current value is updated nor contacts are turned ON or OFF.
When one particular OUT_T operates more than once within the same scan, the current value is updated as many times as the timer operates.

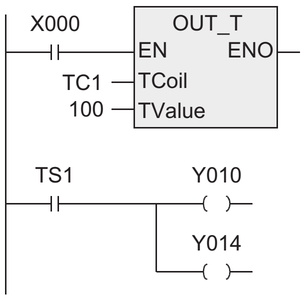
Cautions

1) When a timer device is specified in a program, use the following depending on the locations of use.
   - Used as contacts: TS
   - Used as a coil: TC
   - Used as a current value: TN
2) Use the timer T192 to T199 within a subroutine or interrupt routine. This timer counts the time when executing a coil instruction or END instruction.
   When the set value is reached, the output contact operates when the coil instruction or the END instruction is executed.
   A general purpose timer counts the time only when the coil instruction is executed. Such a timer does not operate normally because it does not count the time if used in a subroutine or an interrupt routine where the coil instruction is executed only under certain conditions.
3) Note: If a 1 ms integrating timer is used in a subroutine or an interrupt routine, the output contact operates when the first coil instruction is executed after the set value is reached
   (FX3U, FX3UC, FX3G, FX1N, FX2N, FX1NC, FX2NC, FX2C and FXU PLCs)
4) Some restrictions to applicable devices
   ▲1: Applicable only to the FX3U, FX3UC and FX3G PLCs.
   ▲2: The target device can be indexed only by the FX3U and FX3UC PLCs.

Program example

1. Program that turns ON Y010 and Y014 in 10 seconds after X000 turns ON.

[Structured ladder]

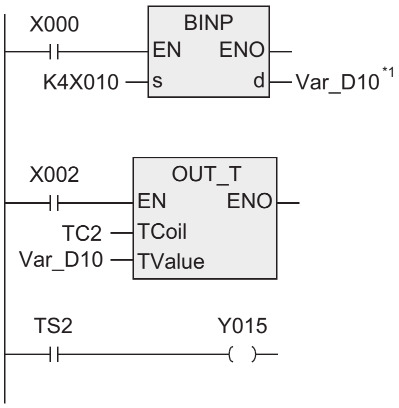


[ST]

```
OUT_T(X000,TC1,100);
OUT(TS1,Y010);
OUT(TS1,Y014);
```

2. Program that sets the BCD data of X010 to X01F to a timer.

[Structured ladder]



BCD data of X010 to X01F is converted to binary data and stored in D10.

When X002 turns ON, timer starts to count using data stored in D10 as set value.

When T2 timer completes counting up, Y015 turns on.

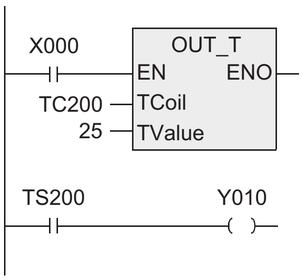
[ST]

```
BINP(X000,K4X010,Var_D10);
OUT_T(X002,TC2,Var_D10);
OUT(TS2,Y015);
```

*1. Var_D10 is a global label and is defined as D10.

3. Program that turns ON Y010 in 250 milliseconds after X000 turns ON.

[Structured ladder]



[ST]

```
OUT_T(X000,TC200,25);
OUT(TS200,Y010);
```

5.5 Operating Counters

5.5.1 OUT_C, OUT_C_32

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

The counter starts counting when the condition turns ON from OFF. It generates an output when counting up to the set value.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
OUT_C	16 bits	Continuous	OUT_C EN ENO CCoil CValue	OUT_C(EN,CCoil,CValue);
OUT_C_32	32 bits	Continuous	OUT_C_32 EN ENO CCoil CValue	OUT_C_32(EN,CCoil,CValue);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	CCoil	Target counter	Bit	
	CValue	Counter set value	ANY16	ANY32
Output variable	ENO	Execution state	Bit	

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others					
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
CCoil					●																			
CValue														●	▲1				▲2	●				

▲: Refer to "Cautions".

Function and operation explanation

1. OUT_C operation

1) When the operation result up to the OUT_C turns ON from OFF, the counter counts up the current value (count value) by +1. When the counter completes counting (the current value reaches the set value), the contact becomes as follows.

NO (normally open) contact	Conductive
NC (normally closed) contac	Nonconductive

2) The counter does not count if the operation result remains ON. (The count input does not need to be in the form of pulse.)

2. Counter reset

After completing to count, the count value and contact condition does not change until the RST is executed.

3. Counter set value

The set value of the counter can be specified directly by a decimal number (K) or indirectly using a data register (D) or extension register (R).
Indirect setting by the extension register (R) is applicable only to the FX3U and FX3UC PLCs.
No negative numbers (-32768 to -1) can be set.
If set to "0", the same process as 1 takes place.

4. When using counter device

When a counter device is specified in a program, use the following depending on the locations of use.

- Used as contacts: CS
- Used as a coil: CC
- Used as a current value: CN

Cautions

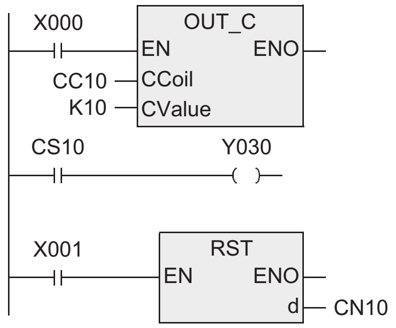
1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

2) Some restrictions to applicable devices
▲1: Applicable only to the FX3U, FX3UC and FX3G PLCs.
▲2: Only the FX3U and FX3UC PLCs can index the target device.
A 32-bit counter cannot be indexed.

Program example

1. **This program turns ON Y30 when X0 turns ON 10 times and resets the counter when X1 turns ON.**

[Structured ladder]

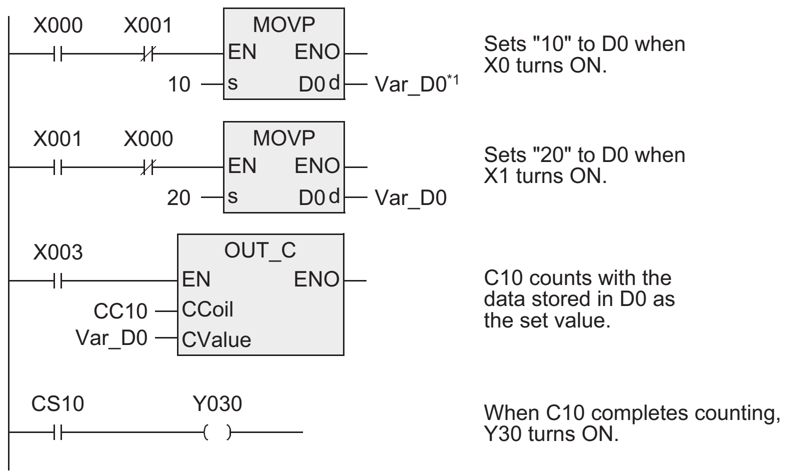


[ST]

```
OUT_C(X0,CC10,10);
OUT(CS10,Y30);
RST(X1,CN10);
```

2. **This program sets "10" to C10 when X0 turns ON and sets to "20" to C10 when X1 turns ON.**

[Structured ladder]



[ST]

```
MOVP(X0 AND NOT X1,10,Var_D0);
MOVP(X1 AND NOT X0,20,Var_D0);
OUT_C(X3,CC10,Var_D0);
OUT(CS10,Y30);
```

*1. Var_D10 is a global label and is defined as D10.

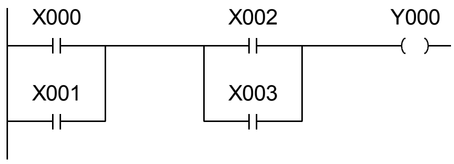
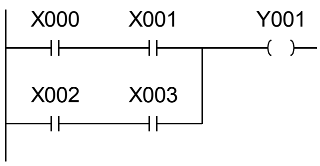
5.6 AND(...), OR(...)

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

Use AND (...) instruction to connect a branch circuit (parallel circuit block) to the preceding circuit in series.
Use OR (...) instruction to connect a series circuit block in parallel.

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language	
		Structured ladder	ST
AND(...)	Continuous		AND(...) The ladder diagram (or LD) is described as follows. Y000:=(X000 OR X001) AND(X002 OR X003);
OR(...)	Continuous		OR(...) The ladder diagram (or LD) is described as follows. Y001:=(X000 AND X001) OR(X002 AND X003);

2. Applicable devices

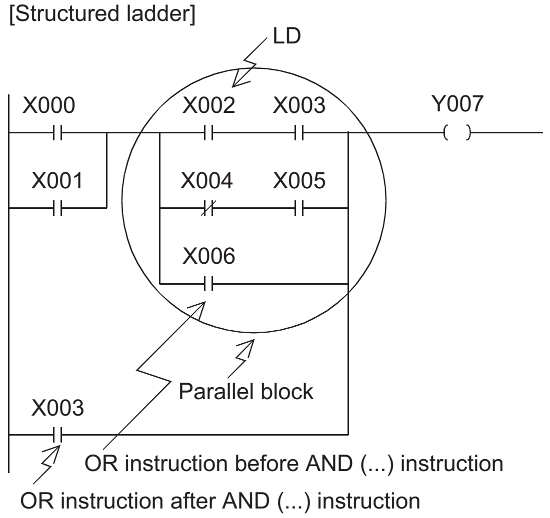
Instruction	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
AND(...)	There are no applicable devices.																							
OR(...)																								

Function and operation explanation

1. AND(...)(Serial connection of circuit blocks)

AND (...) is an independent instruction not associated with any device number in the same way as the OR (...) instruction described later.
When there are many parallel circuits, the AND (...) instruction can be used for each circuit block to connect them.

[Structured ladder]

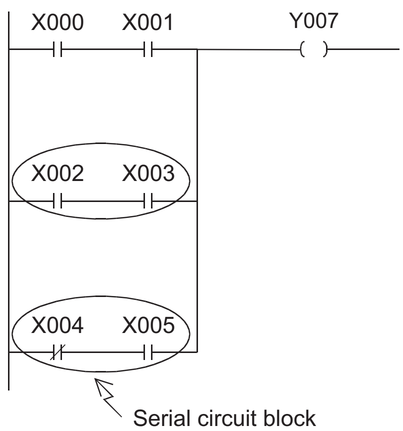


[ST]
Y007:= ((X000 OR X001) AND ((X002 AND X003) OR (NOT X004 AND X005) OR X006)) OR X003;

2. OR(...)(Parallel connection of circuit blocks)

OR (...) is an independent instruction not associated with any device number in the same way as the AND (...) instruction.
When there are many parallel circuits, the OR (...) instruction can be used for each circuit block to connect them.

[Structured ladder]



[ST]
Y007:=(X000 AND X001) OR (X002 AND X003) OR (NOT X004 AND X005);

5.7 MPS, MRD, MPP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

These PLCs have 11 memories called "Stack" which store the intermediate result (ON or OFF) of operations.

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language	
		Structured ladder	ST
MPS	Continuous	MPS EN ENO	MPS(EN);
MRD	Continuous	MRD EN ENO	MRD(EN);
MPP	Continuous	MPP EN ENO	MPP(EN);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	MPS: Bit MRD, MPP: Always TRUE
Output variable	ENO	Execution state	Bit

3. Applicable devices

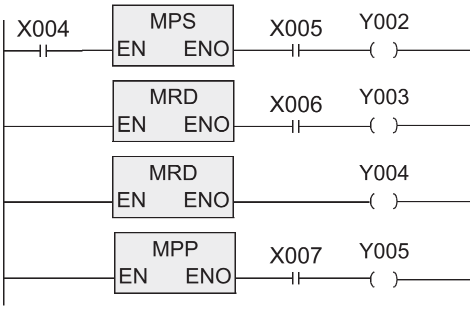
Instruction	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
MPS	There are no applicable devices.																							
MRD																								
MPP																								

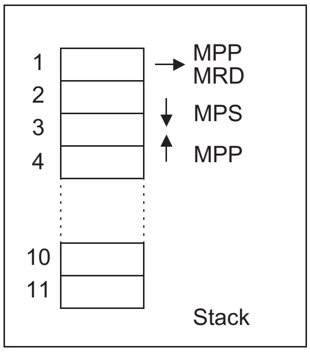
Function and operation explanation

These instructions are convenient in programming branched multi-output circuits.

1. MPS, MRD, MPP (Stack push down, stack read and stack popup)

[Structured ladder]





[ST]

```
Y002:= X005 AND MPS(X004);
Y003:= MRD(TRUE) AND X006;
Y004:= MRD(TRUE);
Y005:= MPP(TRUE) AND X007;
```

1) Use MPS instruction to store the intermediate result of operation, and then drive the output Y002.
2) Use MRD instruction to read the stored data, and then drive the output Y003.
   MRD instruction can be programmed as many times as necessary.
3) In the final output circuit, use MPP instruction instead of MRD instruction.
   MPP instruction reads the stored data described above, and then resets it.

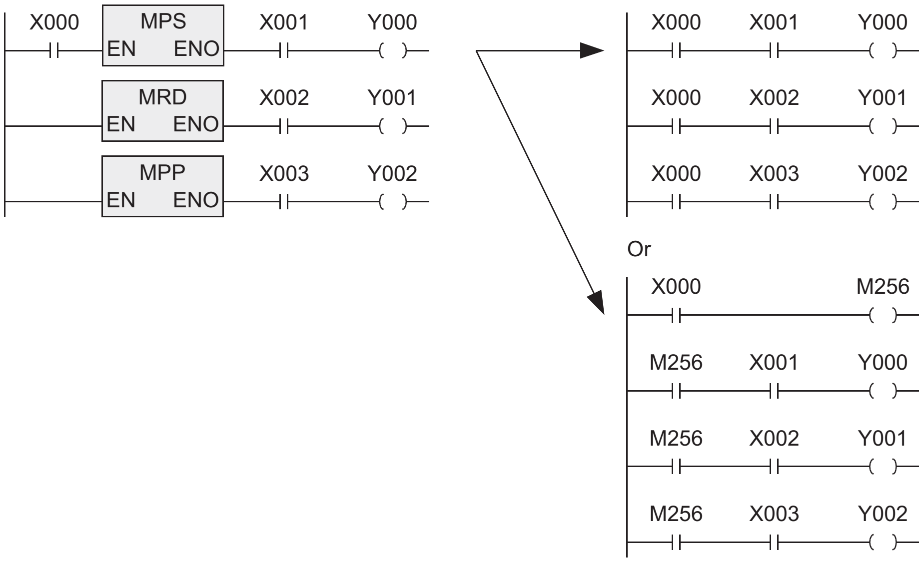
Error

MPS instruction can be used two or more times. However, the difference between the number of MPS instructions and the number of MPP instructions should be 11 or less, and should be 0 at the end.

Caution

When a circuit is programmed as shown on the left, it is compiled in fact as the program on the right that does not use MPS, MRD or MPP instruction.

Program representing actual operation
(program in the PLC)

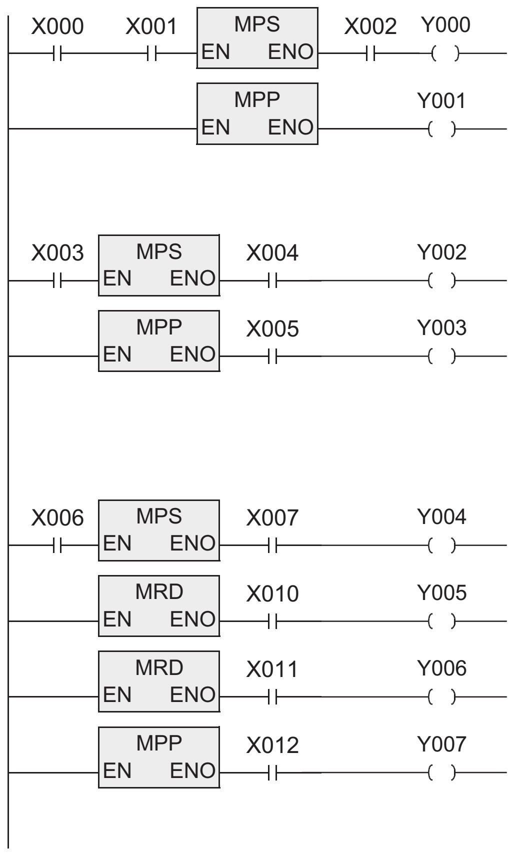


Program example

1. Program example 1 (One stack)

Only one stack is used in this example.

[Structured ladder]



[ST]

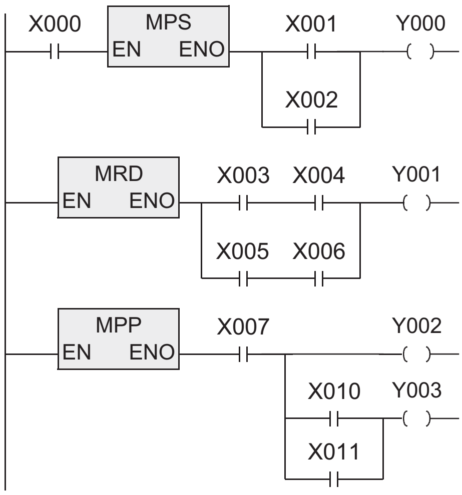
```
Y000:= MPS(X000 AND X001) AND X002;
Y001:= MPP(TRUE);

Y002:= MPS(X003) AND X004;
Y003:= MPP(TRUE) AND X005;

Y004:= MPS(X006) AND X007;
Y005:= MRD(TRUE) AND X010;
Y006:= MRD(TRUE) AND X011;
Y007:= MPP(TRUE) AND X012;
```

2. Program example 2 (One stack with AND (...) and OR (...) instructions)

[Structured ladder]

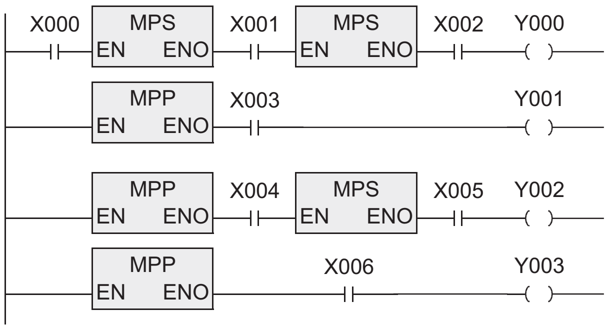


[ST]

```
Y000:= MPS(X000) AND (X001 OR X002);
Y001:= MRD(TRUE) AND ((X003 AND X004) OR (X005 AND X006));
Y002:= MPP(TRUE) AND X007;
Y003:= Y002 AND (X010 OR X011);
```

3. Program example 3 (Two stacks)

[Structured ladder]

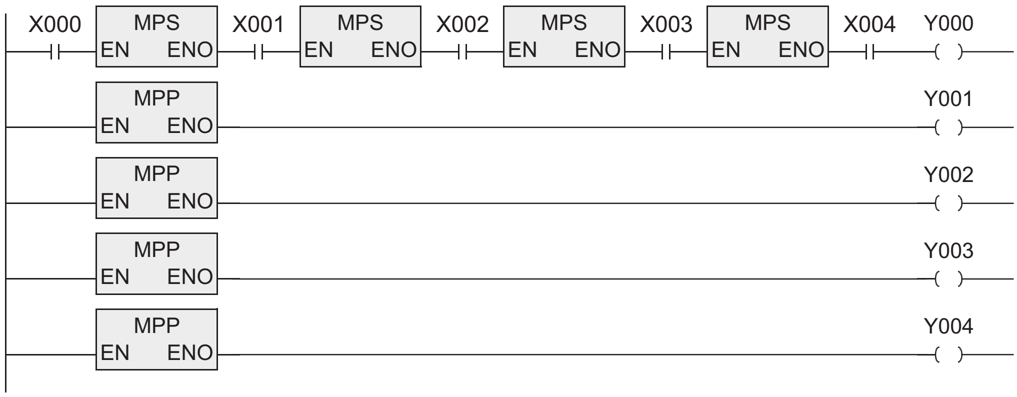


[ST]

```
Y000:= (MPS(X000) AND MPS(X001)) AND X002;
Y001:= MPP(TRUE) AND X003;
Y002:= (MPP(TRUE) AND MPS(X004)) AND X005;
Y003:= MPP(TRUE) AND X006;
```

4. Program example 4 (Four stacks)

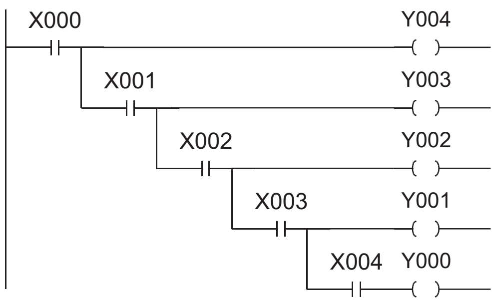
[Structured ladder]



[ST]

```
Y000:= (((MPS(X000) AND MPS(X001)) AND MPS(X002)) AND MPS(X003)) AND X004;
Y001:= MPP(TRUE);
Y002:= MPP(TRUE);
Y003:= MPP(TRUE);
Y004:= MPP(TRUE);
```

[Structured ladder]



[ST]

```
Y004:= X000;
Y003:= Y004 AND X001;
Y002:= Y003 AND X002;
Y001:= Y002 AND X003;
Y000:= Y001 AND X004;
```

In programming a circuit on the upper side, it is necessary to use MPS instruction three times.
By changing the circuit on the upper side into the circuit on the lower side, the same contents can be programmed easily without MPS instruction.

5.8 INV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

INV instruction inverts the operation result up to just before INV instruction.

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language	
		Structured ladder	ST
INV	Continuous	INV EN ENO	INV(EN);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit

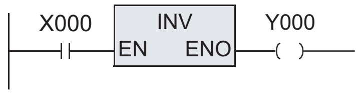
3. Applicable devices

Instruction	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
INV	There are no applicable devices.																							

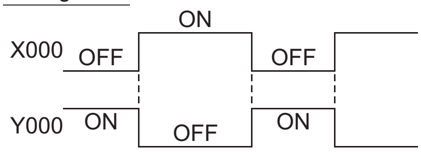
Function and operation explanation

1. INV(inverts the result of operations)

[Structured ladder]



Timing chart



[ST]

Y000:= INV(X000)

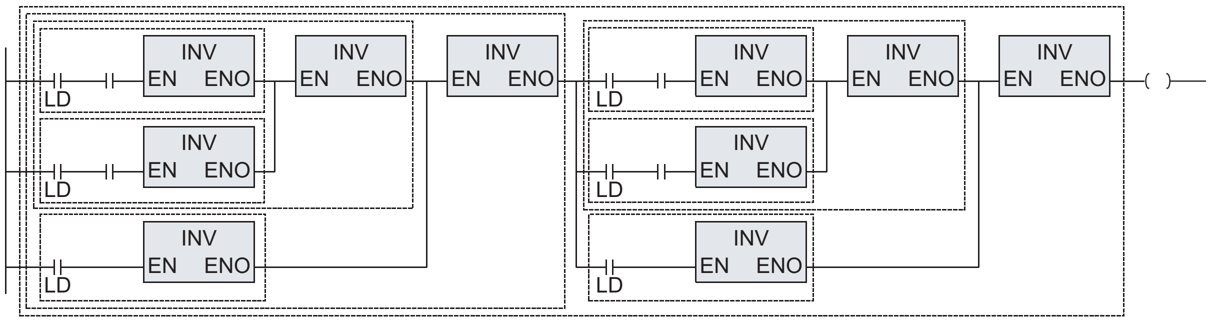
Operation result until just before INV instruction		Operation result after INV instruction is executed
OFF	→	ON
ON	→	OFF

Inverted

In the figure above, Y000 turns ON when X000 is OFF, and Y000 turns OFF when X000 is ON.
INV instruction can be used in a same position as serial contact instructions (AND, ANI, ANDP and ANDF). Different from LD, LDI, LDP and LDF instructions shown in the list, INV instruction cannot execute connection to bus lines. Different from OR, ORI, ORP and ORF instructions, INV instruction cannot be used independently in parallel to a contact instruction.

2. Operation range of INV instruction

When INV instruction is used in a complicated circuit containing ORB and ANB instructions, the operation range of INV instruction is as shown in the figure below:



INV instruction inverts the operation result after LD, LDI, LDP or LDF instruction located before INV instruction.
Accordingly, if INV instructions are used inside ORB and ANB instructions, blocks after LD, LDI, LDP or LDF instruction seen from each INV instruction are regarded as the target of INV operation.

5.9 MEP, MEF

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	○	×	×	×	×	×	×

Outline

MEP and MEF commands are instructions that change the operation results to pulses so that device numbers do not have to be specified.

1) MEP
The operation results up to the MEP instruction become conductive when the driving contacts turn ON from OFF.
The use of MEP instructions simplifies the process of changing driving contacts to pulses when multiple contact points connect in a series.

2) MEF
The operation results up to the MEF instruction become conductive when the driving contacts turn OFF from ON.
The use of MEF instructions simplifies the process of changing driving contacts to pulses when multiple contact points connect in a series.

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language	
		Structured ladder	ST
MEP	Pulse	MEP EN ENO	MEP(EN);
MEF	Pulse	MEF EN ENO	MEF(EN);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit

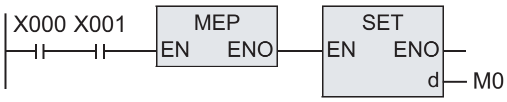
3. Applicable devices

Instruction	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
MEP	There are no applicable devices.s																							
MEF																								

Function and operation explanation

1. MEP(ON during rising edge of driving contacts results)

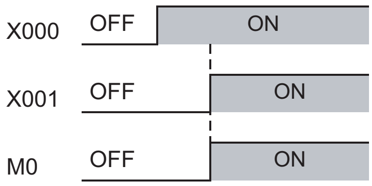
[Structured ladder]



[ST]

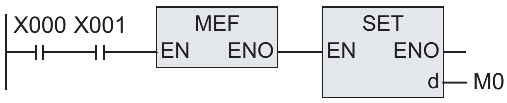
```
IF X000 AND X001 THEN
    MEP(TRUE);
    SET(TRUE,M0);
END_IF;
```

Timing chart



2. MEF(ON during falling edge of driving contacts results)

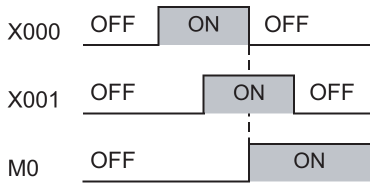
[Structured ladder]



[ST]

```
IF X000 AND X001 THEN
    MEF(TRUE);
    SET(TRUE,M0);
END_IF;
```

Timing chart



Cautions

1) The FX3U and FX3UC PLCs of V2.30 or later support MEP and MEF instructions.
2) MEP and MEF instructions may not operate normally if the indexed contact is modified and changed to pulses by sub-routine programs, the FOR and NEXT instructions, etc.
3) As the MEP and MEF instructions operate using the operation results immediately before them, use at the list program as the AND instruction.
The MEP and MEF instructions cannot be used at the list program as LD or OR.
4) Caution on writing during RUN
   a) Pulse command during rising edge of operation (MEP instruction) results
   After writing to the circuit with MEP instructions during RUN, the MEP instruction result turns ON (conductive) while the operation results up to the MEP instruction are ON.
   b) Pulse instruction during falling edge of operation (MEF command) results
   After writing to the circuit with MEF instructions during RUN , the MEF instruction result turns OFF (nonconductive), regardless of the operation results up to the MEF instruction. The operation results of MEF instruction turns ON (conductive) when the operation results up to the MEF instruction turn OFF.

Operation Results up to MEP/MEF Instruction (while writing is excuted during RUN)	MEP	MEF
OFF	OFF (non-conductive)	OFF (non-conductive)
ON	ON (conductive)	OFF (non-conductive)

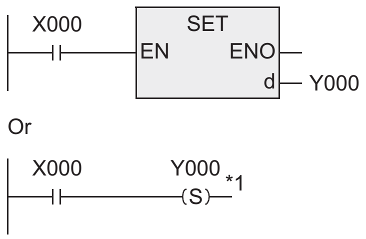
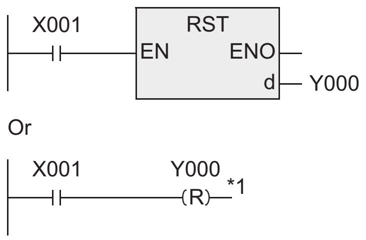
5.10 SET, RST

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

1) Setting a bit device (SET instruction [holding operation])
   When the command input turns ON, SET instruction sets to ON an output relay (Y), auxiliary relay (M), state relay (S) and bit specification of word device.
   Even if the command input turns OFF after that, the device which was set to ON by SET instruction remains ON.
2) Resetting a bit device (RST instruction [resetting folding operation])
   RST instruction resets an output relay (Y), auxiliary relay (M), state relay (S), timer (T), counter (C) or bit specification of a word device.
   Use the RST instruction to reset (reset to OFF) a device which was set to ON by SET instruction.
3) Clearing the current value of a word device (RST instruction [Clearing current value and resister])
   RST instruction clears the current value data of a timer (T), counter (C), data register (D), extension register or (R)index register (V) (Z). (The same result can be obtained by MOV instruction which transfers the constant K0.)
   RST instruction can be used also to reset the current value and return the contact of retentive type timers.
   SET and RST instructions can be used for a same device as many times as necessary in an arbitrary order.

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language	
		Structured ladder	ST
SET	Continuous		SET(EN,d); **Example:** SET(X000,Y000);
RST	Continuous		RST(EN,d); **Example:** RST(X001,Y000);

*1. This symbol is applicable to the bit type data only.

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	(d)	Applicable device or variable	SET Bit RST ANY_SIMPLE

3. Applicable devices

Instruction	Bit Devices							Word Devices													Others					
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
SET		●	●			●	▲1												▲3					
RST		●	●	●	●	●	▲1					●	●	●	▲2		●	●	▲3					

▲: Refer to "Cautions".

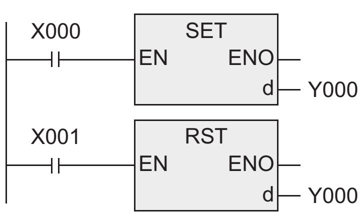
Function and operation explanation

SET instruction drives the coil for an output relay (Y), auxiliary relay (M), state relay (S) and bit specification of data register (D).

1. When using a bit device

SET instructions located in parallel can be used consecutively as many times as necessary.
In the program example shown below, RST (X1001, Y000) after SET (X000, Y000) corresponds to this usage.

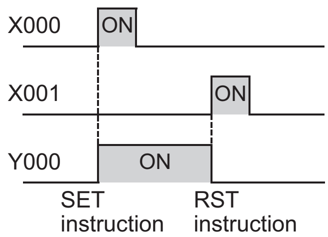
[Structured ladder]



[ST]

```
SET(X000,Y000);
RST(X001,Y000);
```

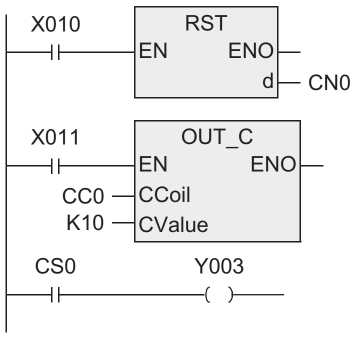
timing chart



2. When using word device (timer or counter)

Use RST instruction to reset a counter or retentive type timer.

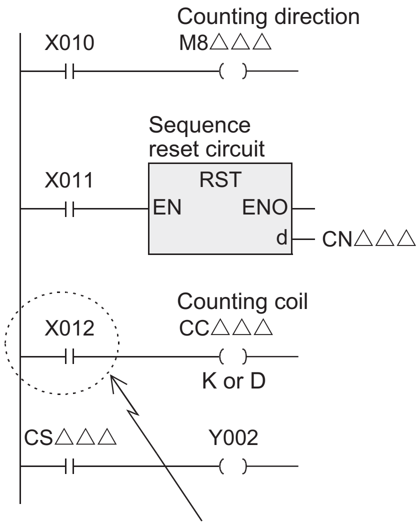
1) Program example of an internal counter



C0 up-counts the number of turning ON from OFF at X011. When the counting result reaches the set value K10, the output contact C0 is activated. Even if X011 changes from OFF to ON after that, the current value of the counter remains unchanged and the output contact remains activated.
For clearing the counter and returning the output contact, X010 is set to ON.

In case of latched (battery backed) type counters, the current value and the operation status and reset status of the output contact are latched even after power failure.

2) Program example of a high speed counter



For one-phase one-input counters, use special auxiliary relays for specifying the counting direction.
X010 in ON status: specifies down counting.
X010 in OFF status: Specifies up counting
When X011 turns ON, the output contact of the counter C△△△ is returned and the current value of the counter is reset to "0".
In counters with reset input, the same situation is achieved by interrupt operation when the corresponding reset input turns ON, but any program is not required for this operation.
When X012 turns ON, turning ON/OFF of a counting input X000 to X005 determined according to the counter number is counted.
In counters having start input, counting is started only after the corresponding start input turns ON.
When the current value of a counter increases and reaches the set value (K or contents of D), the output contact is set. When the current value decreases and reaches the set value, the output contact is reset.

As a contact driving the counting coil of a high speed counter, program a contact which is normally ON when high speed counting is executed.
If an input relay (X000 to X005) assigned for high speed counters is used for driving the counting coil, accurate counting cannot be achieved.

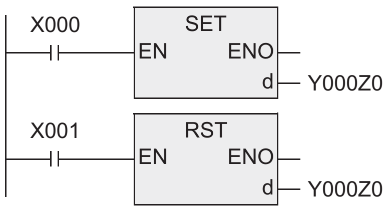
3) Caution on using RST instruction for a jumped program, subroutine program or interrupt program
When RST instruction for a timer or counter is executed in a jumped program, subroutine program or interrupt program, the timer or counter may be kept in the reset status and the timer or counter may be disabled.
For details, refer to the following sections.

→ For a jumped program, refer to subsection 7.1.1.
→ For a subroutine program, refer to subsection 7.1.2.
→ For an interrupt program, refer to subsection 8.2.3.

3. Indexing

Devices used in SET and RST instructions can be indexed with index registers (V, Z).
(State relays (S), special auxiliary relays (M), 32-bit counters, "D□.b" and word devices cannot be indexed.)
This is applicable only to the FX3U and FX3UC PLCs.

[Structured ladder]



[ST]

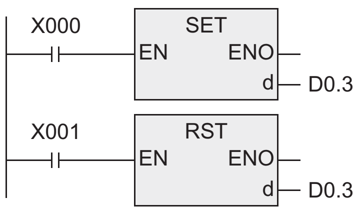
```
SET(X000,Y000Z0);
RST(X001,Y000Z0);
```

When a used device is an input (X) or output (Y), the value of an index register (V, Z) is converted into octal, and then added.
Example: When Z0 is "20", Y024 turns ON or OFF.

4. Bit specification of a data register (D)

A bit data register (D) can be specified as a device used in SET or RST instruction.
This is applicable only to the FX3U and FX3UC PLCs.

[Structured ladder]



[ST]

```
SET(X000,D0.3);
RST(X001,D0.3);
```

When specifying a bit in data register, input "." after a data register (D) number, and then input a bit number (0 to F) consecutively.
Only 16-bit data registers are available.
Specify a bit number as "0, 1,2, ..., 9, A, B, ..., F" from the least significant bit.
Example: In the example shown on the left, when X000 turns ON once, the bit 3 of D0 turns ON. When X001 turns ON, the bit 3 of D0 turns OFF.

Cautions

1) Some restrictions to applicable devices
   ▲1: The FX3U and FX3UC PLCs only are applicable.
   ▲2: The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲3: Only the FX3U and FX3UC PLCs are capable of indexing applicable devices.
   The following devices cannot be indexed.
   - Special auxiliary relays (M)
   - 32-bit counters (C)
   - State (S)
   - Word device
   - Word bit specification "D□.b"
2) When SET and RST instructions are executed for an output relay (Y) in a same operation, the result of the instruction located nearest the END instruction (which specifies the end of program) is output.
3) When using the retentive type timers of the FX1N, FX2N, FX1NC and FX2NC, be sure to create a program where the RST instruction resets the retentive type timers to be used. If no such a reset circuit by RST is present in the program, the timers remain in the state of reset, possibly causing the timers not to operate.

Error

1) When an I/O number used in SET or RST instruction does not exist due to indexing, M8316 (non-existing I/O specification error) turns ON. (Applicable only to the FX3U and FX3UC PLCs.)
2) When the device number of a device (M, T or C) other than I/O used in SET or RST instruction does not exist due to indexing, an operation error (error code: 6706) occurs. (Applicable only to the FX3U and FX3UC PLCs.)

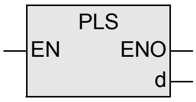
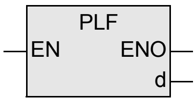
5.11 PLS, PLF

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

When PLS instruction is executed, an applicable device is activated during only one operation cycle after a drive input turns ON.
When PLF instruction is executed, an applicable device is activated during only one operation cycle after a drive input turns OFF.
For example, when PLC mode is changed in the way "RUN → STOP → RUN while a drive input remains ON, "PLS(**, M0) operates, but "PLS (**, M600) (backed up by the battery)" does not operate (when the PLC mode switches from STOP to RUN) because the status of M600 is latched even while the PLC is in the STOP mode.

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
PLS	Pulse		PLS(EN,d);
PLF	Pulse		PLF(EN,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	(d)	Applicable device or variable	Bit

3. Applicable devices

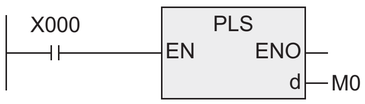
Instruction	Bit Devices: System User							Word Devices: Digit Specification				Word Devices: System User				Word Devices: Special Unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
PLS		●	▲1																▲2					
PLF		●	▲1																▲2					

▲: Refer to "Cautions".

Function and operation explanation

1. PLS (rising edge differential output)

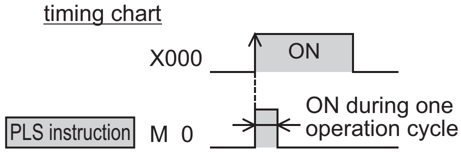
[Structured ladder]



[ST]

PLS(X000, M0);

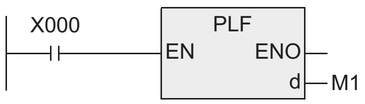
timing chart



In the figure above, M0 is ON during only one operation cycle when X000 changes from OFF to ON.

2. PLF (falling edge differential output)

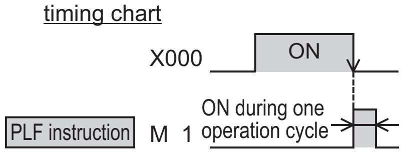
[Structured ladder]



[ST]

PLF(X000, M1);

timing chart

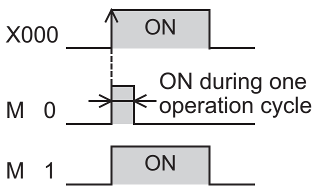


In the figure above, M1 is ON during only one operation cycle when X000 changes from ON to OFF.

3. Output drive side

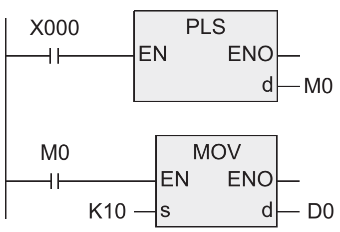
The following two circuits cause a same operation.

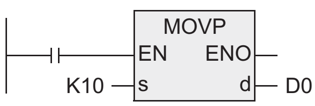
<<OUT instruction>> <<PLS instruction>>



In each case, M0 is ON during only one operation cycle when X000 changes from OFF to ON.

<<PLS instruction>> <<Pulse operation type applied instruction>>





In each case, MOV instruction is executed only once when X000 changes from OFF to ON.

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Cautions

1) When write during RUN is completed for a circuit including an instruction for falling edge pulse (LDF, ANDF or ORF instruction), the instruction is not executed without regard to the ON/OFF status of the target device of the instruction for falling edge pulse.
When write during RUN is completed for a circuit including an instruction for falling edge pulse (PLF instruction), the instruction is not executed without regard to the ON/OFF status of the operation condition device.
It is necessary to set to ON the target device or operation condition device once and then set it to OFF for executing the instruction for falling edge pulse.
2) When write during RUN is completed for a circuit including an instruction for rising edge pulse, the instruction is executed if a target device of the instruction for rising edge pulse or the operation condition device is ON.
Target instructions for rising edge pulse: LDP, ANDP, ORP, and pulse operation type applied instructions (such as MOVP)

Contact ON/OFF status (while write during RUN is executed)	Instruction for rising edge pulse	Instruction for falling edge pulse
OFF	Not executed	Not executed
ON	Executed*1	Not executed

*1. PLS instruction is not executed.

3) Some restrictions to applicable devices
▲1: Excluding special auxiliary relays (M)
▲2: Only the FX3U and FX3UC PLCs are capable of indexing applicable devices.
The following devices cannot be indexed.
   - Special auxiliary relays (M)

5.12 MC, MCR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

When MC instruction is executed, the bus line (LD or LDI point) is moved to a position after MC contact.
The bus line can be returned to the original position by MCR instruction.
By changing a device (Y or M) number, MC instruction can be used as many times as necessary.
If a same device number is used twice, however, it results in the double coil operation in the same way as OUT instruction.

1. Format and operation, execution form

Instruction name	Execution form	Expression in each language	
		Structured ladder	ST
MC	Continuous	MC block: EN → ENO, n, d	MC(EN,n,d);
MCR	Continuous	MCR block: EN → ENO, n	MCR(EN,n);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	MC: Bit MCR: Always TRUE
	(n)	Nesting level When adopting a nesting structure, use it in order of 0 → 1 → 2 → 3 → 4 → 5 → 6 → 7. If not adopting a nesting structure, it is always "0".	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Device or variable of common connection contact	Bit

3. Applicable devices

Instruction	Bit Devices							Word Devices													Others				
	System User							Digit Specification				System User				Special Unit	Index				Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier		K	H	E	"□"	P
MC		●	▲1																						
MCR	There are no applicable devices.																								

▲: Refer to "Cautions".

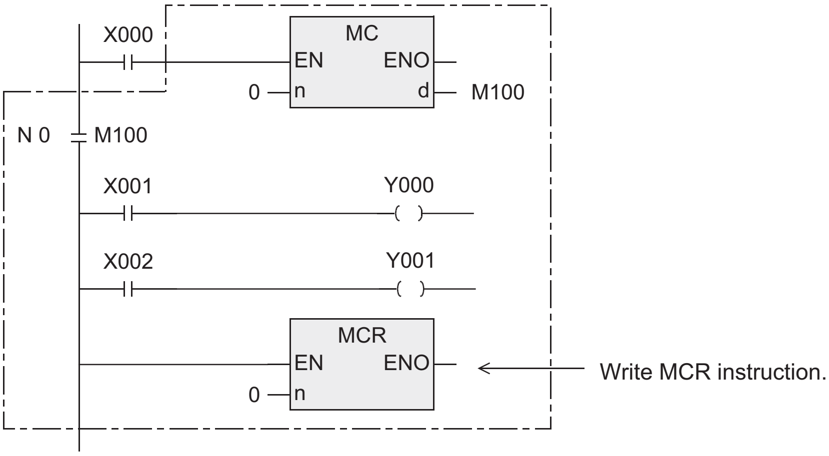
Function and operation explanation

When MC instruction is executed, the bus line is moved to a position after MC contact.
Drive instructions connected to the bus line after MC contact execute each operation only when MC instruction is executed, and do not execute the operation when MC instruction is not executed (the same operation with the contact OFF).
In the program example below, the instructions from MC to MCR are executed as they are while the input X000 is ON.
However, while the input X000 is OFF, each driven device offers the following operation.

Timers (except retentive type timers) and devices driven by OUT instruction: Turn OFF
Retentive type timers, counters and devices driven by SET/RST instruction : Hold the current status.

The expressions of circuit programs used to explain operations are circuits (for reading or monitoring) of GX Works2.

[Structured ladder]



[ST]

```
MC(X000,0,M100);
  Y000:= X001;
  Y001:= X002;
MCR(TRUE,0);
```

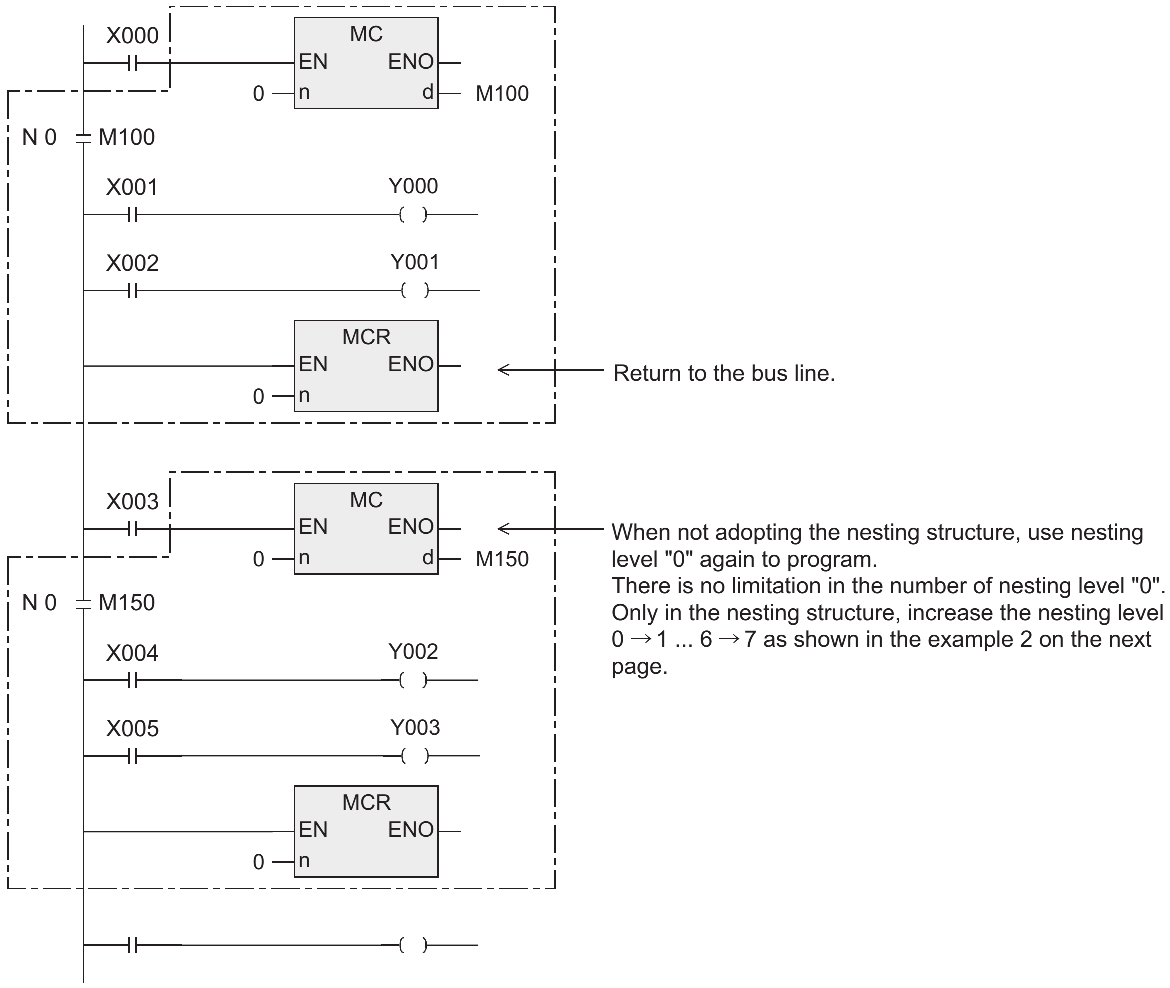
Caution

Some restrictions to applicable devices
▲1: Excluding special auxiliary relays (M)

Program examples

1. When the nesting structure is not adopted.

[Structured ladder]



[ST]

```
MC(X000,0,M100);
  Y000:= X001;
  Y001:= X002;
MCR(TRUE,0);
MC(X003,0,M150);
  Y002:= X004;
  Y003:= X005;
MCR(TRUE,0);
```

2. When the nesting structure is adopted.

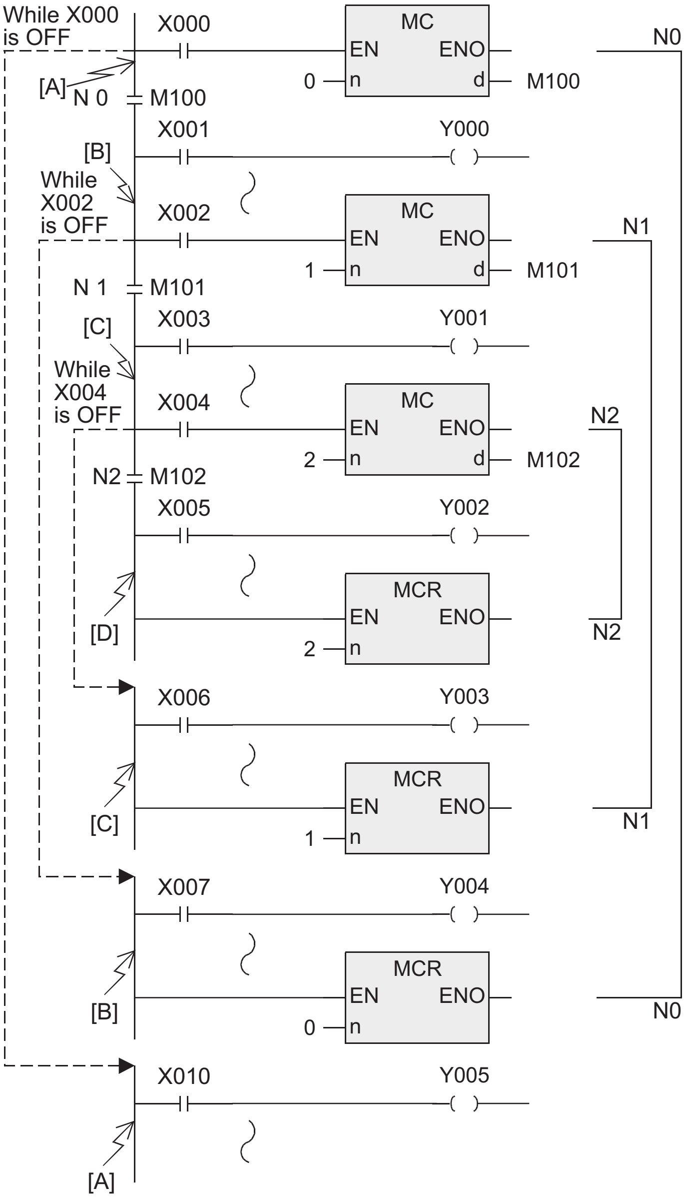
When using MC instructions inside MC instruction, increase the nesting level "N" in turn in the way "N0 → N1 → N2 → N3 → N4 → N5 → N6 → N7".

For returning from the nesting structure, reset the nesting levels from the highest one in turn using MCR instruction in the way "N7 → N6 → N5 → N4 → N3 → N2 → N1 → N0".

For example, if "MCR N5" is programmed without programming "MCR N6" and "MCR N7", the nesting level is returned to 5 at one time.

Available nesting levels are from N0 to N7 (eight layers).

[Structured ladder]



N0
The bus line B is active while X000 is ON.

N1
The bus line C is active while both X000 and X002 are ON.

N2
The bus line D is active while all of X000, X002 and X004 are ON.

N1
The bus line returns to the status of the bus line C by "MCR N2".

N0
The bus line returns to the status of the bus line B by "MCR N1".

Initial status
The bus line returns to the initial status of the bus line A by "MCR N0". Accordingly, Y005 turns ON or OFF by turning ON or OFF of X010 without regard to X000, X002 and X004.

[ST]

```
MC(X000,0,M100);
  Y000:= X001;
  MC(X002,1,M101);
    Y001:= X003;
    MC(X004,2,M102);
      Y002:= X005;
    MCR(TRUE,2);
    Y003:= X006;
  MCR(TRUE,1);
  Y004:= X007;
MCR(TRUE,0);
Y005:= X0010;
```

5.13 END

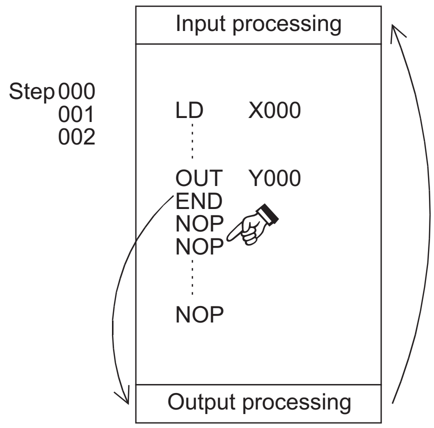
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

END instruction specifies the end of a program.
(Do not write the END instruction in the middle of a program.)
END instruction for ending a program and input/output processing and returning to 0 step is automatically written at the end of the program. It cannot be programmed into program structural elements (POU).

Function and operation explanation

PLCs repeat "input processing → program execution → output processing". When END instruction is written at the end of a program, PLCs immediately execute the output processing without executing steps after END instruction.
If END instruction is not written at the end of a program, PLCs execute the program until the final step, and then execute the output processing.
At the first execution after the PLC mode was changed from STOP to RUN, PLCs start from END instruction.
When END instruction is executed, the watchdog timer (which checks to see if the operation cycle is too long) is refreshed.



Cautions

Do not write END instruction in the middle of a program.

5.14 NOP (for simple project only)

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	×	×	×

This instruction is available for use only in the simple project. It cannot be programmed in the structured project.

6. Step Ladder Instructions

6.1 Step Ladder

This chapter introduces the instructions of structured project that correspond to the MELSEC-LD step ladder instructions.

6.1.1 Outline

In programs using step ladder instructions, a state relay S is assigned to each process based on machine operations, and input condition and output control are programmed as sequences connected to the state output.

6.1.2 Function and operation explanation

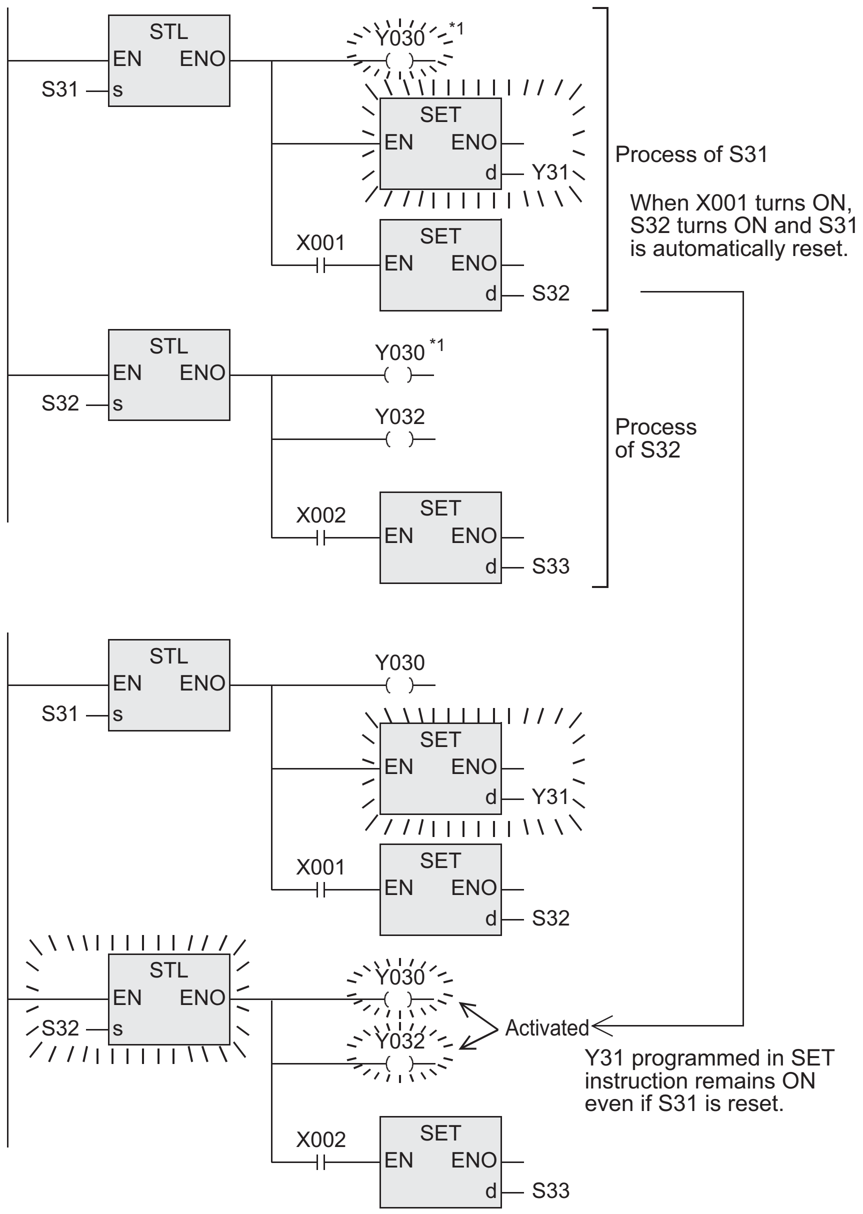
In step ladder program, a state S is regarded as one control process, and a sequence of input condition and output control are programmed in a state relay.
Because the preceding process is not performed any more when the program execution proceeds to the next process, a machine can be controlled using simple sequences for each process.

1. Operation of instruction

In a step ladder program, each process performed by the machine is expressed by a state relay.
A state relay consists of a drive coil and contact (STL output) in the same way as other relays.
Use SET or OUT instruction to drive a coil, and use STL instruction for a contact.

- When a state relay turns ON, a connected circuit (internal circuit) is activated by way of an STL output.
  When a state relay turns OFF, a connected internal circuit is deactivated by way of an STL output.
  After one operation cycle, non-driving of an instruction (jump status) is not available.
- When a condition (transfer condition) provided between state relays is satisfied, the next state relay turns ON, and the state relay which has been ON so far turns OFF (transfer operation).
  In the state relay transfer process, the both state relays are ON only instantaneously (during one operation cycle).
  In the next operation cycle after the ON status was transferred the former state is reset to OFF.
  When the transfer state relay S is used in a contact instruction, however, the contact image is executed in the OFF status immediately after the transfer condition is satisfied.

- One state relay number can be used only once.



*1. Output coils can be used again in different state relays.

2. Primary knowledge for creating programs

- List of sequence instructions available between STL instruction and RET instruction

State relay		Instruction		
		LD/LDI/LDP/LDF AND/ANI/ANDP/ANDF, OR/ORI/ORP/ORF, OUT, SET/RST, PLS/PLF	ANB/ORB/MPS/MRD/ MPP	MC/MCR
Initial/general state relay		Available	Available*1	Not available
Branch/recombination state relay	Drive processing	Available	Available*1	Not available
	Transfer processing	Available	Not available	Not available

- STL instruction cannot be used in interrupt program and subroutine programs.
- It is not prohibited to use jump instructions in state relays. But it is not recommended to use jump instructions because complicated movements will be resulted.

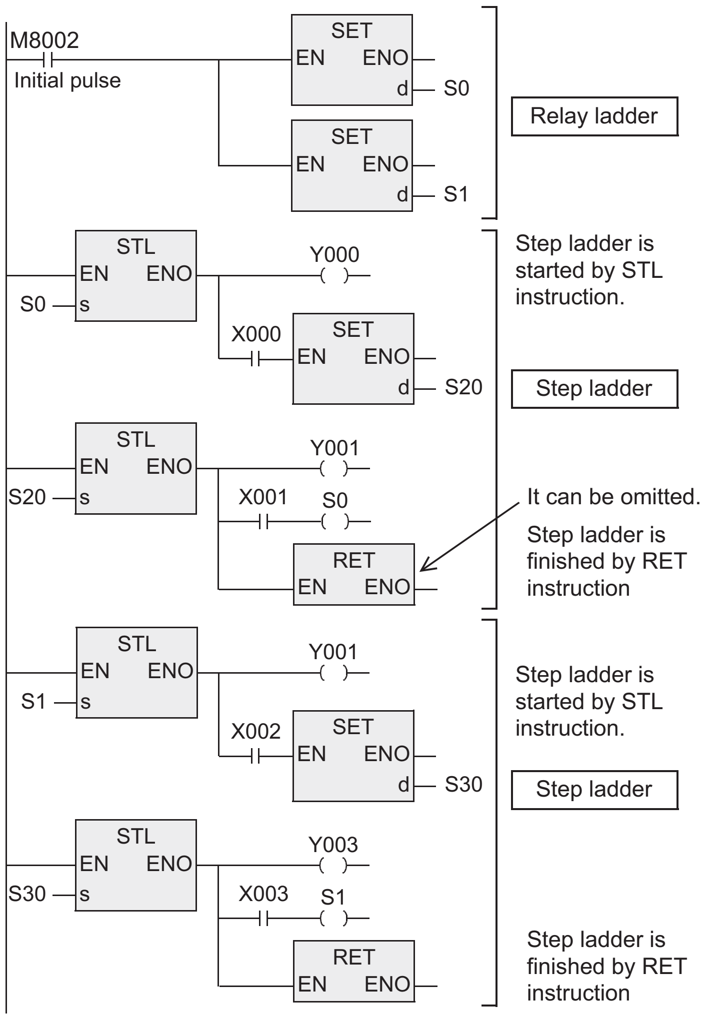
*1. MPS instruction cannot be used immediately after an STL instruction, even in a drive processing circuit.

- Special auxiliary relays
  For efficiently creating step ladder programs, it is necessary to use some special auxiliary relays. The table below shows major ones.

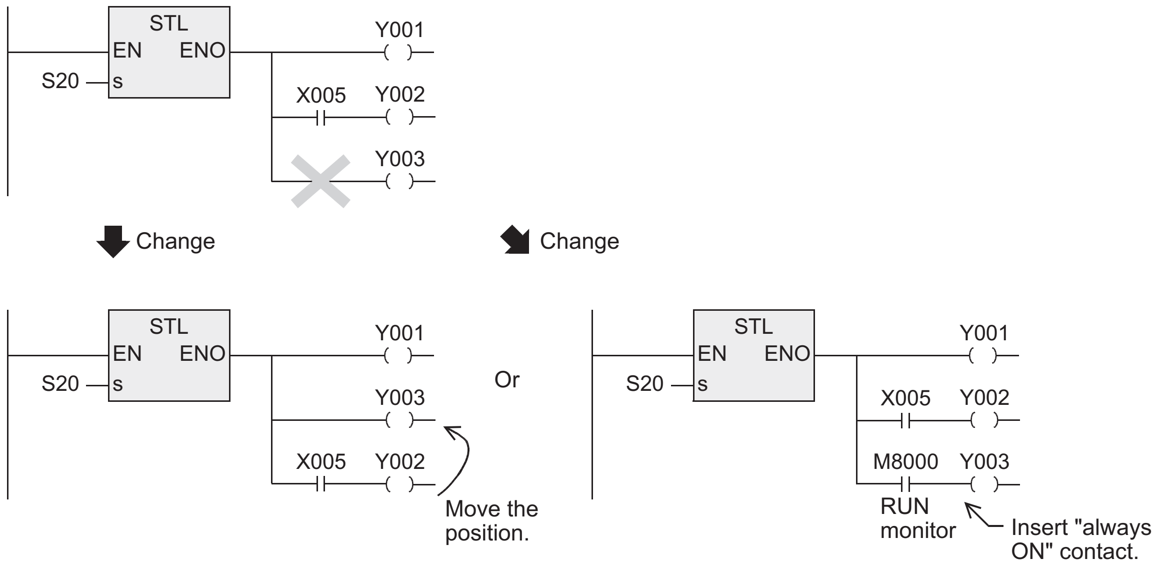
Device number	Name	Function and application
M8000	RUN monitor	This relay is normally ON while the PLC is in the RUN mode. Use this relay as the program input condition requiring the normally driven status or for indicating the PLC operation status.
M8002	Initial pulse	This relay turns ON and remains ON only instantaneously (during one operation cycle) when the PLC mode is changed from STOP to RUN. Use this relay for the initial setting of a program or for setting the initial state relay.
M8040	STL transfer disable	When this relay is set to ON, transfer of the ON status is disabled among all state relays. Because programs in state relays are operating even in the transfer disabled status, output coils do not turn OFF automatically.
M8046*1	STL state ON	This relay automatically turns ON M8046 when any of the state relays S0 to S899 or S1000 to S4095 turn ON. Use this relay to prevent simultaneous start up of another float or as a process ON/OFF flag.
M8047*1	Enable STL monitoring	When this device is driven, the state relays in the ON status on the programming function are automatically read and displayed. For details, refer to the manual of each peripheral equipment.

*1. Processed when END instruction is executed.

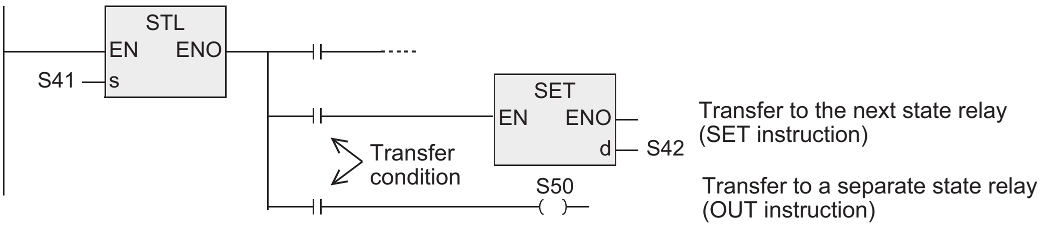
- Block
  When there are relay ladder blocks and step ladder blocks, put RET instruction at the end of each step ladder program. A PLC starts the step ladder processing by STL instruction, and returns to the relay ladder processing from the step ladder processing by RET instruction. However, when consecutively programming a step ladder in a different flow (when there is no relay ladder before the step ladder in the different flow), RET instruction between flows can be omitted, and RET instruction can be programmed only at the end of the last flow.



- Output driving method
  It is required to include a LD or LDI instruction before the last OUT instruction in a state relay.
  Change such a circuit as shown below.

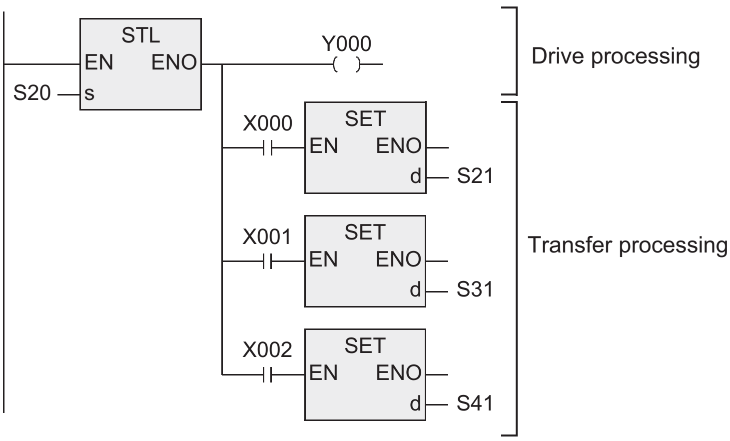


- State relay transfer method
  Each OUT and SET instruction in state relays automatically resets the transfer source, and has the self-holding function.
  OUT instructions can be used only for transfer to a separate state relay in an SFC program.

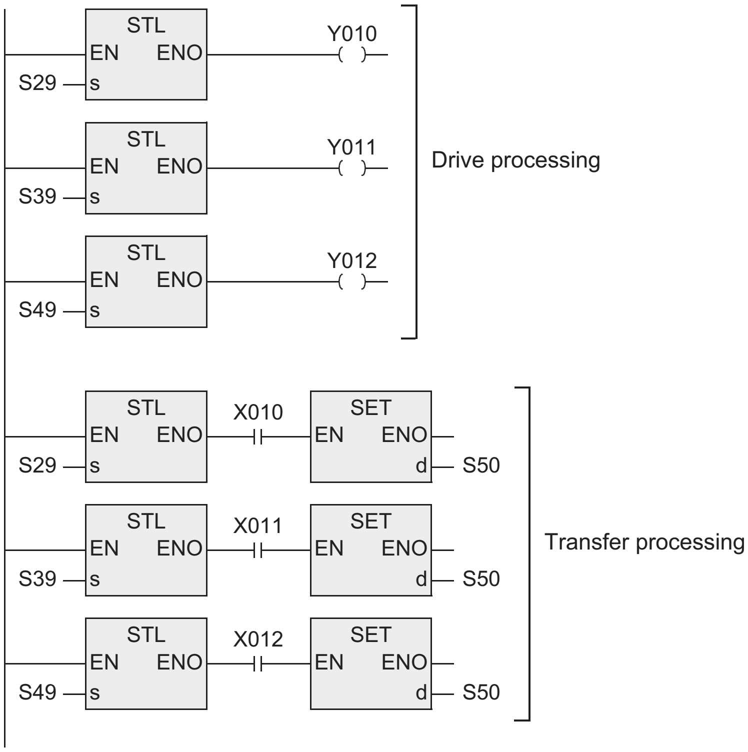


3. Program with state relays in branches and recombination

- Example of selective branch
  Do not use MPS, MRD, MPP, AND (…) and OR (…) instructions in a transfer processing program with branches and recombination.
  Even in a load driving circuit, MPS instructions cannot be used immediately after STL instructions.
  In the same way as programs for general state relays, program the drive processing first, and then program the transfer processing.
  Continuously program all transfer processing.

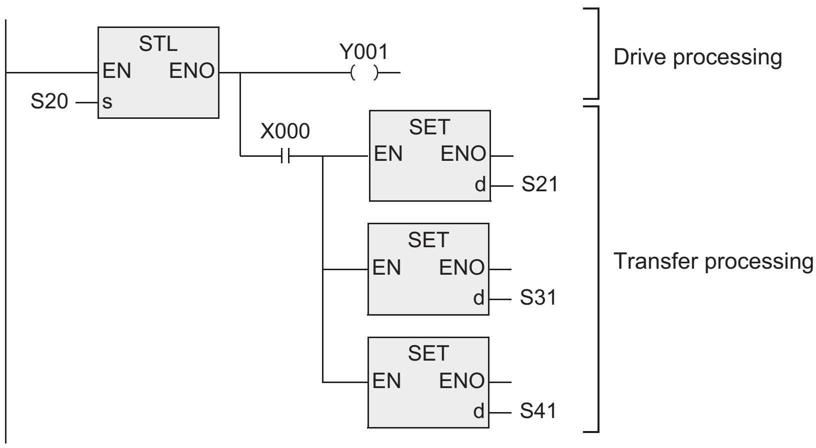


- Example of selective recombination
  Do not use MPS, MRD, MPP, AND (…) and OR (…) instructions in a transfer processing program with branches and recombination.
  Even in a load driving circuit, MPS instructions cannot be used immediately after STL instructions.
  Pay attention to the programming order so that a branch line does not cross a recombination line.

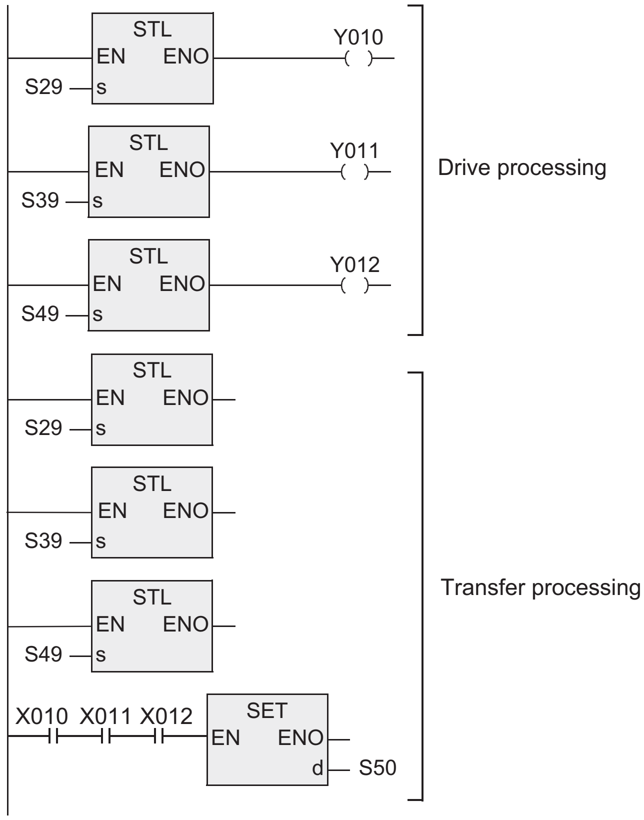


  Before recombination, program the drive processing of state relays first.
  After that, program only the transfer processing to recombination state relays continuously.
  This rule should be observed to enable inverse conversion into an SFC program.

- Example of parallel branch
  Do not use MPS, MRD, MPP, AND (…) and OR (…) instructions in a transfer processing program with branches and recombination.
  Even in a load driving circuit, MPS instructions cannot be used immediately after STL instructions.
  In the same way as programs for general state relays, program the drive processing first, and then program the transfer processing.
  Continuously program all transfer processing.



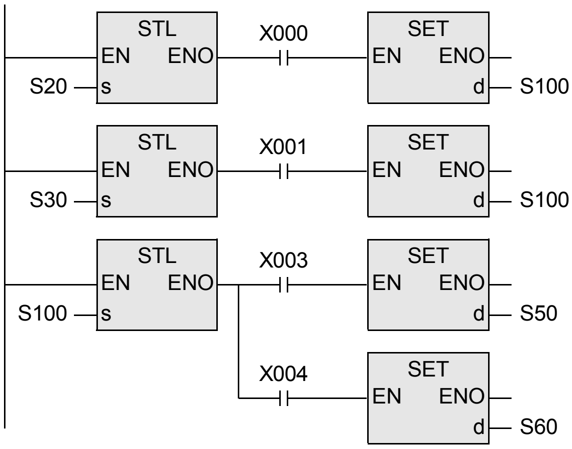
- Example of parallel recombination
  Do not use MPS, MRD, MPP, AND (…) and OR (…) instructions in a transfer processing program with branches and recombination.
  Even in a load driving circuit, MPS instructions cannot be used immediately after STL instructions.
  Pay attention to the programming order so that a branch line does not cross a recombination line.



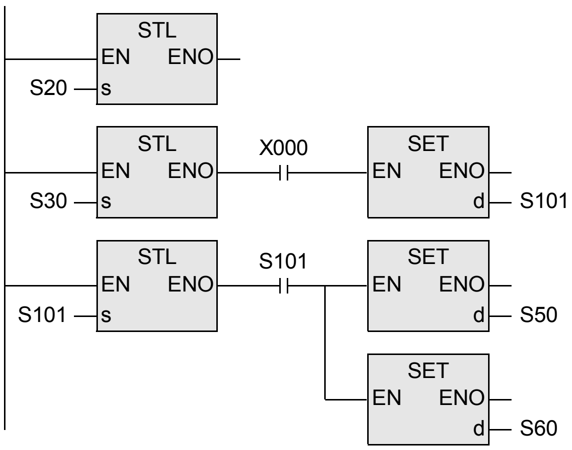
Before recombination, program the drive processing of state relays first.
After that, program only the transfer processing to recombination state relays continuously.

- Composition of branches and recombination
  When a recombination line is directly connected to a branch line (not by way of a state relay as shown below), it is recommended to provide a dummy state relay between the lines.
  Create step ladder programs as shown below.

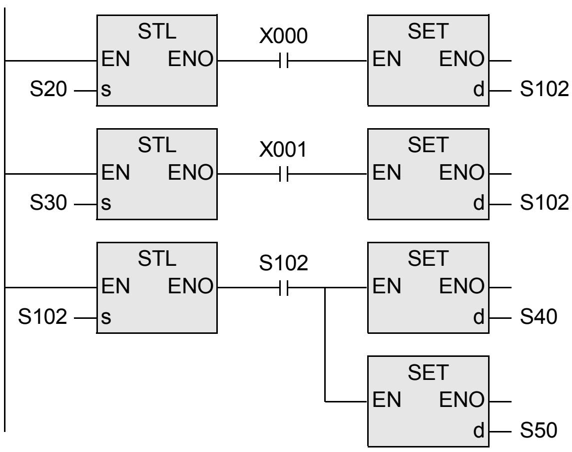
1) Selective recombination and selective branch



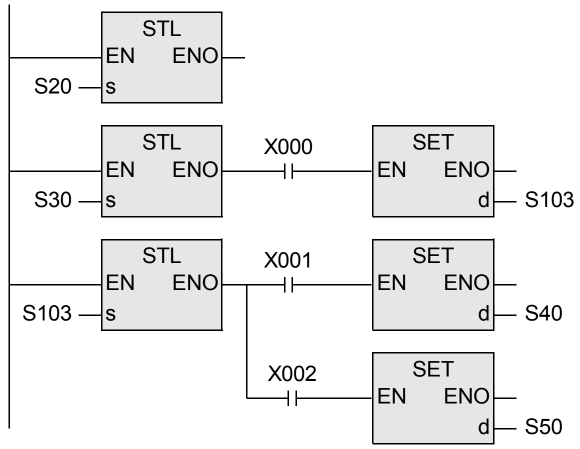
2) Parallel recombination and parallel branch



3) Selective recombination and parallel branch



4) Parallel recombination and selective branch

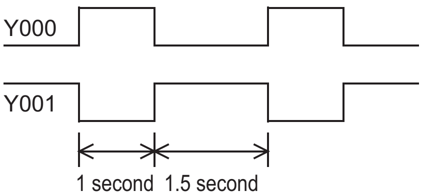


6.1.3 Program examples

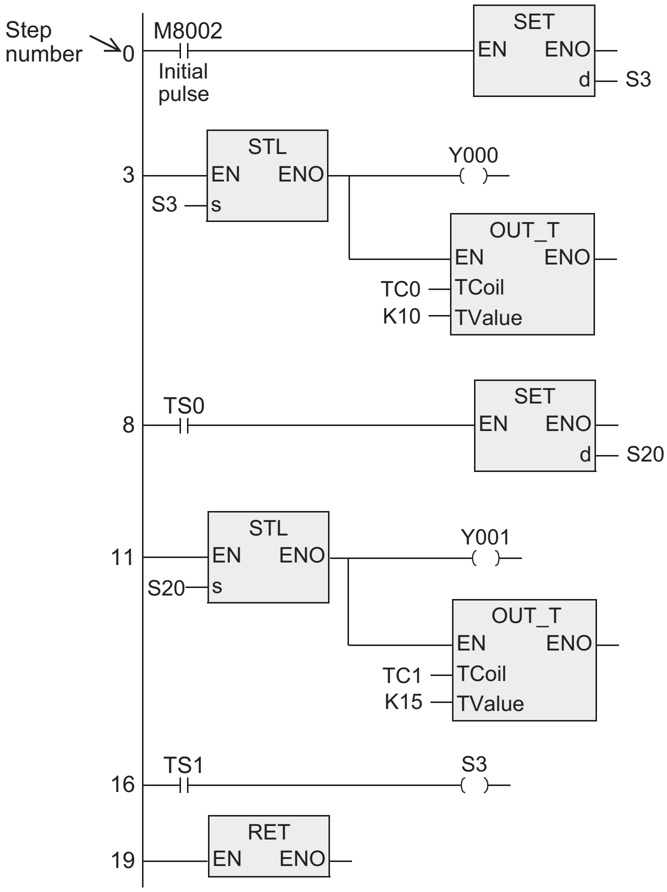
Examples of single flows

1. Example of flicker circuit

- When the PLC mode is changed from STOP to RUN, the state relay S3 is driven by the initial pulse (M8002).
- The state relay S3 outputs Y000. One second later, the ON status transfers to the state relay S20.
- The state relay S20 outputs Y001. 1.5 seconds later, the ON status returns to the state relay S3.



[Structured ladder]



[ST]

```
SET(M8002,S3);
STL(TRUE, S3);
      Y000:=TRUE;
      OUT_T(TRUE, TC0,K10);
       SET(TS0, S20);
STL(TRUE, S20);
      Y001:=TRUE;
      OUT_T(TRUE, TC1, K15);
       S3:=TS1;
RET(TRUE);
```

6.2 STL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

In programs using step ladder instructions, a state relay State S is assigned to each process based on machine operations, and input condition and output control are programmed as sequences connected to the state output.

STL instruction for step ladder programs is expressed as follows in each language.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
STL	16 bits	Continuous	STL EN ENO s	STL(EN,s);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Always TRUE
Output variable	(s)	Target device or variable	Bit
	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices													Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)						●																		

4. Caution

Refer to the cautions in the items below for expressing step ladders in a structured project (structured ladder, ST).

→ Section 6.3 RET

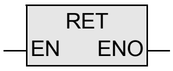
6.3 RET

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

RET instruction for step ladder programs is expressed as follows in each language.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RET	16 bits	Continuous		RET(EN);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Always TRUE
Output variable	ENO	Execution state	Bit

3. Applicable devices

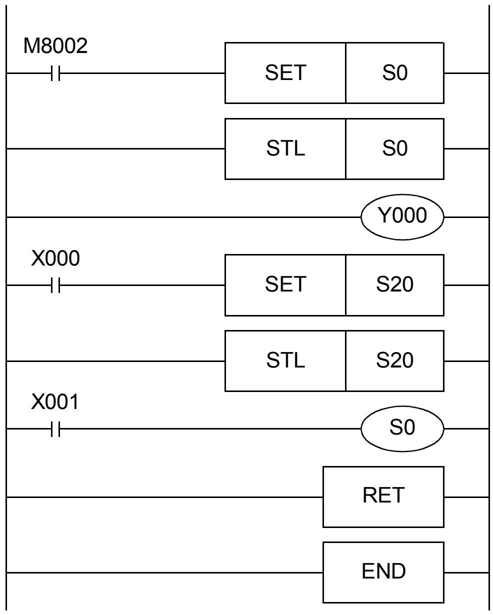
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
-	No target device is available.																							

4. Caution

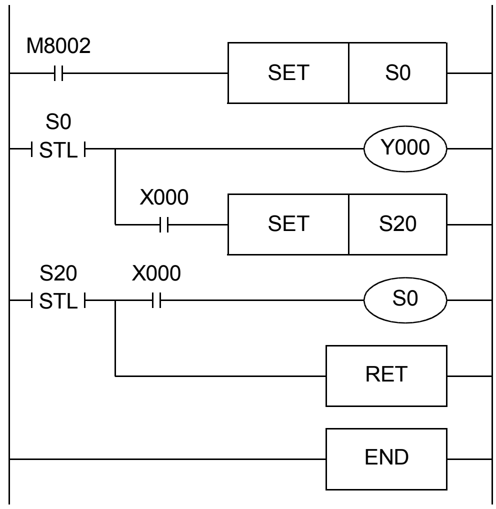
The following examples show how MELSEC-LD step ladders are expressed in the structured programs (structured ladder, ST).

Reference: MELSEC-LD step ladder expression

1) When expressing step ladder (STL) instructions in the coil format. (Same as that for GX Developer)

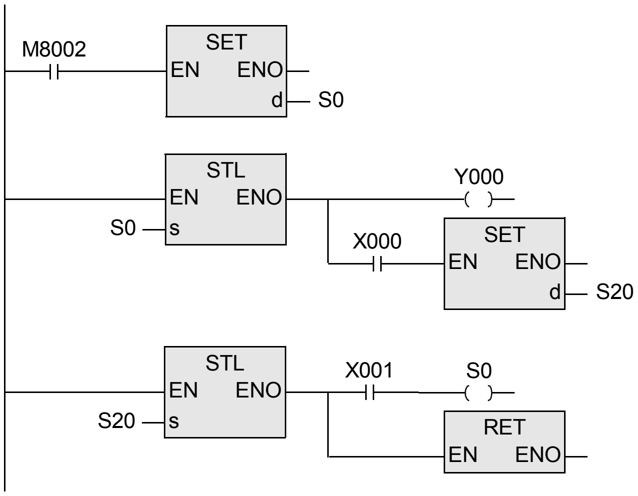


2) When expressing step ladder (STL) instructions in the contact format.



Expressing step ladder in structured program

1) Structured ladder



2) ST

```
SET(M8002, S0);
STL(TRUE, S0);
    Y000:=TRUE;
    SET(X000, S20);
STL(TRUE, S20);
    S0:=X001;
RET(TRUE);
```

7. Applied Instructions

This chapter introduces the structured project instructions corresponding to the applied instructions for the simple project.

→ **Q/FX Structured Programming Manual (Fundamentals)**

7.1 Program Flow

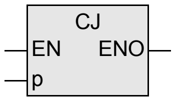
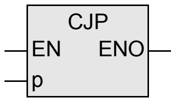
7.1.1 CJ

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

CJ or CJP instruction jumps to a pointer p.
The sequence program steps between CJ or CJP instruction and the pointer are not executed.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
CJ	16 bits	Continuous		Syntax such as condition sentence is used. Refer to the following manual for syntaxes. → **Q/FX Structured Programming Manual (Fundamentals)**
CJP	16 bits	Pulse		

2. Input and output data types

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(p)	Circuit block label for the jump destination	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

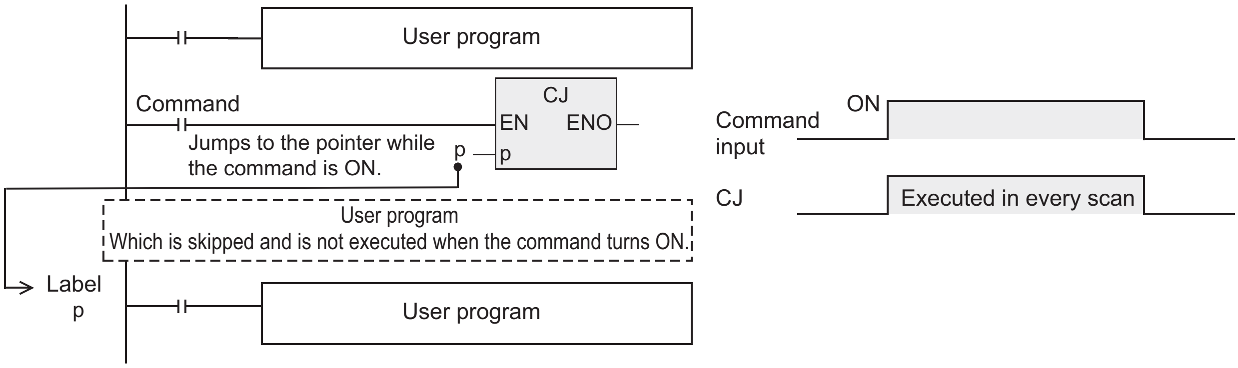
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(p)																			●					●

Function and operation explanation

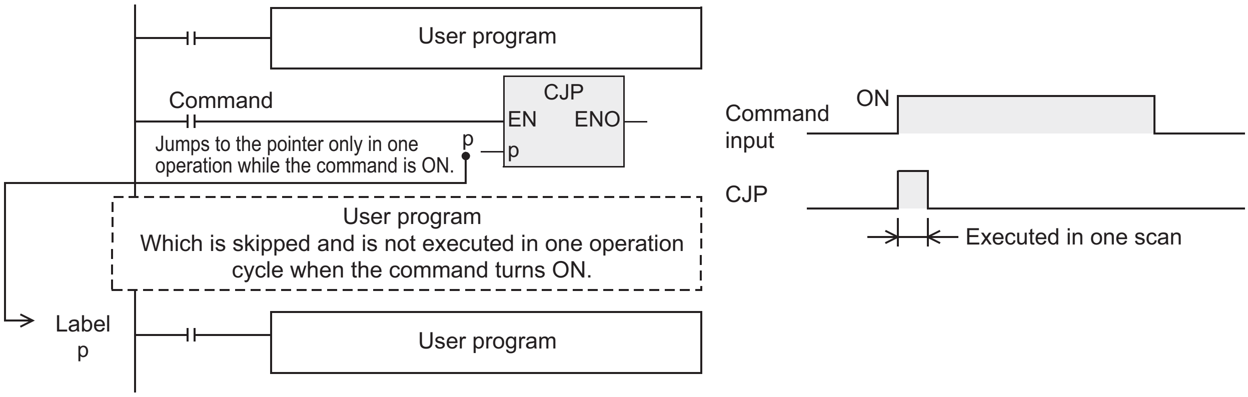
1. 16-bit operation(CJ, CJP)

While the command input is ON, CJ or CJP instruction executes a program with a specified label (pointer number).

1) In the case of CJ instruction

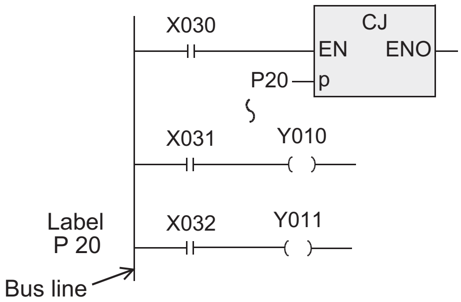


2) In the case of CJP instruction

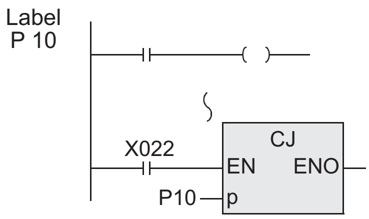


Cautions

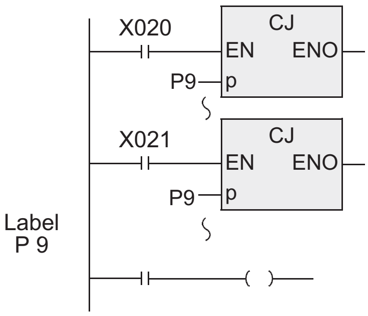
1) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
2) The figure below shows programming of a label.
   When creating a circuit program, move the cursor to the left side of the bus line in the ladder diagram, and input a label P at the head of the circuit block.



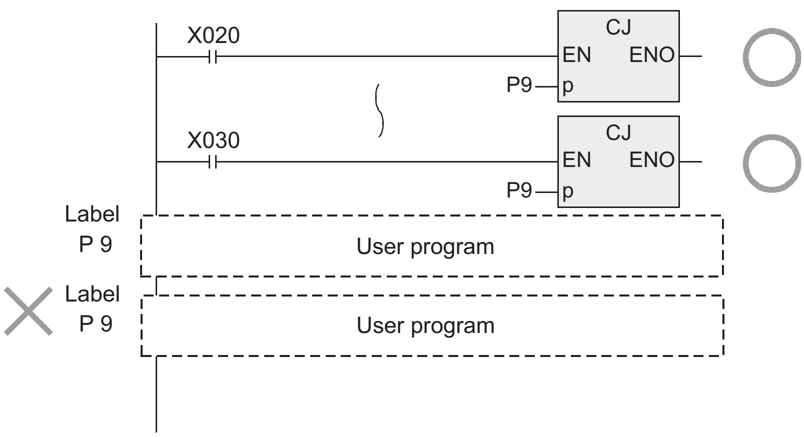
3) A label can be programmed in a smaller number step than CJ instruction. However, note that a watchdog timer error occurs when the scan time exceeds 200 ms (default setting).



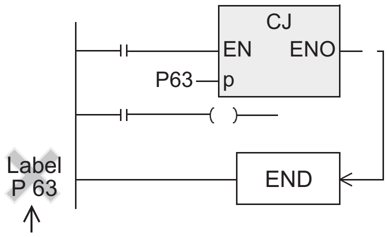
4) When the pointer number in operands is same and there is one label, the following operation is caused. When X020 turns ON, the program execution jumps from CJ instruction corresponding to X020 to the label P9. When X020 turns OFF and X021 turns ON, the program execution jumps from CJ instruction corresponding to X021 to the label P9.



5) When a label number (including labels for CALL instructions described later) is used two or more times, an error is caused.

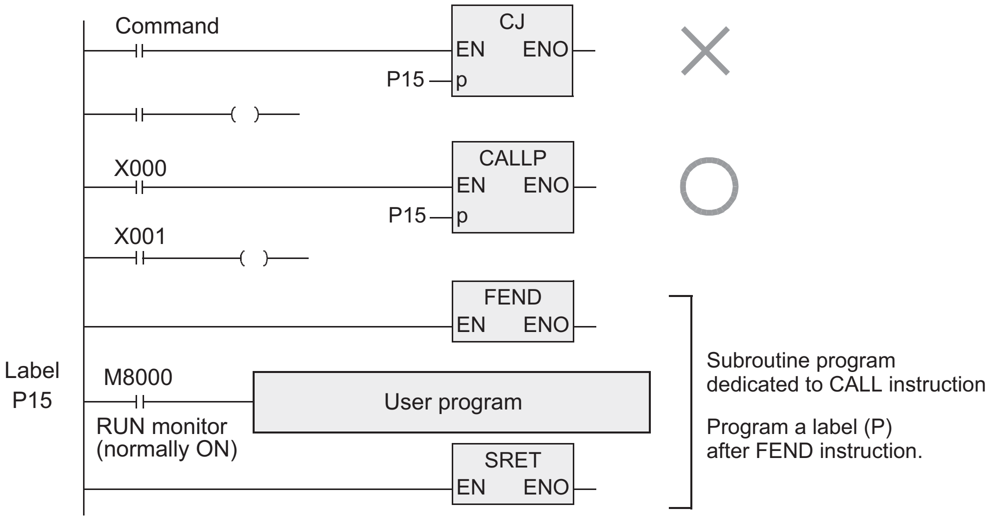


6) The pointer P63 specifies jump to END step. Do not program P63. If P63 is programmed, PLCs will display the error code 6507 (defective label definition) and stop.

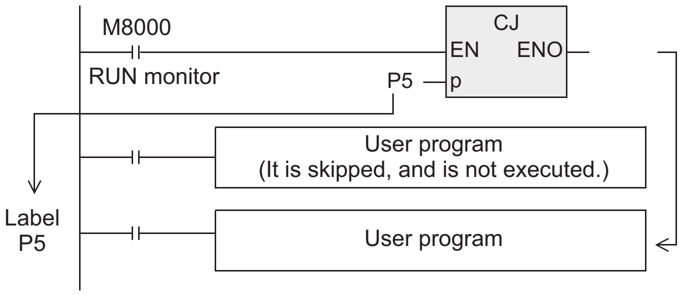


Do not program P63.

7) Any label cannot be shared by CALL instruction and CJ instruction.



8) Because M8000 is normally ON while a PLC is operating, unconditional jump is specified when M8000 is used in the following example.

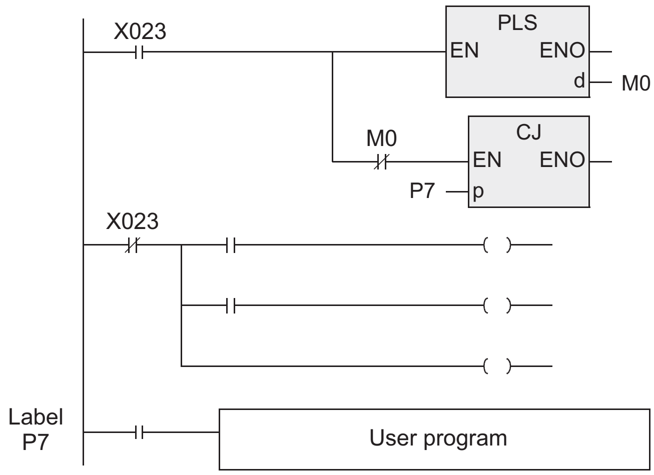


9) The operation of the CJ and contact coils are described later.

10) The relationships between the master control instructions and jump instructions are described later.

Program examples

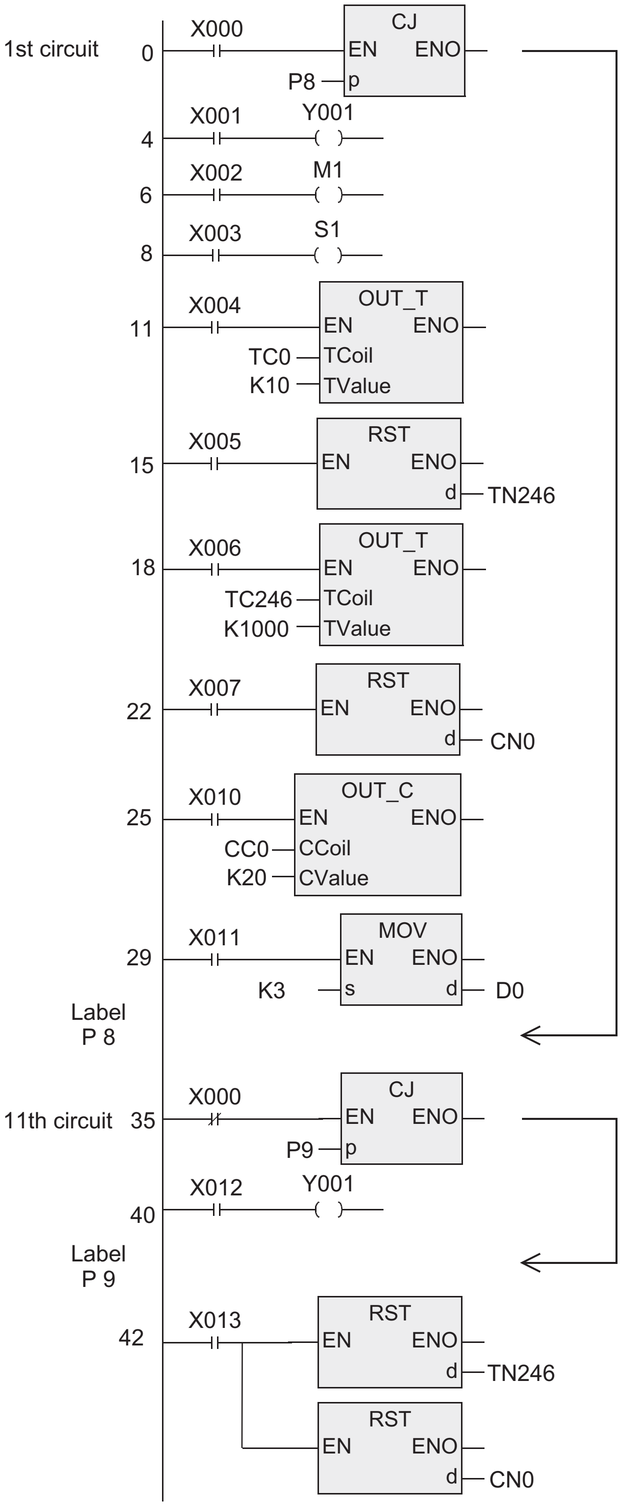
In one operation cycle after X023 changes to ON from OFF, CJ P7 instruction becomes valid.
By using this method, jump can be executed after all outputs between CJ P7 instruction and the label P7 turn OFF.



CJ instruction and operations of contact and coil

In the program example shown below, when X000 turns ON, the program execution jumps from CJ instruction in the first circuit to the label P8. While X000 is OFF, jump is executed. The program is sequentially executed from first step, and jumps from 11th circuit to the label 9.
The jumped instruction is not executed.

1. Circuit example 1 for explain operations

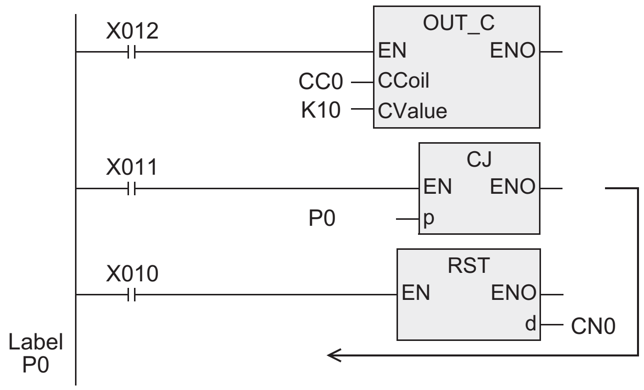


- Double coil operation of output Y001
  While X000 is OFF, output Y001 is activated by X001.
  While X000 is ON, output Y001 is activated by X012.
  Even in a program divided by conditional jumps, if a same coil (Y000 in this case) is programmed two or more times within the jump area or outside the jump area, such a coil is handled as double coil.
- When the reset (RST) instruction for the retentive type timer (T246) is located outside jump area:
  Even if the counting coil (T246) is jumped, reset (return of the contact and clearing of the current value) is valid.
- When the reset (RST) instruction for the counter (C0) is located outside the jump area:
  Even if the counting coil is jumped, reset (return of the contact and clearing of the current value) is valid.
- Operation of the routine timers:
  A routine timer continues its operation even if it is jumped after the coil is driven, and the output contact is activated.
- Operation of the high speed counters:
  A high speed counter continues its operation even if it is jumped after the coil is driven, and the output contact is activated.

When each input changes during jump in the program shown on the left, each coil executes the following operation:

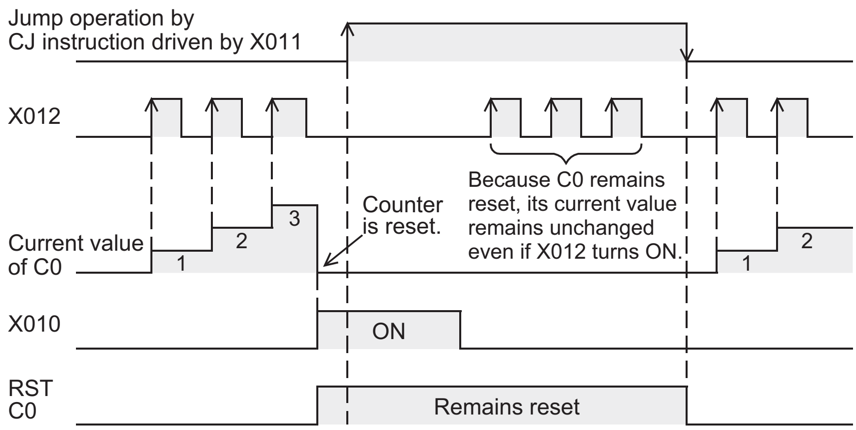
Classification	Contact status before jump	Coil operation during jump
Y, M, S (Y001, M1, S1)	X001, X002, X003 OFF	Y001, M1, S1 OFF
	X001, X002, X003 ON	Y001, M1, S1 ON
10 ms timer and 100 ms timer (T0)	X004 OFF	Timer is not activated.
	X004 ON	Counting is paused (and is restarted after X000 turns OFF).
1 ms timer (T246)	X005 OFF X006 OFF	Timer is not activated. The deactivation status is reset when X013 turns ON.
	X005 OFF X006 ON	Counting is continued (and the contact is activated after X000 turns OFF).
Counter (C0)	X007 OFF X010 OFF	Countint is not activated. The deactivation status is reset when X013 turns ON.
	X007 OFF X010 ON	Counting is paused (and is restarted after X000 turns OFF).
Applied instruction (MOV)	X011 OFF	FNC instruction is not executed during jump. But MTR, HSCS, HSCR, HSZ, SPD, PLSY and PWM instructions continue their operations.
	X011 ON	

2. Circuit example 2 for explaining operations (when only an RST instruction for timer or counter is jumped)

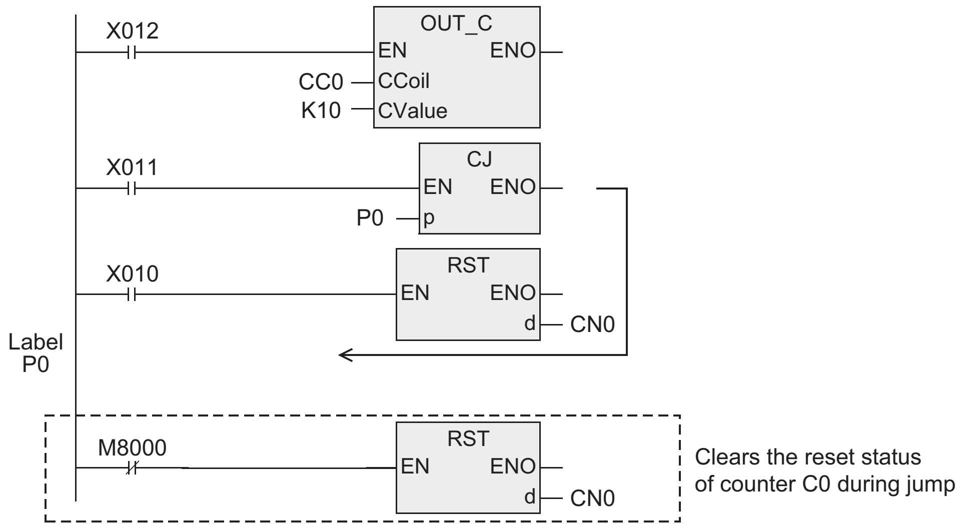


When X011 turns ON while the RST instruction for the counter C0 is operating (X010 is ON), the program execution jumps past the RST instruction due to the CJ instruction.
In this jump status, the counter C0 remains reset. Accordingly, the current value of C0 remains "0" even if X012 turns ON.
To clear this reset status, it is necessary to turn OFF the RST instruction for counter C0. (Refer to the program shown below.)

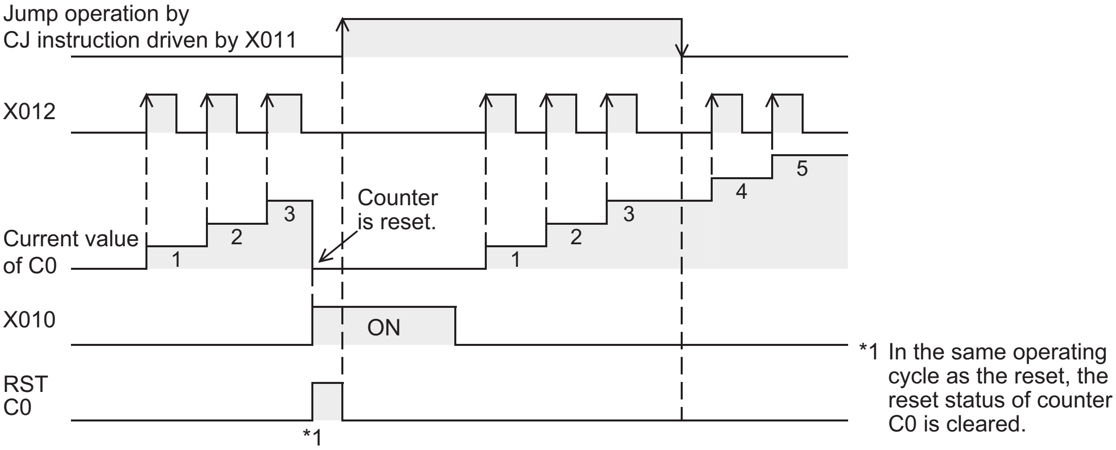
Timing chart



Program example for activating a timer and counter even during a jump

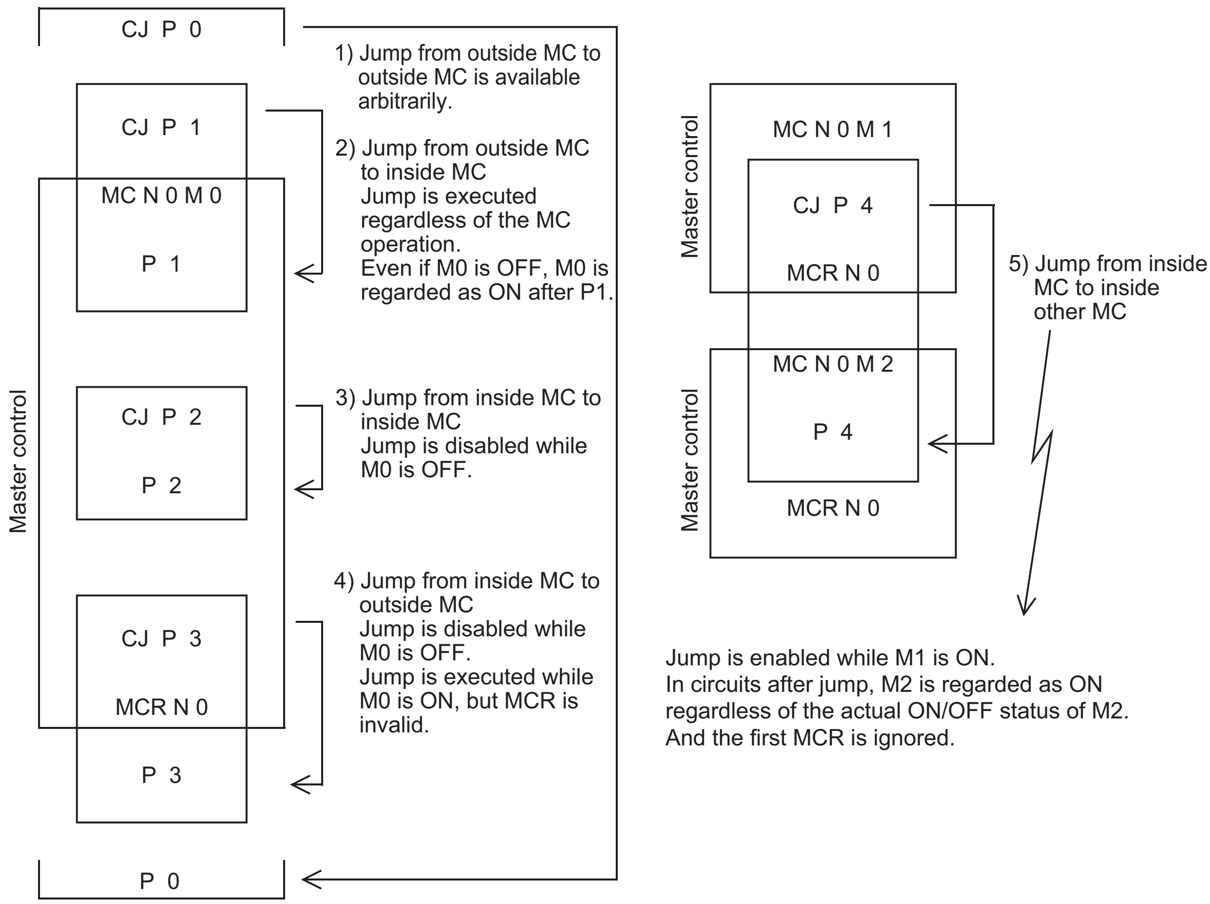


Timing chart



Relationship between master control instruction and jump instruction

The figure below shows the contents of operation and the relationship between the master control instruction.
Avoid using 2), 4) and 5) because the operation will be complicated.



7.1.2 CALL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This instruction calls and executes a program which should be processed commonly in a sequence program.
This instruction saves the number of program steps, and achieves efficient program design.
For creating a subroutine program, FEND and SRET instructions are required.
A similar processing is available by creating a function block and read it out from the program block.
Refer to the following manual for creating function blocks.

→ **GX Works2 Version1 Operating Manual (Structured Project)**
→ **Q/FX Structured Programming Manual (Fundamentals)**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
CALL	16 bits	Continuous	CALL —EN ENO— —p	Use a subroutine program by reading out the function block made of other program parts.
CALLP		Pulse	CALLP —EN ENO— —p	

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(p)	Circuit block label of subroutine program to be executed.	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

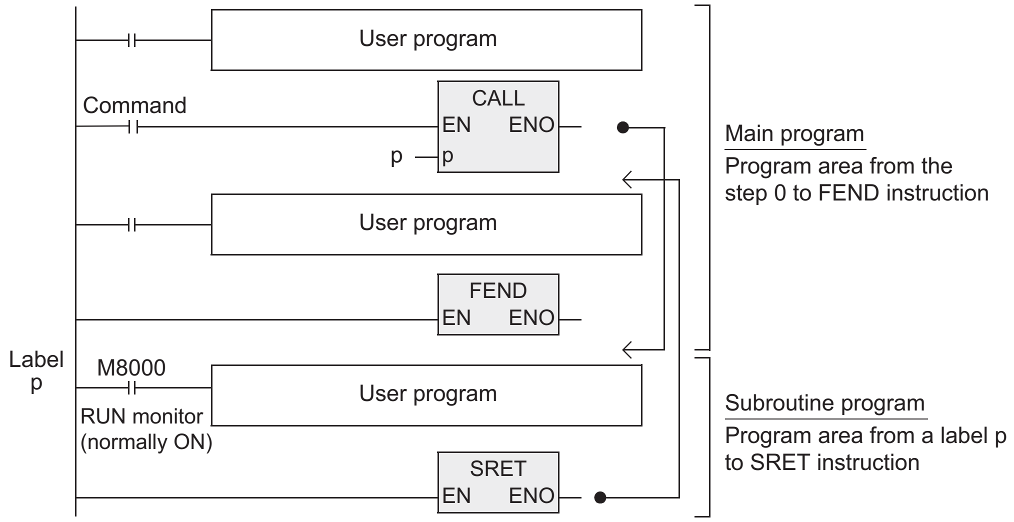
Operand type	Bit Devices							Word Devices													Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(p)																			●					●

Function and operation explanation

1. 16-bit operation

While the command input is ON, CALL instruction is executed and the program execution jumps to a step with a label p.
Then, a subroutine program with the label p is executed.
When SRET instruction is executed, the program execution returns to the step after CALL instruction.

- At the end of the main program, put FEND instruction.
- Put a label p for CALL instruction after FEND instruction.



Cautions

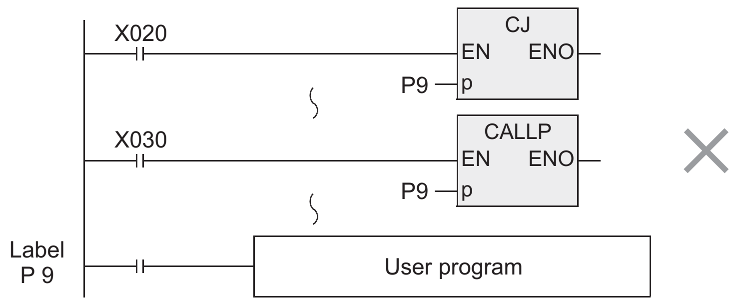
1) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
To execute pulse operation, make the instruction execution condition pulse type.

2) To use the subroutine call, follow the steps below.
Name the task "MELSEC_MAIN".
Using a different task name prompts an error because the "one set in the program block" by the function FEND and the "one finally added to the program block during compiling" become redundant.

Be sure to program in combination with the SRET and FEND functions.

→ **Refer to Section 7.1.3 for SRET.**
→ **Refer to Section 7.1.7 for FEND.**

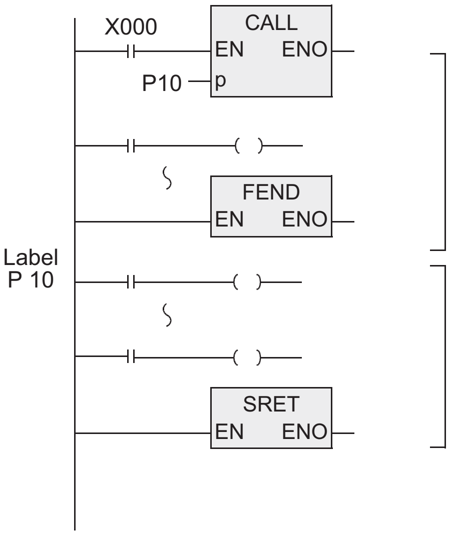
3) In CALL instructions, a same number can be used two or more times in operands (P).
However, do not use a label (P) and number used in another instruction (CJ).



4) Cautions about the use in subroutines or interrupt routines are described later.

Program examples

1. Example of fundamental use (no nesting)

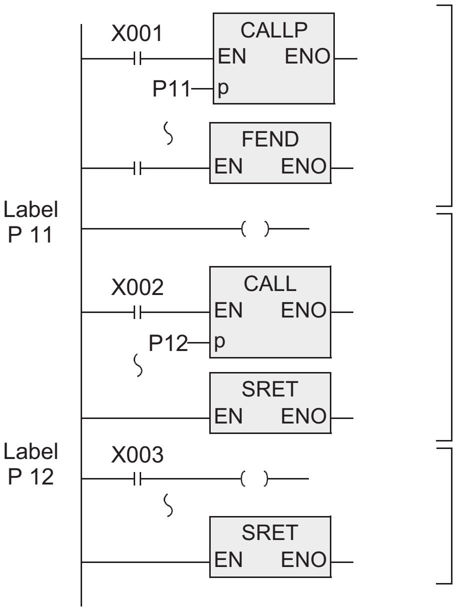


Main program
While X000 is ON, the program execution jumps to a step with the label P10.

Subroutine program
When SRET instruction is executed after the subroutine program has executed, the program execution returns to the original step +1.

2. Example of multiple CALL instructions in subroutines (multiple nesting)

CALL instruction can be used up to 4 times in subroutine programs. Nesting of up to five layers is allowed.



Main program
When X001 turns ON from OFF, the program execution jumps to the label P11 only once.

Subroutine program1
When SRET instruction is executed, the program execution returns to the main program.
If X002 is ON while the subroutine program 1 is executed, the program execution jumps to a step with the label P12.

Subroutine program2
The subroutine program with P12 is executed, and then the program execution returns to the subroutine program with P11 by SRET instruction.

Cautions on subroutines and interrupt routines

This section explains cautions on creating programs in subroutines and interrupt routines.
The explanation below is given for subroutines, but the situation also applies to interrupt routines.

1. When using timers in subroutines (or interrupt routines)

Use retentive type timers T192 to T199 in subroutines.
These timers execute counting when the coil instruction or END instruction is executed.
After a timer reaches the set value, the output contact is activated when the coil instruction or END instruction is executed.
Because general timers execute counting only when the coil instruction is executed, they do not execute counting if they are used in subroutines in which the coil instruction is executed only under some conditions.

2. When using retentive type 1 ms timers in subroutines (or interrupt routines)

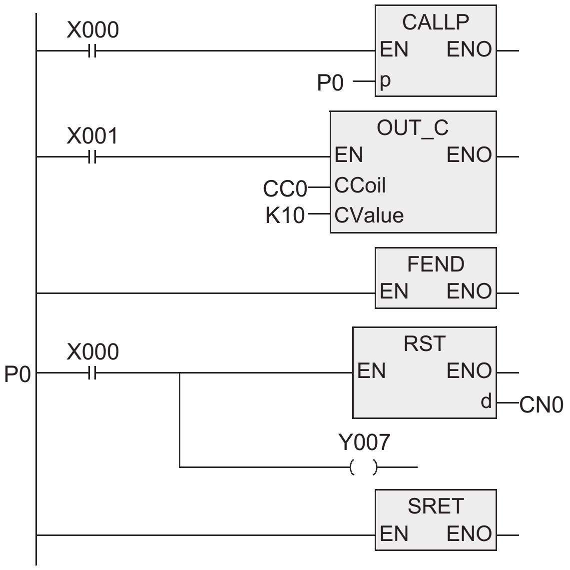
If a retentive type 1 ms timer is used in a subroutine, note that the output contact is activated when the first coil instruction (or subroutine) is executed after the timer reaches its set value.

3. Countermeasures against latches of devices used in subroutines (or interrupt routines)

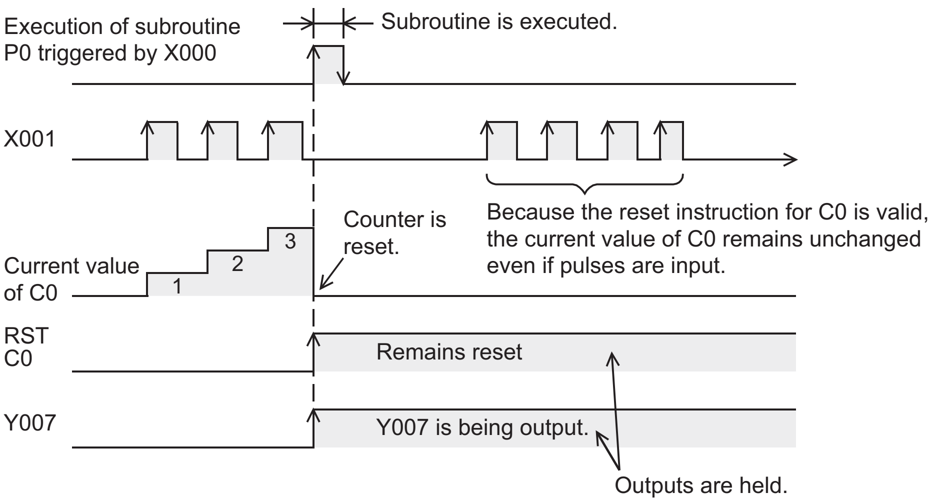
Devices which were set to ON in a subroutine are latched in the ON status even after the subroutine is finished. (Refer to the program described later.)
When RST instruction for a timer or counter is executed, the reset status of the timer or counter is latched also.
For turning OFF such a device latched in the ON status or for canceling such a timer or counter latched in the reset status, reset such a device in the main program after the routine is finished, or program a sequence for resetting such a device or for deactivating RST instruction in the routine. (Refer to the program described later.)

1) Example in which outputs are latched
   In the following program example, the counter C0 is provided to count X001. When X000 is input, the subroutine P0 is executed only in one scan, and then the counter is reset and Y007 is output.

• Program examples

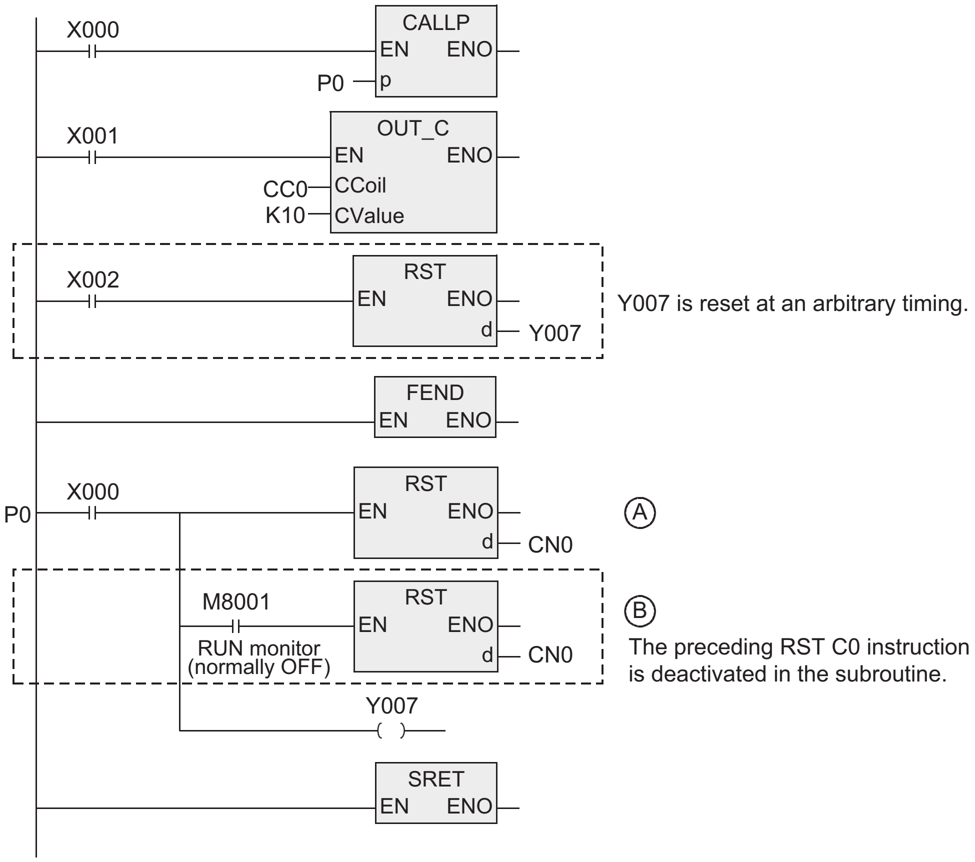


• Timing chart

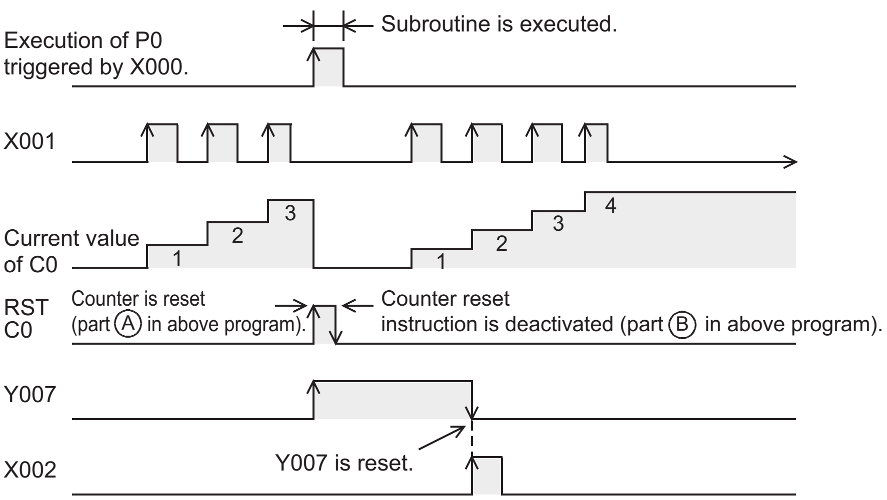


2) Example for resetting held outputs (countermeasures)

- Program examples



- Timing chart



7.1.3 SRET

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This instruction returns the program execution from a subroutine to the main program.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SRET	16 bits	Continuous	SRET EN ENO	Use a subroutine program by reading out the function block made of other program parts.

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Always TRUE
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
-	No target device is available.																							

Function and operation explanation

When CALL instruction in the main program is executed, the program execution jumps to a subroutine. SRET instruction returns the program execution to the main routine.

→ **Refer to Section 7.1.2.**

7.1.4 IRET

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction returns the program execution from an interrupt routine to the main program.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
IRET	16 bits	Continuous	[IRET block: EN — ENO]	IRET(EN);

2. Input and output data types

Variable		Description	Data type
Input variable	EN	Execution condition	Always TRUE
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
-	No target device is available.																							

Function and operation explanation

When an interrupt (input, timer or counter) is generated while the main program is executed, the program execution jumps to an interrupt (I) routine.
IRET instruction returns the program execution to the main routine.
The table below shows the three types of jump to an interrupt routine.

Function	Description
Input interrupt	Executes the interrupt processing when an input(X) signal turns ON or OFF.
Timer interrupt	Executes the interrupt processing at a specified time interval (constant cycle).
Counter interrupt [*1]	Executes the interrupt processing when a high speed counter reaches its set value.

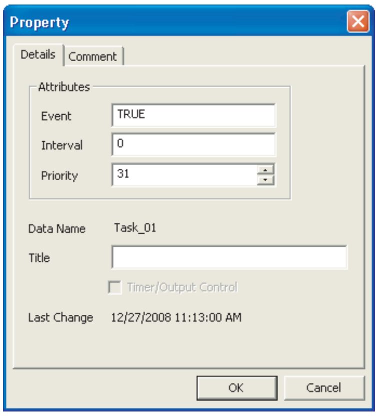
*1. Only the FX3U, FX3UC and FX2C PLCs of V3.07 or later support this function.

→ **For the interrupt function, refer to Chapter 8.**

Cautions

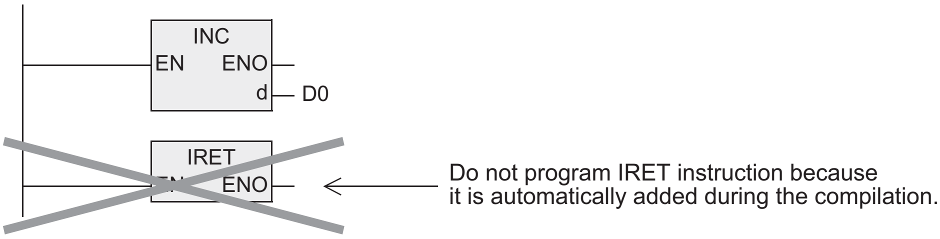
1) Create a task for the interrupt program and the main program.
2) Use "Event" to specify the interrupt pointer to be used for the task for the interrupt program.

→ **For the interrupt pointer, refer to Chapter 8.**

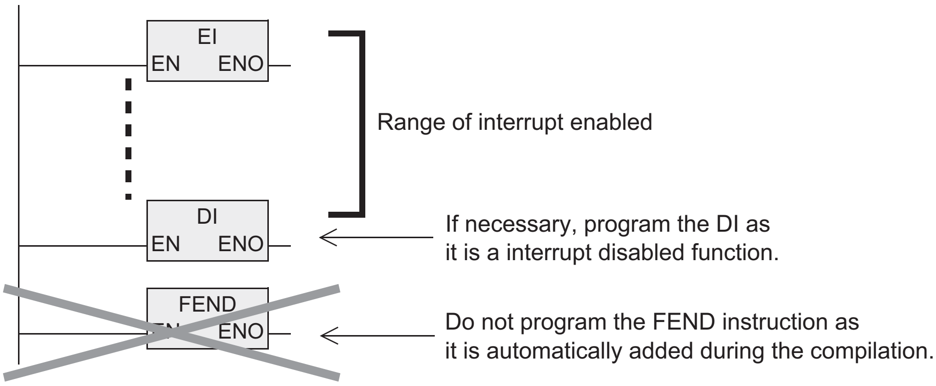


3) IRET instruction needs not to be programmed because the function IRET is automatically added during the compilation at the end of the program block that is registered in the task for the interrupt program.

Interrupt program



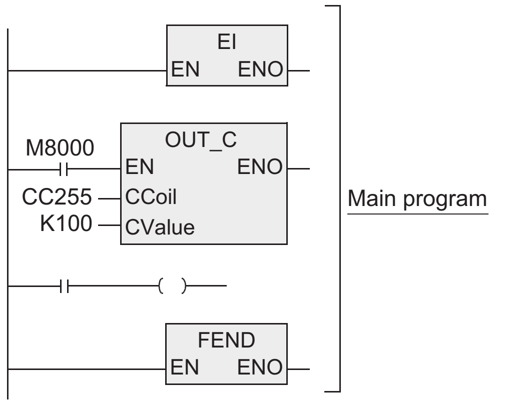
4) The program block registered in the task for the main program requires the function EI (interrupt enabled). Program the function DI (interrupt disabled) as necessary.



Program examples

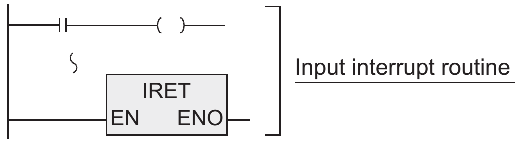
[Structured ladder]

Task for main program

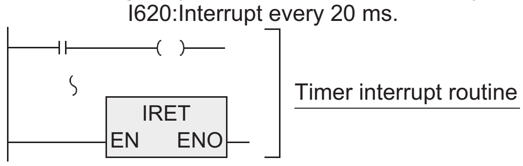


Interrupts are usually disabled in PLCs.
Use EI instruction to enable interrupts.
When X000 turns ON while the main program is executed,instructions after the interrupt routine pointer I001 are executed, and the program execution returns to the original main program by IRET instruction.

Task for interrupt program(Interrupt pointer I001 is set by event.)
I001:The rising edge of X000 is detected.

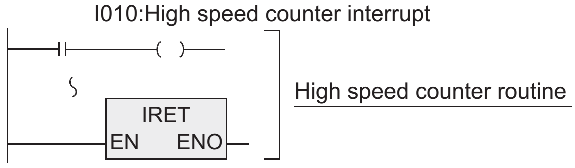


Task for interrupt program(Interrupt pointer I620 is set by event.)
I620:Interrupt every 20 ms.



The timer interrupt of the pointer I620 is executed every timer time of 20 ms, and the program execution is returned to the original main program by IRET instruction each time.

Task for interrupt program(Interrupt pointer I010 is set by event.)
I010:High speed counter interrupt



The high speed counter interrupt of the pointer I010 is executed when the current value of a high speed counter becomes equivalent to a value specified by DHSCS instruction, and the program execution returns to the original main program by IRET.

[ST]

Task for main program

```
EI(TRUE);
OUT_C(M8000, CC255, K100);
   ∫
FEND(TRUE)
```

Main program

Interrupts are usually disabled in PLCs.
Use EI instruction to enable interrupts.
When X000 turns ON while the main program is executed,instructions after the interrupt routine pointer I001 are executed, and the program execution returns to the original main program by IRET instruction.

Task for interrupt program(Interrupt pointer I001 is set by event.)
I001:The rising edge of X000 is detected.

```
Y000: =X000;
   ∫
IRET(TRUE);
```

Input interrupt routine

Task for interrupt program(Interrupt pointer I620 is set by event.)
I620:Interrupt every 20 ms.

```
Y000: =X000;
   ∫
IRET(TRUE);
```

Timer interrupt routine

The timer interrupt of the pointer I620 is executed every timer time of 20 ms, and the program execution is returned to the original main program by IRET instruction each time.

Task for interrupt program(Interrupt pointer I010 is set by event.)
I010:High speed counter interrupt

```
Y000: =X000;
   ∫
IRET(TRUE);
```

High speed counter routine

The high speed counter interrupt of the pointer I010 is executed when the current value of a high speed counter becomes equivalent to a value specified by DHSCS instruction, and the program execution returns to the original main program by IRET.

7.1.5 DI

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction disables interrupts after interrupts were enabled by EI instruction.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DI	16 bits	Continuous	DI EN ENO	DI(EN);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Always TRUE
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
-	No target device is available.																							

Function and operation explanation

DI instruction is the independent type, and does not require command (drive) contact.

Cautions

Interrupts (requests) generated after DI instruction are processed after EI instruction is executed.

7.1.6 EI

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

Interrupts are usually disabled in PLCs. This instruction enables interrupts in PLCs.
Use this instruction for using the input interrupt, timer interrupt and counter interrupt functions.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
EI	16 bits	Continuous	EI (EN → ENO)	EI(EN);

2. Set data

Variable		Description	Data type
Input variable	EN	Input condition	Always TRUE
Output variable	ENO	Input status	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
-	No target device is available.																							

Function and operation explanation

EI instruction is the independent type, and does not require command (drive) contact.

Cautions

1) Refer to the following items for the cautions on the interrupt program.

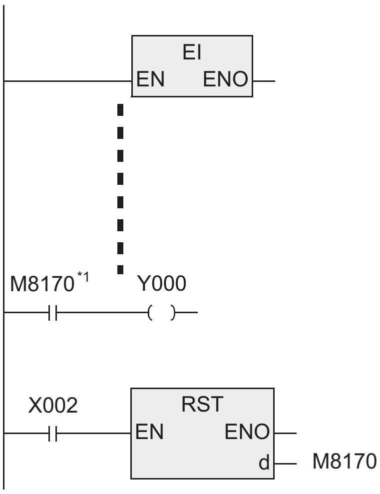
→ **Refer to Section 7.1.4.**

2) Use the EI instruction as follows when the FXU, FX2C, FX2N, FX2NC, FX3U and FX3UC PLCs use the pulse catch function. The IE instruction does not need to be programmed when the FX0, FX0S, FX0N, FX1S, FX1N, FX1NC or FX3G PLC uses the pulse catch function.
For the details of special auxiliary relays and other devices used with the pulse catch function, refer to the following manual.

→ **FX Structured Programming Manual (Device & Common)**

When using the FX3U PLC

[Structured ladder]



[ST]

```
EI(TRUE)
  :
Y000:=M8170;*1
RST(X001, M8170);*1
```

With the rising edge of X000 detected, M8170 is reset by interrupt.

Resets pulse catch results.

*1. A special auxiliary relay for the X000 pulse catch function used in the FX1S, FX1N, FXU, FX2C, FX2N, FX2NC, FX3G, FX3U and FX3UC PLCs. The special auxiliary relay depends on the PLC used and input number. For the pulse catch function, refer to Chapter 8.

7.1.7 FEND

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction indicates the end of the main program.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
FEND	16 bits	Continuous	FEND (EN — ENO)	FEND(EN);

2. Set data

Variable		Description	Data type
Input variable	EN	Input condition	Always TRUE
Output variable	ENO	Input status	Bit

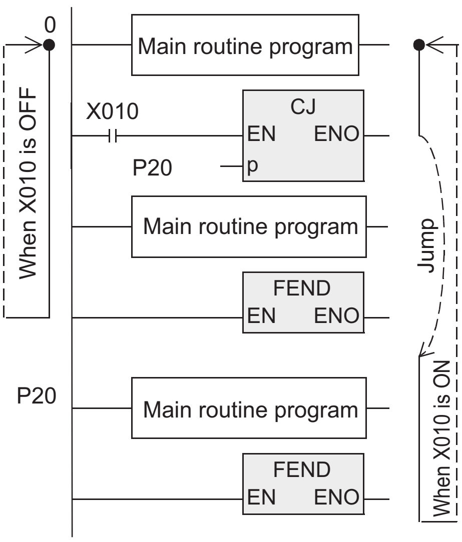
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
-	No target device is available.																							

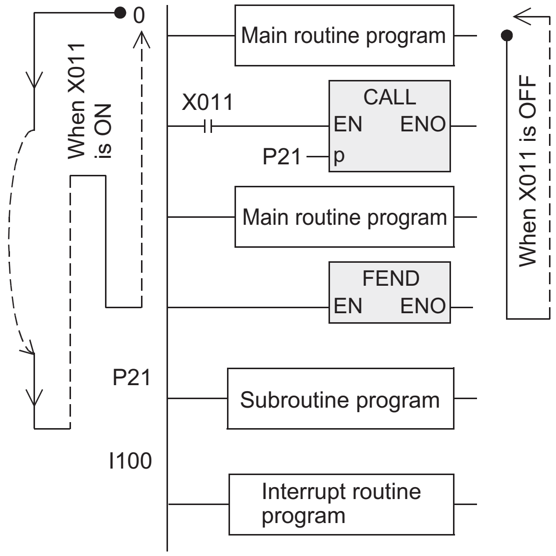
Function and operation explanation

When FEND instruction is executed, output processing, input processing and watchdog timer refresh are executed, and then the program execution returns to the step 0.
FEND instruction is required in creating subroutine programs and interrupt programs.

1. In the case of CJ instruction



2. In the case of CALL instruction



Cautions

1) The function FEND instruction is usually added automatically during compilation.
   It is not necessary to program the FEND instruction in the program block except when creating a subroutine. As for the subroutine programs, refer to the following.
   → **Refer to Section 7.1.2.**
2) When FEND instruction is programmed two or more times, put a subroutine program or interrupt routine program between the last FEND instruction and END instruction.
3) When CALL or CALLP instruction is used, put a label after FEND instruction. And the SRET instruction is required in every case.
4) When CALL or CALLP instruction is used, if FEND instruction is executed after CALL or CALLP instruction was executed and before SRET instruction is executed, an error is caused.
5) When FOR instruction is used, if FEND instruction is executed after FOR instruction was executed and before NEXT instruction is executed, an error is caused.
6) When the interrupt function (I) is used, be sure to program an interrupt label (pointer) after FEND instruction. And IRET instruction is required in every case.

7.1.8 WDT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction refreshes the watchdog timer in a sequence program.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
WDT	16 bits	Continuous	WDT EN ENO	WDT(EN);
WDTP		Pulse	WDTP EN ENO	WDTP(EN);

2. Set data

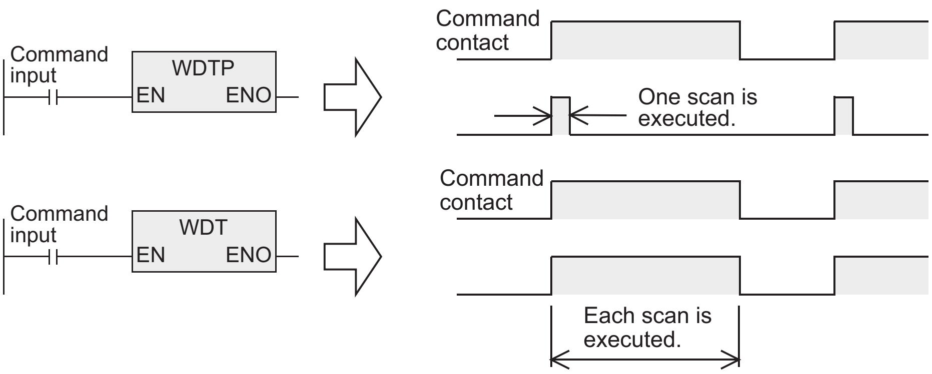
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
-	No target device is available.																							

Function and operation explanation

When the operation cycle (time until END or FEND instruction is executed after the step 0) of a PLC exceeds 200 ms, a watchdog timer error (indicating abnormal operation) occurs. The CPU error LED lights, and the PLC stops. When the operation cycle is long, insert WDT instruction in the middle of the program to avoid the watchdog timer error.

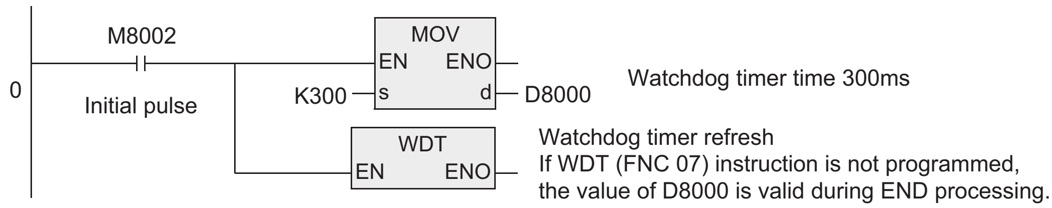


Related device

Device	Name	Description
D8000	Watchdog timer time	Up to 32767 ms can be set in units of ms (initial value: 200 ms).

Cautions

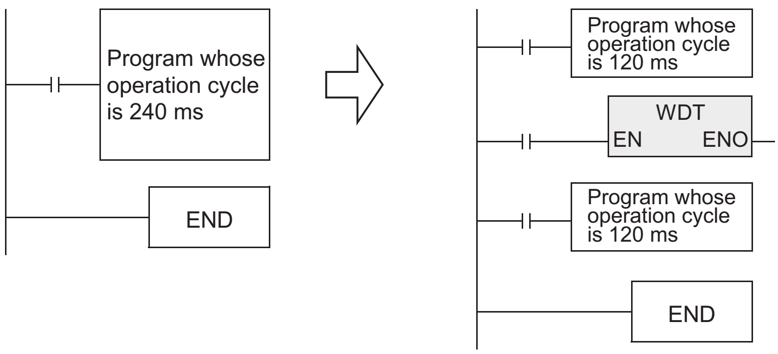
1) The FX0, FX0S or FX0N PLC does not support the pulse operation type instructions.
   To execute pulse operation, make the instruction execution condition pulse type.
2) A watchdog timer error may occur in the following cases. To avoid the error, input a program shown below near the head step to extend the watchdog timer time, or shift FROM/TO instruction execution timing.
   - Caution when many special extension devices are connected.
     In such configuration that many special extension devices (such as positioning units, cam switches, analog units and link units) are connected, the buffer memory initialization time may become longer, thus the operation time may become longer, and a watchdog timer error may occur.
   - Caution when many FROM/TO instructions are driven at one time.
     When many FROM/TO instructions are executed or when many buffer memories are transferred, the operation time may become longer, and a watchdog timer error may occur.
   - Caution when there are many high speed counters (software counters).
     When many high speed counters are provided and high frequency are counted at one time, the operation time may become longer, and a watchdog timer error may occur.
3) The watchdog timer time can be changed.
   By overwriting the contents of D8000 (watchdog timer time), the watchdog timer detection time (initial value: 200 ms) can be changed.
   By inputting the program shown below, the sequence program after this insertion is monitored by a new watchdog timer time.



Program examples

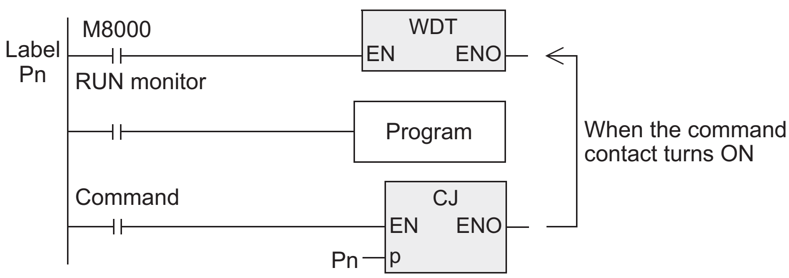
1. When the operation cycle is long and causes an error

For example, by dividing a program whose operation cycle is 240 ms into two portions and inserting WDT instruction between them, the operation cycle becomes less than 200 ms in both the former half portion and the latter half portion.



2. When a label (P) of CJ instruction is located in a step number smaller than the step number of CJ instruction

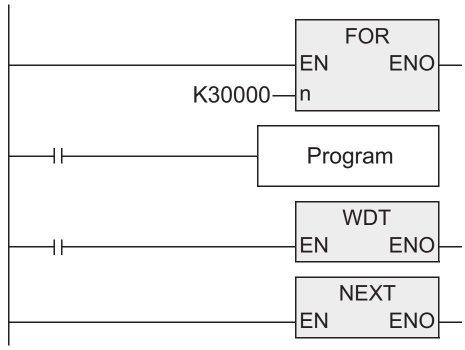
Put WDT instruction after the label (P).



If an input relay (X) is used as the command contact, input refresh is disabled, so the program execution cannot be returned from the area between P and CJ. As the command contact, use such device that can be set to OFF in a program being jumped.

3. When FOR/NEXT instruction is repeated many times

Put WDT instruction between FOR and NEXT.



7.1.9 FOR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

FOR instruction specifies the number of repetition of the loop between FOR and NEXT instructions.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
FOR	16 bits	Continuous	FOR block: EN, ENO, n	FOR(EN);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Always TRUE
	n	Number of repetition of the loop between FOR and NEXT instructions	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
n								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			

▲: Refer to "Cautions".

Function and operation explanation

→ **Refer to Section 7.1.10 for details.**

Related instruction

FOR instruction and NEXT instruction are set as a pair in programming.

Cautions

1) The repeat syntax (FOR...DO syntax) can program the same function.
   For the repeat syntax of the ST program, refer to the following manual.
   → **FX Structured Programming Manual (Device & Common)**

2) Some restrictions to applicable devices
   ▲1: The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2: The FX3U and FX3UC PLCs only are applicable.

7.1.10 NEXT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

FOR instruction specifies the number of repetition of the loop between FOR and NEXT instructions.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
NEXT	16 bits	Continuous	NEXT EN ENO	NEXT(EN);

2. Set data

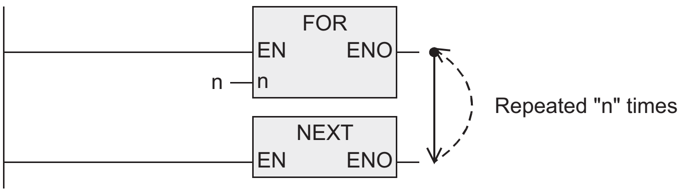
Variable		Description	Data type
Input variable	EN	Execution condition	Always TRUE
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
-	No target device is available.																							

Function and operation explanation

The loop between FOR and NEXT instruction is repeated "n" times (which is specified by the input variable (n)).
After the loop is repeated by the specified number of times, steps after NEXT instruction are executed.

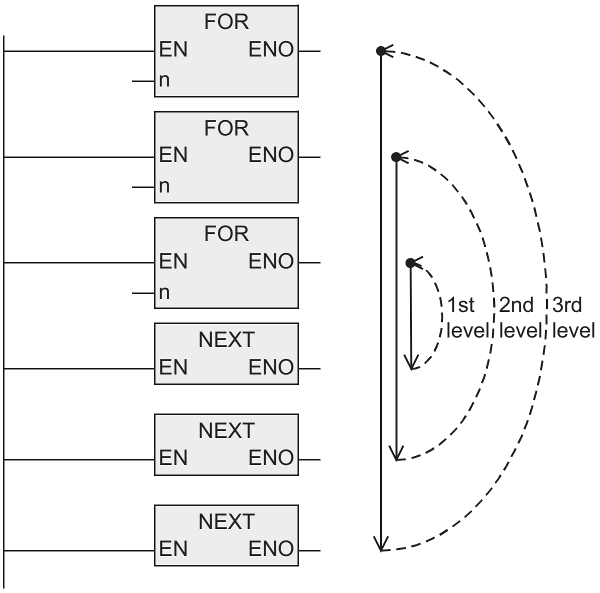


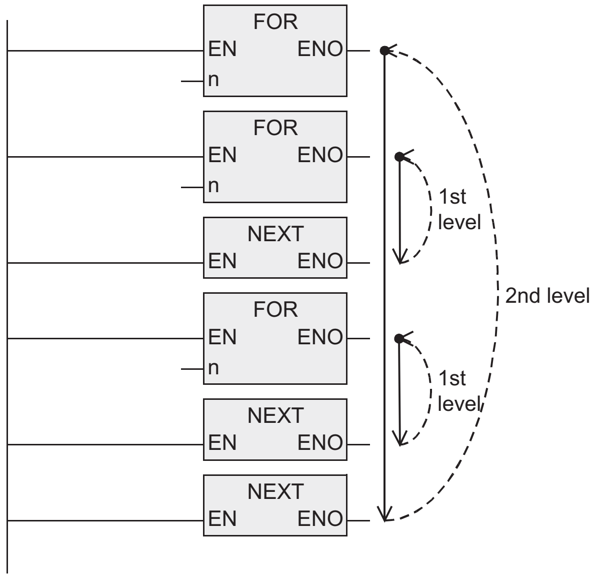
Related instruction

NEXT instruction and FOR instruction are set as a pair in programming.

Cautions

FOR-NEXT loop can be nested up to 5 levels.



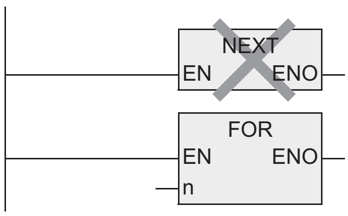


Error

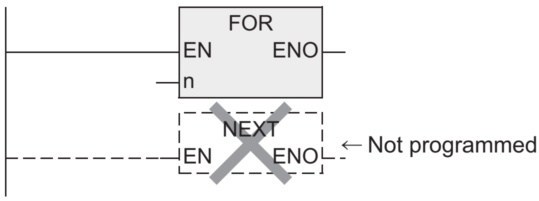
1) When FOR-NEXT loop is repeated many times, the operation cycle (D8010) is too long, and a watchdog timer error may occur. In such a case, change the watchdog timer time or reset the watchdog timer.
→ **For details on changing and resetting the watchdog timer, refer to Section 7.1.8.**

2) The following programs are regarded as errors.

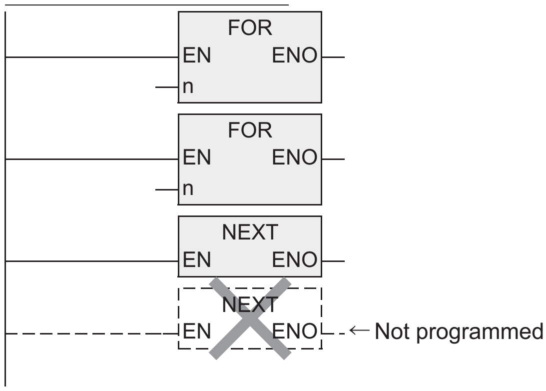
When NEXT instruction is located before FOR instruction



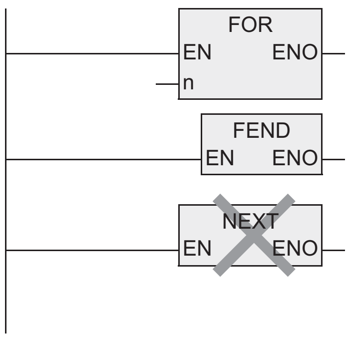
When NEXT instruction does not exist



When number of FOR instructions is not equivalent to the number of NEXT instructions.

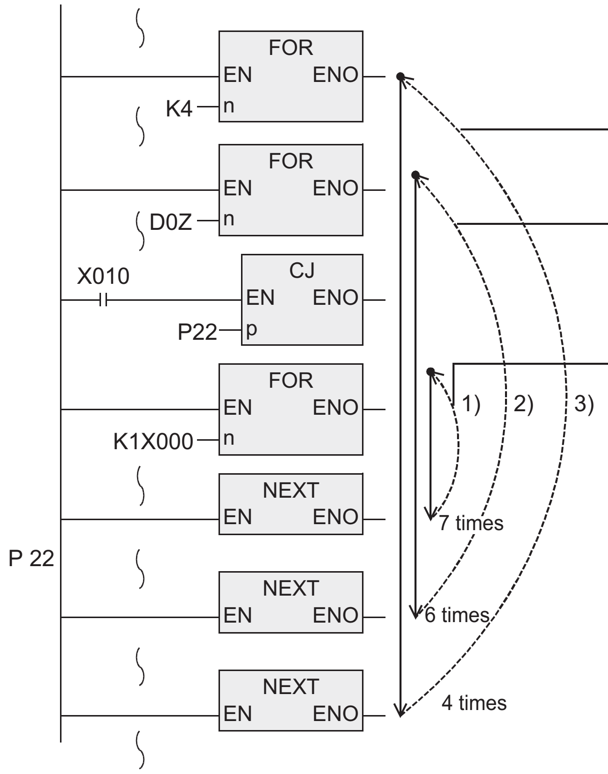


When NEXT instruction exits after FEND instruction.



Program examples

1. Program example with three FOR-NEXT loops



The loop 3) is repeated 4 times.

When the data value (current value) of D0Z (D4 when Z is "4") is "6", the loop 2) is repeated 6 times.

- When X010 is OFF
  When K1X000 is "7", the loop 1) is repeated 7 times.
- When X010 is ON
  The program execution jumps to the pointer P22, and the loop 1) is skipped.

Number of times of repeating the loops 1), 2) and 3).

	X010=OFF	X010=ON
1)	7×6×4 = 168 times	0 time
2)	6×4 = 24 times	24 times
3)	4 times	4 times

7.2 Move and Compare

7.2.1 CMP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction compares two values, and outputs the result (smaller, equal or larger) to bit devices (3 points).

→ **For the contact comparison instruction, refer to Section 7.22.**
→ **For floating point comparison, refer to Section 7.12.1.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
CMP	16 bits	Continuous	CMP EN ENO s1 d s2	CMP(EN,s1,s2,d);
CMPP	16 bits	Pulse	CMPP EN ENO s1 d s2	CMPP(EN,s1,s2,d);
DCMP	32 bits	Continuous	DCMP EN ENO s1 d s2	DCMP(EN,s1,s2,d);
DCMPP	32 bits	Pulse	DCMPP EN ENO s1 d s2	DCMPP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Data or device number handled as comparison value	ANY16	ANY32
	s2	Data or device number handled as comparison source	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head bit device to which comparison result is output.	ARRAY [0..2] OF Bit	

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others					
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
(s2)								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
(d)		●	●			●	▲1												●					

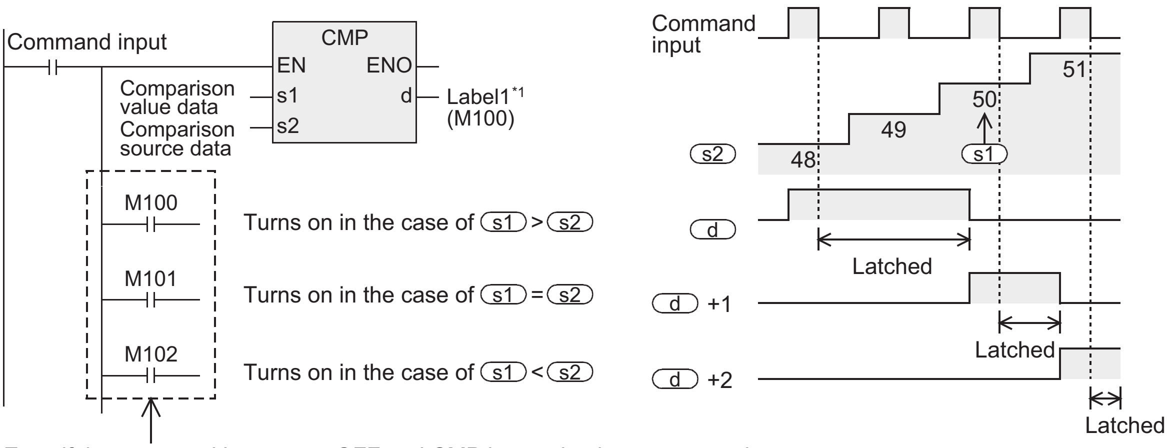
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(CMP, CMPP)

The comparison value specified by (s1) and the comparison source specified by (s2) are compared with each other. According to the result (smaller, equal or larger), any of the three points of the devices specified by (d) turns on.

- The source data specified by (s1) and (s2) are handled as BIN (binary) values.
- Comparison is executed algebraically. Example: -10 < 2



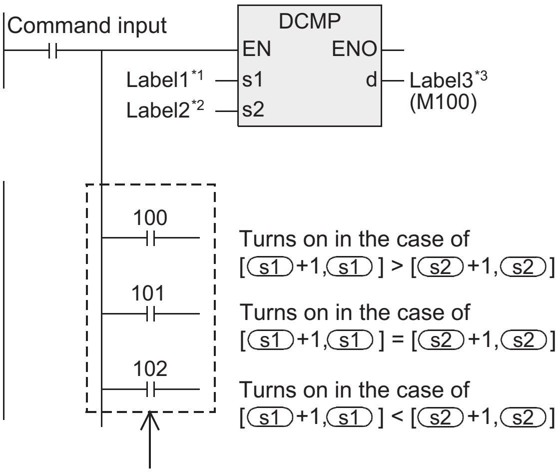
Even if the command input turns OFF and CMP instruction is not executed,
(d) to (d) +2 latch the status just before the command input turns OFF from ON.

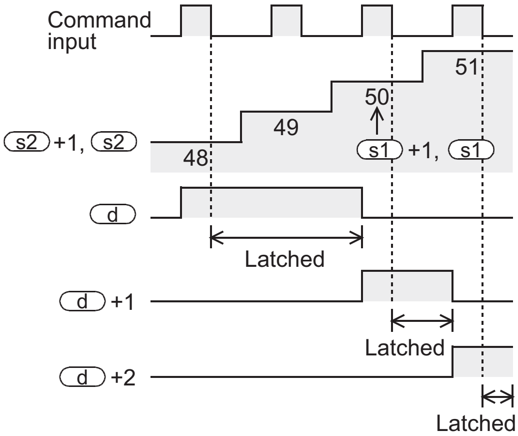
*1 This defines the head bit device that stores the comparison result. (Defines M100)

2. 32-bit operation(DCMP, DCMPP)

The comparison value specified by (s1) and the comparison source specified by (s2) are compared with each other. According to the result (smaller, equal or larger), any of the three points of the devices specified by (d) turns on.

- The source data specified by (s1) and (s2) are handled as BIN (binary) values.
- Comparison is executed algebraically. Example: -125400 < 22466





Even if the command input turns OFF and DCMP instruction is not executed, (d) to (d) +2 latch the status just before the command input turns OFF from ON.

*1 This defines comparison value data or the device that stores the comparison value data.
*2 This defines the comparison source data or the device that stores the comparison source data.
*3 This defines the head bit device that stores the comparison result. (Defines M100)

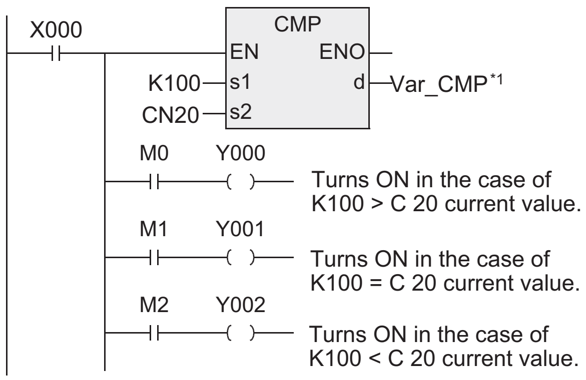
Cautions

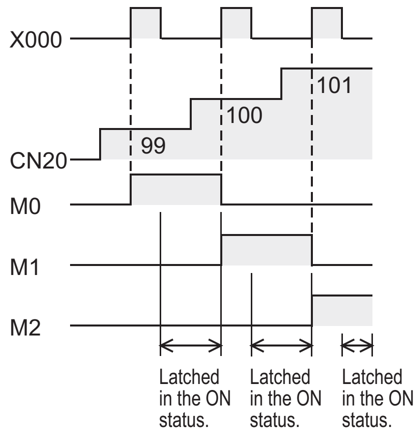
1) Some restrictions to applicable devices
▲1:The FX3U and FX3UC PLCs only are applicable. Not indexed (V,Z).
▲2:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲3:The FX3U and FX3UC PLCs only are applicable.
2) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle array data or 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
3) The FX0, FX0S or FX0N PLC does not support the pulse operation type instructions.
To execute pulse operation, make the instruction execution condition pulse type.
4) From the device specified as (d), three devices are occupied.
Be sure not to use those devices in another control.

Program examples

1. When comparing present value of a counter

[Structured ladder]





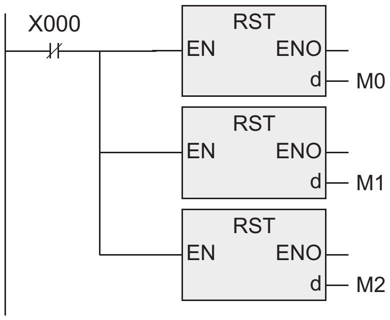

```
CMP(X000, K100, CN20, Var_CMP*1 );
Y000:=X000 AND M0;
Y001:=X000 AND M1;
Y002:=X000 AND M2;
```

*1 Var_CMP is a global label and is defined as M0.

If it is necessary to clear the comparison result when the instruction is not executed, add the following contents under the above program.

1) RST

[Structured ladder]

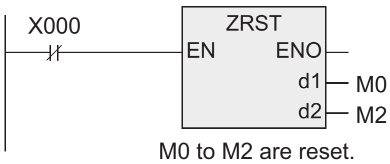


[ST]

```
RST(NOT X000, M0);
RST(NOT X000, M1);
RST(NOT X000, M2);
```

2) ZRST

[Structured ladder]



M0 to M2 are reset.

[ST]

```
ZRST(NOT X000, M0, M2);
```

7.2.2 ZCP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction compares two values (zone) with the comparison source, and outputs the result (upper, equal or lower) to bit devices (3 points).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ZCP	16 bits	Continuous	ZCP: EN → ENO; s1 → d; s2; s3	ZCP(EN,s1,s2,s3,d);
ZCPP	16 bits	Pulse	ZCPP: EN → ENO; s1 → d; s2; s3	ZCPP(EN,s1,s2,s3,d);
DZCP	32 bits	Continuous	DZCP: EN → ENO; s1 → d; s2; s3	DZCP(EN,s1,s2,s3,d);
DZCPP	32 bits	Pulse	DZCPP: EN → ENO; s1 → d; s2; s3	DZCPP(EN,s1,s2,s3,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Data or device handled as lower comparison value	ANY16	ANY32
	s2	Data or device handled as upper comparison value	ANY16	ANY32
	s3	Data or device number handled as comparison source	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head bit device to which comparison result is output.	ARRAY [0..2] OF Bit	

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others					
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
s2								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
s3								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
d		●	●			●	▲1												●					

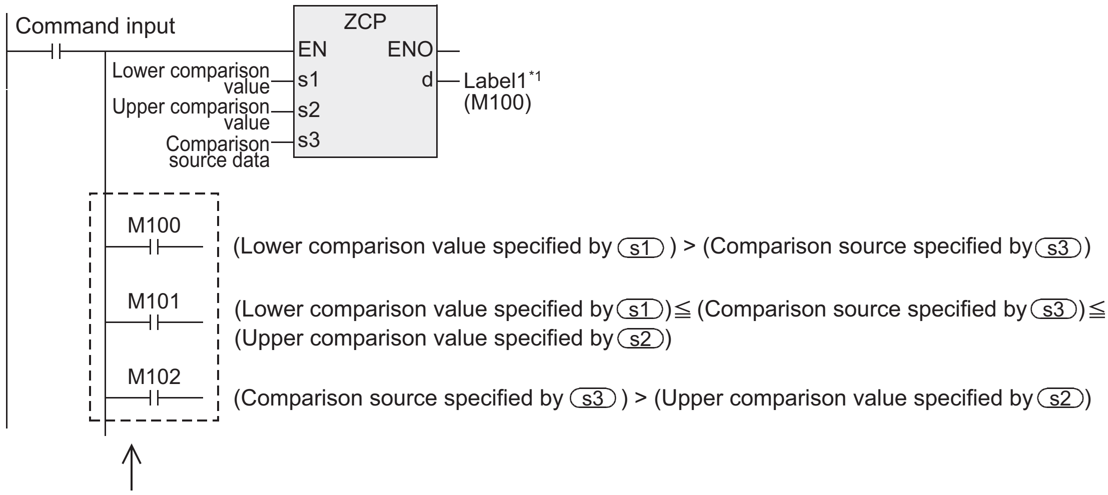
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(ZCP, ZCPP)

The lower comparison value specified by (s1) and the upper comparison value specified by (s2) are compared with the contents of the comparison source specified by (s3). According to the result (smaller, within zone or larger), any of the three points of the devices specified by (d) turns ON.

- Comparison is executed algebraically. Example: -10 < 2 < 10



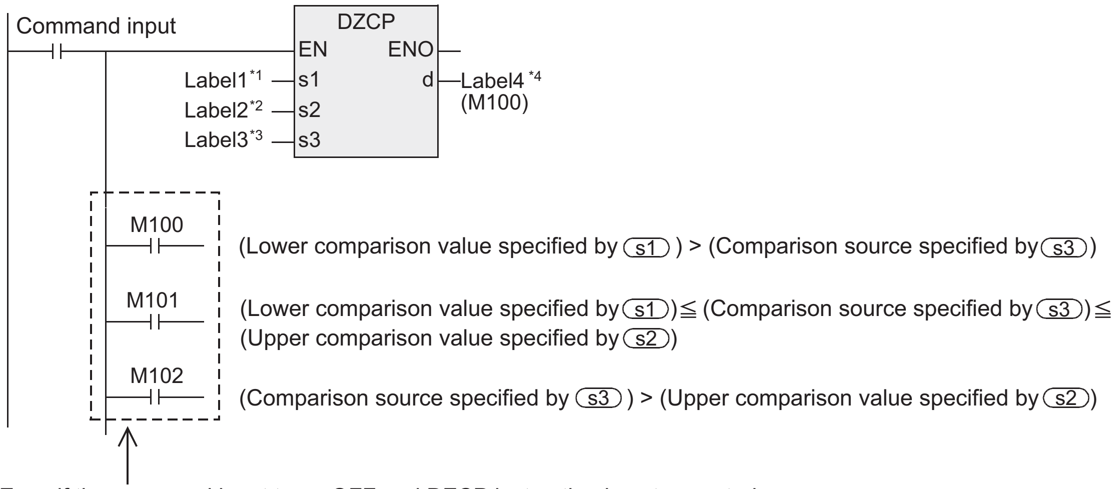
Even if the command input turns OFF and ZCP instruction is not executed,
(d) to (d)+2 latch the status just before the command input turns OFF from ON.

*1 This defines the head bit device that stores the comparison result. (Defines M100)

2. 32-bit operation(DZCP, DZCPP)

The lower comparison value specified by (s1) and the upper comparison value specified by (s2) are compared with the contents of the comparison source specified by (s3). According to the result (smaller, within zone or larger), any of the three points of the devices specified by (d) turns ON.

- Comparison is executed algebraically. Example: -125400 < 22466 < 1015444



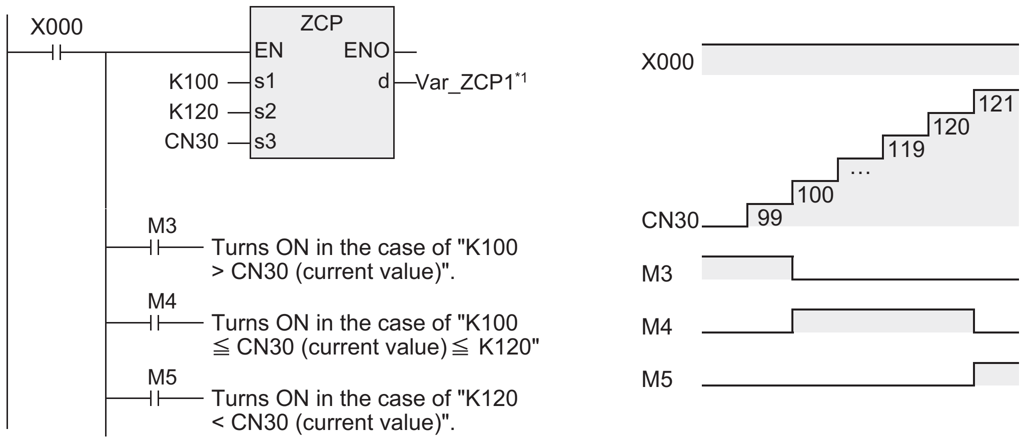
Even if the command input turns OFF and DZCP instruction is not executed,
(d) to (d)+2 latch the status just before the command input turns OFF from ON.

*1 This defines the lower comparison value data or the devices that stores the lower comparison value data.
*2 This defines the upper comparison value data or the devices that stores the upper comparison value data.
*3 This defines the comparison source data or the device that stores the comparison source data.
*4 This defines the head bit device that stores the comparison result. (Defines M100)

Cautions

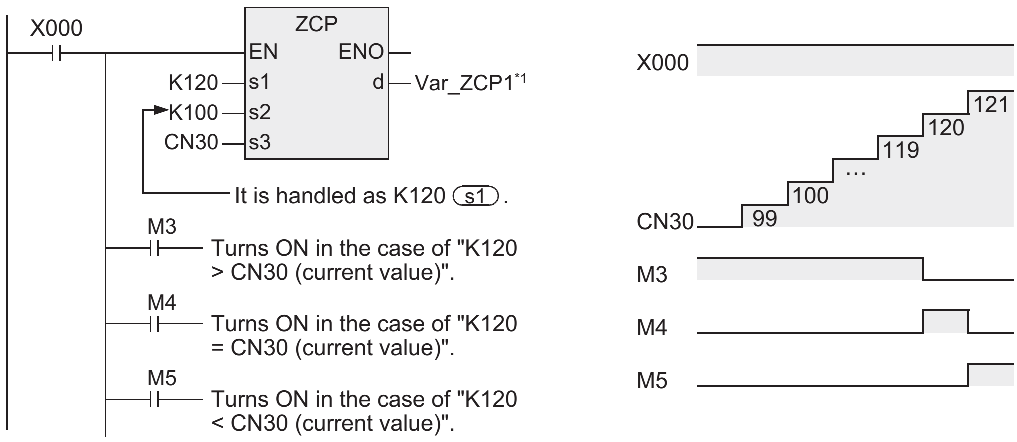
1) Some restrictions to applicable devices
   ▲1:The FX3U and FX3UC PLCs only are applicable. Not indexed (V,Z).
   ▲2:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲3:The FX3U and FX3UC PLCs only are applicable.
2) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle array data or 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
3) The FX0, FX0S or FX0N PLC does not support the pulse operation type instructions.
   To execute pulse operation, make the instruction execution condition pulse type.
4) From the device specified as (d), three devices are occupied.
   Be sure not to use those devices in another control.
5) The lower comparison value specified by (s1) should be smaller than the upper comparison value specified by (s2).

- When the lower comparison value (s1) is smaller than the upper comparison value (s2)



*1 Var_ZCP1 is a global label and is defined as M3.

- When the lower comparison value (s1) is larger than the upper comparison value (s2)



*1 Var_ZCP1 is a global label and is defined as M3.

7.2.3 MOV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction transfers (copies) the contents of a device to another device.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
MOV	16 bits	Continuous	MOV EN ENO s d	MOV(EN,s,d); Or an assignment statement
MOVP	16 bits	Pulse	MOVP EN ENO s d	MOVP(EN,s,d); Or an assignment statement
DMOV	32 bits	Continuous	DMOV EN ENO s d	DMOV(EN,s,d); Or an assignment statement
DMOVP	32 bits	Pulse	DMOVP EN ENO s d	DMOVP(EN,s,d); Or an assignment statement

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Data or device of transfer source	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Transfer destination device	ANY16	ANY32

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User			Special Unit		Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

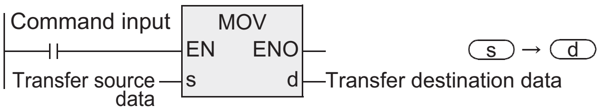
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(MOV, MOVP)

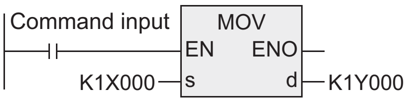
The contents of the transfer source specified by (s) are transferred to the transfer destination specified by (d).

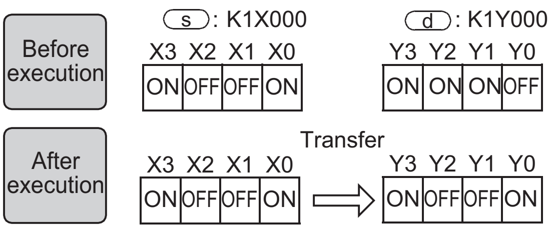
- While the command input is OFF, the transfer destination specified by (d) does not change.
- When a constant (K) is specified as the transfer source specified by (s), it is automatically converted into binary.

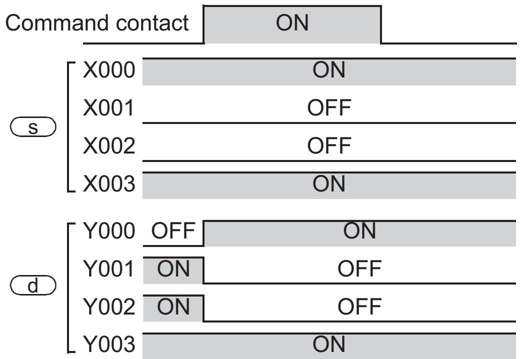


When specifying digits of a bit device (K1X000 → K1Y000)

The bit device transfers a maximum of 16 points (multiple of 4).

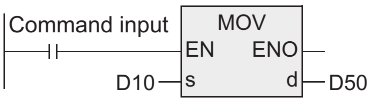


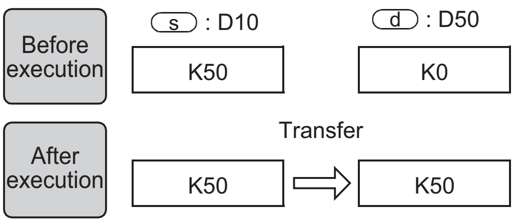


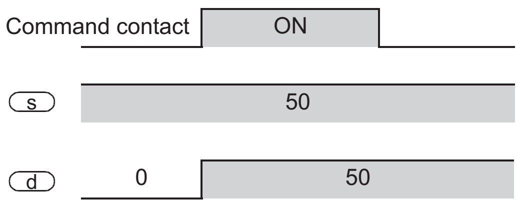


When a word device is specified

The word device transfers 1 point.



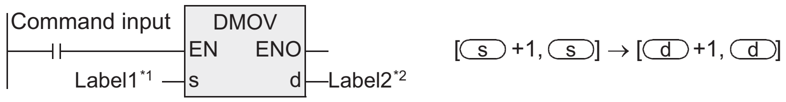




2. 32-bit operation(DMOV, DMOVP)

The contents of the transfer source specified by (s) are transferred to the transfer destination specified by (d).

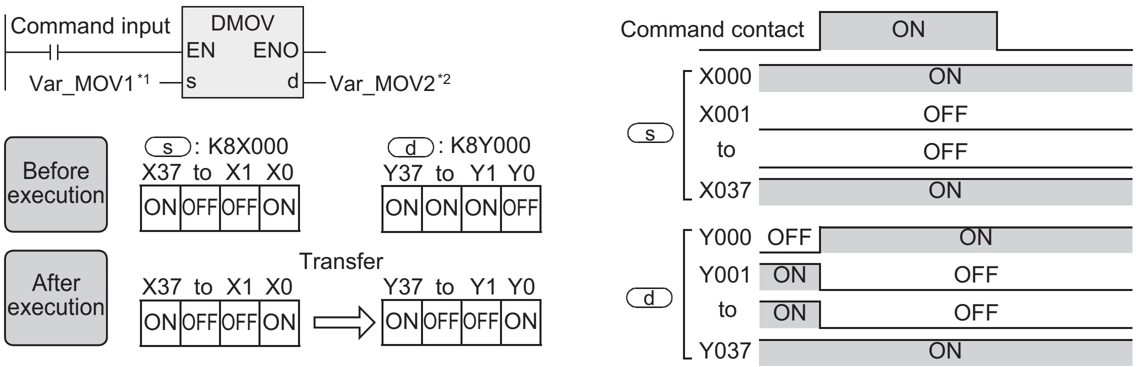
- While the command input is OFF, the transfer destination specified by (d) does not change.
- When a constant (K) is specified as the transfer source specified by (s), it is automatically converted into binary.



*1 This defines the transfer source data or the device that stores the transfer source data.
*2 This defines the transfer destination device.

When specifying digits of a bit device (K8X000 → K8Y000)

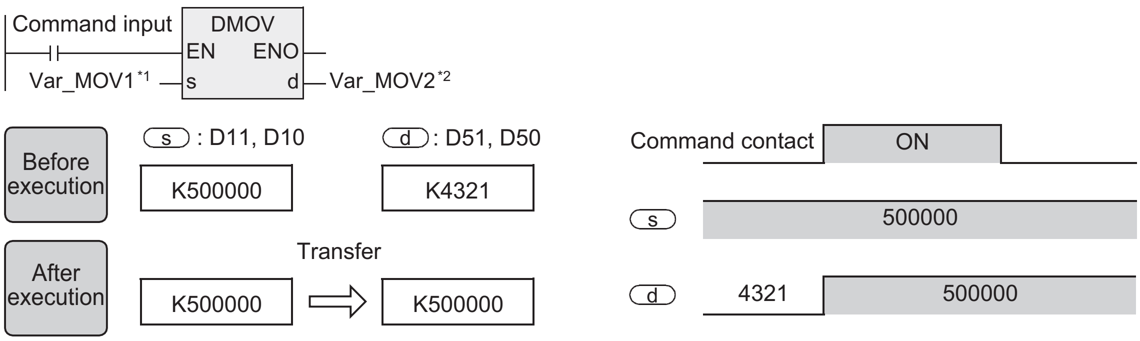
The bit device transfers a maximum of 32 points (multiple of 4).



*1 Var_MOV1 is a global label and is defined as K8X000.
*2 Var_MOV2 is a global label and is defined as K8Y000.

When a word device is specified

The word device transfers 1 point.



*1 Var_MOV1 is a global label and is defined as D10.
*2 Var_MOV2 is a global label and is defined as D50.

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.

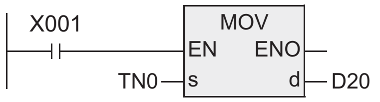
2) The FX0, FX0S or FX0N PLC does not support the pulse operation type instructions.
   To execute pulse operation, make the instruction execution condition pulse type.

3) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

1. When reading the current value of a timer and counter

[Structured ladder]



[ST]

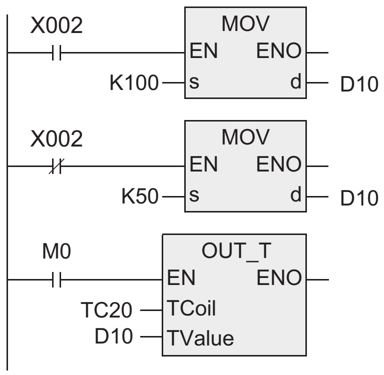
MOV(X001,TN0,D20);

(Current value of T0) → (D20)
The operation is the same as a counter.

2. When indirectly specifying the set value of a timer or counter

As the set value of the timer T20, two values can be specified by turning ON or OFF the switch X002. For specifying more than two set values, more than one switch is required.

[Structured ladder]



(K100) → (D 10)

(K50) → (D 10)

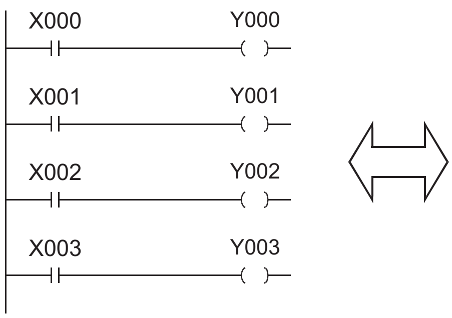
When X002 is ON, D 10=K100(10-second timer)
When X002 is OFF, D 10=K50(5-second timer)

[ST]

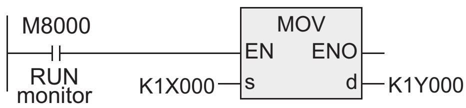
```
MOV(X002, K100, D10);
MOV(NOT X002, K50, D10);
OUT_T(M0, TC20, D10);
```

3. When transferring a bit device

The program written by basic instructions shown below can be expressed using MOV instruction.



[Structured ladder]



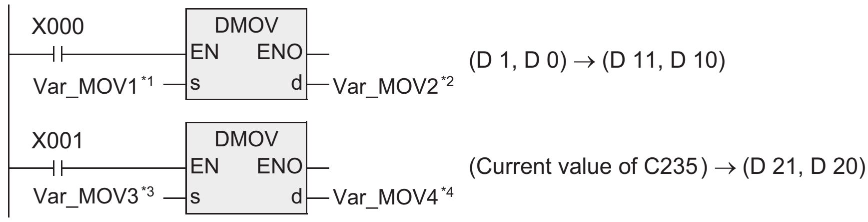
[ST]

MOV(M8000, K1X000, K1Y000);

4. When transferring 32-bit data

Be sure to use DMOV instruction for transferring an applied instruction (such as MUL) whose operation result is output in 32 bits, and for transferring a 32-bit numeric value or transferring the current value of a high speed counter (C235 to C255) which is a 32-bit device.

[Structured ladder]



[ST]

```
MOV(X000, Var_MOV1*1 , Var_MOV2*2 );
MOV(X001, Var_MOV3*3 , Var_MOV4*4 );
```

*1 Var_MOV1 is a global label and is defined as D0.
*2 Var_MOV2 is a global label and is defined as D10.
*3 Var_MOV3 is a global label and is defined as CN235.
*4 Var_MOV4 is a global label and is defined as D20.

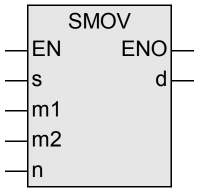
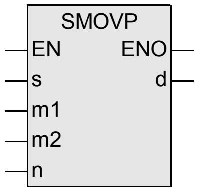
7.2.4 SMOV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction distributes and composes data in units of digit (4 bits).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SMOV	16 bits	Continuous		SMOV(EN,s,m1,m2,n,d);
SMOVP	16 bits	Pulse		SMOVP(EN,s,m1,m2,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Word device storing data whose digits will be moved.	ANY16
	(m1)	Head digit position to be moved	ANY16
	(m2)	Number of digits to be moved	ANY16
	(n)	Word device storing data whose digits are moved	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Head digit position of movement destination	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●					
(m1)																				●	●			
(m2)																				●	●			
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					
(n)																				●	●			

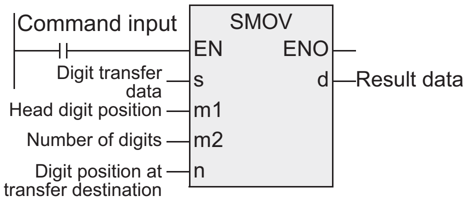
▲: Refer to "Cautions".

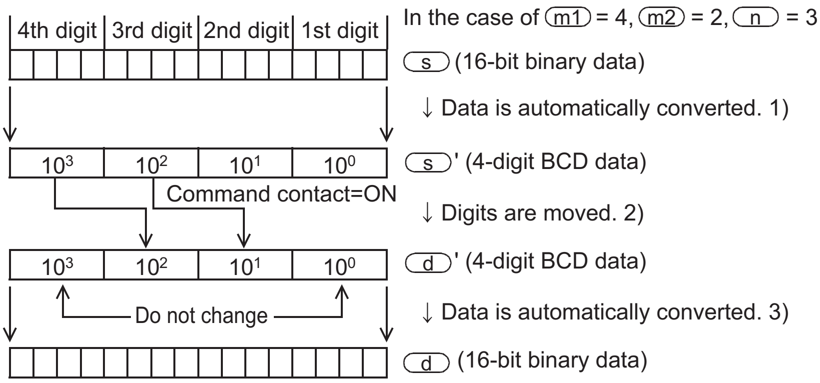
Function and operation explanation

1. 16-bit operation(SMOV, SMOVP)

The contents of the transfer source specified by (s) and transfer destination specified by (d) are converted into 4-digit BCD (0000 to 9999) respectively. "m2" digits starting from "m1"th digit are transferred (composed) to the transfer destination specified by (d) starting from "n"th digit, converted into binary, and then stored to the transfer destination specified by (d).

- While the command input is OFF, the transfer destination specified by (d) does not change.
- When the command input turns ON, the data of the transfer source specified by (s) and unspecified digits in the transfer destination specified by (d) do not change.

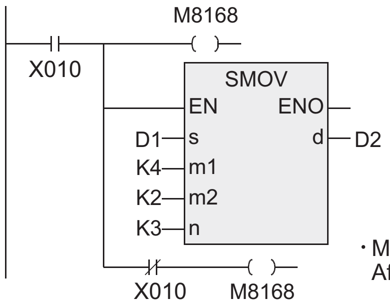




1) (s) is converted from binary into BCD.
2) "m2" digits starting from "m1"th digit are transferred (composed) to (d)' starting from "n"th digit. The digits of 10^3 and 10^0 of (d)' are not affected even if data is transferred from (s)'.
3) The composed data (BCD) is converted into binary, and stored to (d).

2. Extension function

When M8168 is set to ON first and then SMOV instruction is executed, conversion from binary to BCD is not executed.
Data is moved in units of 4 bits.



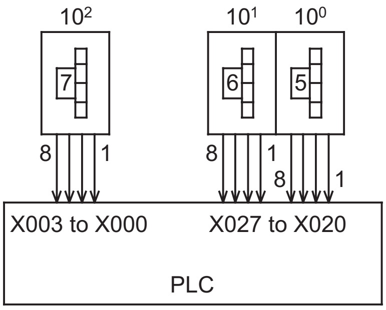
• M8168 is available for other instructions also.
After using M8168 for SMOV instruction, return it to OFF.

Cautions

1) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

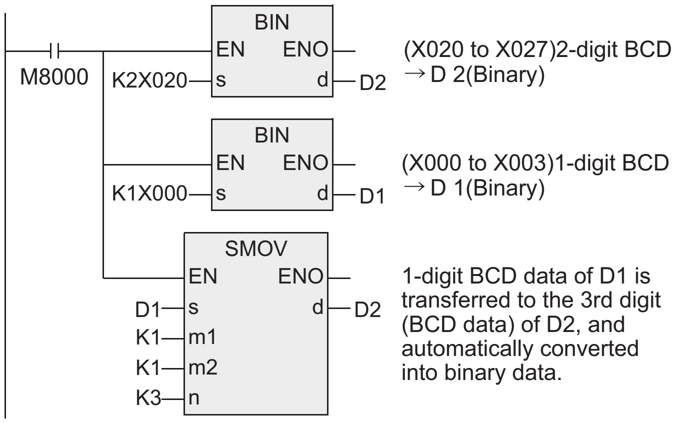
Program examples

The data on three-digit digital switches are composed, and stored as binary data to D2.



Data on three digital switches connected to non-consecutive input terminals are composed.

[Structured ladder]



[ST]

```
BIN(M8000, K2X020, D2);
BIN(M8000, K1X000, D1);
SMOV(M8000, D1, K1, K1, K3, D2);
```

7.2.5 CML

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction inverts data in units of bit, and then transfers (copies) the inverted data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
CML	16 bits	Continuous	CML: EN ENO, s d	CML(EN,s,d);
CMLP	16 bits	Pulse	CMLP: EN ENO, s d	CMLP(EN,s,d);
DCML	32 bits	Continuous	DCML: EN ENO, s d	DCML(EN,s,d);
DCMLP	32 bits	Pulse	DCMLP: EN ENO, s d	DCMLP(EN,s,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Data to be inverted or word device storing the data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Destination word device storing inverted data	ANY16	ANY32

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

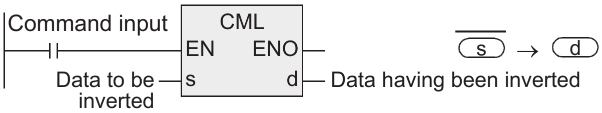
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(CML, CMLP)

Each bit of a device specified by (s) is inverted (from 0 to 1 or from 1 to 0), and then transferred to the device specified by (d).

- When a constant (K) is specified as (s), it is automatically converted into binary.
- This operation is useful when a logically inverted output is required as an output from a PLC.



	b15	b14	b13	b12	b11	b11	b9	b8	b7	b6	n5	b4	b3	b2	b1	b0
(s)	1	0	1	0	1	0	1	0	0	0	1	1	1	0	1	0
(d)	0	1	0	1	0	1	0	1	1	1	0	0	0	1	0	1

sign bit (0=Positive number 1=Negative number)

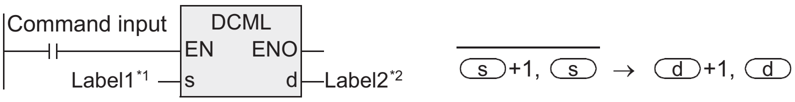
Command contact=ON

Inverted data is transferred.

2. 32-bit operation(DCML, DCMLP)

Each bit of a device specified by (s) is inverted (from 0 to 1 or from 1 to 0), and then transferred to the device specified by (d).

- When a constant (K) is specified as (s), it is automatically converted into binary.
- This operation is useful when a logically inverted output is required as an output from a PLC.



*1 This defines the data to be inverted or the word device that stores the data.
*2 This defines the word device that stores the inverted data.

	b31	b30	b29	b28	b27	b26	b25	to	b7	b6	n5	b4	b3	b2	b1	b0
(s)	1	0	1	0	1	0	1		0	0	1	1	1	0	1	0
(d)	0	1	0	1	0	1	0		1	1	0	0	0	1	0	1

sign bit (0=Positive number 1=Negative number)

Command contact=ON

Inverted data is transferred.

Cautions

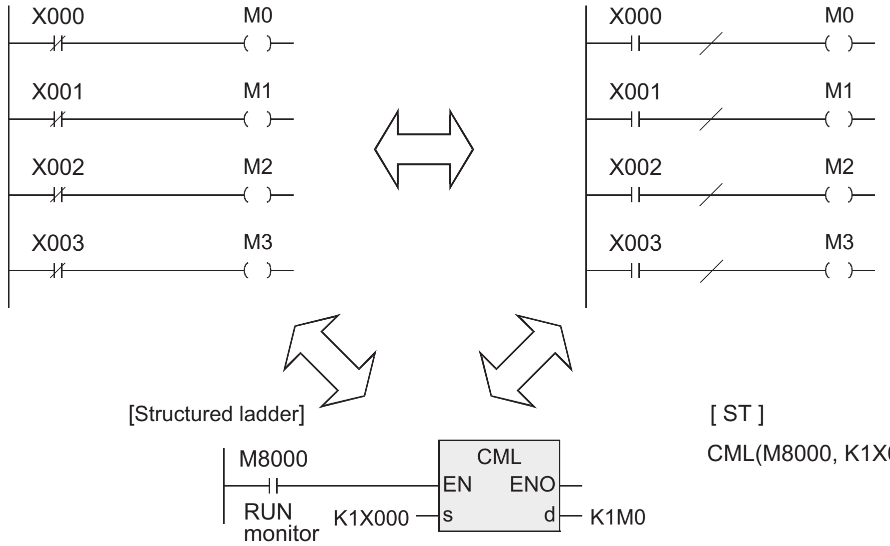
1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

2) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

1. When receiving an inverted input

The sequence program shown below can be written by CML instruction.

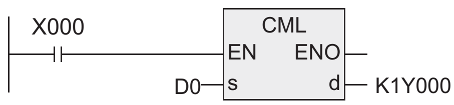


[ST]

CML(M8000, K1X000, K1M0);

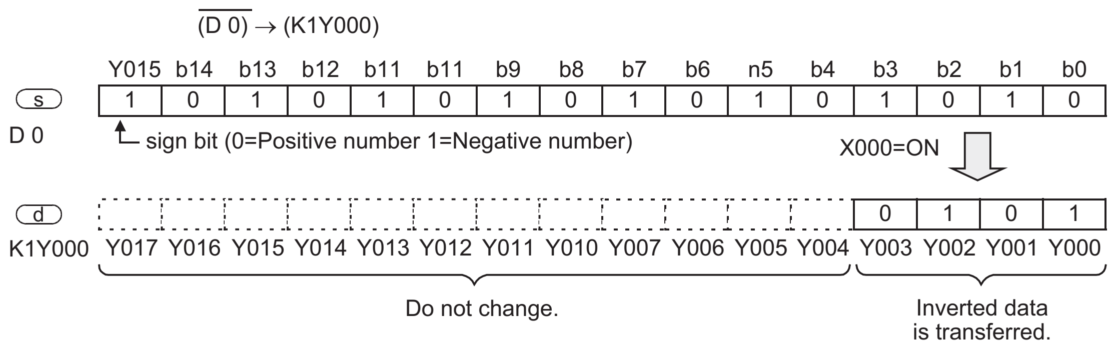
2. When four bits are specified for a device with digit specification

[Structured ladder]



[ST]

CML(X000, D0, K1Y000);



7.2.6 BMOV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	×

Outline

This instruction transfers (copies) a specified number of data at one time.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BMOV	16 bits	Continuous	BMOV EN ENO s d n	BMOV(EN,s,n,d);
BMOVP	16 bits	Pulse	BMOVP EN ENO s d n	BMOVP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Transfer source device	ANY16
	n	Number of transferred points (including file registers)	ANY16
Output variable	ENO	Execution state	Bit
	d	Transfer destination device	ANY16

3. Applicable devices

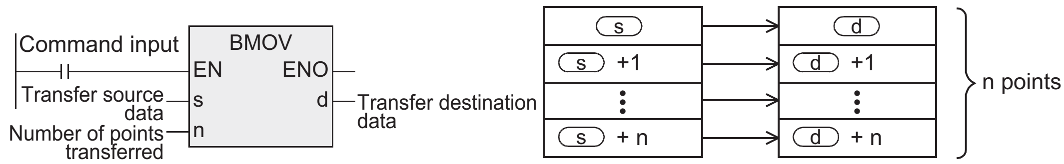
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲1	▲2			●					
d									●	●	●	●	●	●	▲1	▲2			●					
n														●						●	●			

▲: Refer to "Cautions".

Function and operation explanation

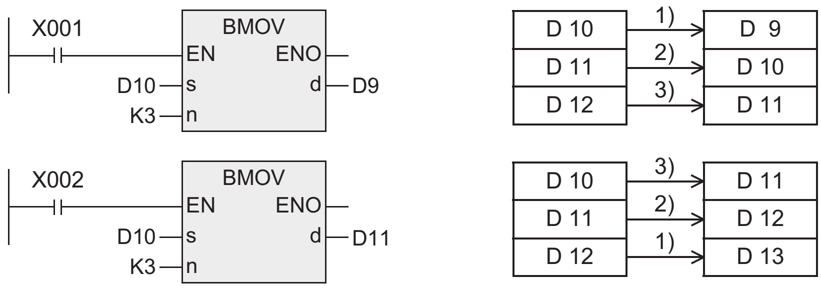
BMOV instruction transfers "n" points of data from the device specified by (s) to the device specified by (d) at one time.

- If the device number range is exceeded, data is transferred within the possible range.



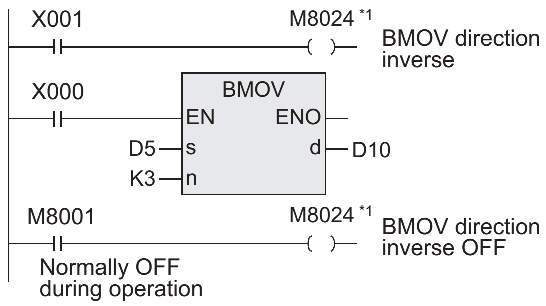
1. Transfer is enabled even if the transfer number range is overlapped.

To prevent overwriting before transfer of source data, data is automatically transferred in the order "1)→2)→3)" according to the number overlap status.



2. Extension function (bi-directional transfer function)

By controlling the direction inverse flag M8024*1 for BMOV instruction, data can be transferred in two directions in one program.



BMOV direction inverse flag	Transfer direction	(s), (d)
M8024*1:OFF	(s) → (d)	D5→D10 D6→D11 D7→D12
M8024*1:ON	(s) ← (d)	D5←D10 D6←D11 D7←D12

*1. M8024 is cleared when the PLC mode is changed from RUN to STOP.

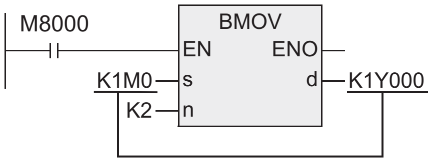
Cautions

1) The FX0N, FXU and FX2C PLCs handle file registers as follows.

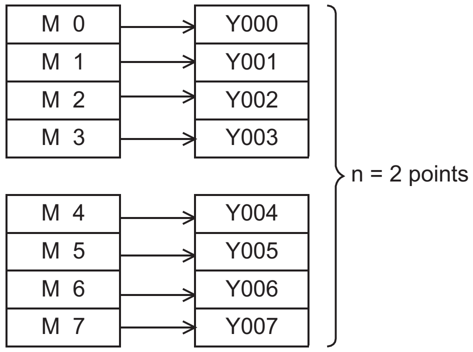
	BMOV instruction	
	Read	Write
FX0N	✓	
FXU FX2C	✓	✓
FXU (V2.30 or earlier)	✓	

2) The FX0N PLC does not support the instructions of pulse operation type.
To execute pulse operation, make the instruction execution condition pulse type.

3) When specifying digits of bit devices, specify a same number of digits for s and d.



Specify a same number of digits.
(Example: K1)



4) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

Function of transfer between file registers and data registers

BMOV instruction has a special function to file registers (D1000 and later).
The maximum number of file register differs from one PLC to another.
This explanation here uses the FX3U and FX3UC PLCs as examples.
For the details of the file registers, refer to the following manual.

→ **FX Structured Programming Manual (Device & Common)**

1. What are file registers

By parameter setting, D1000 to D7999 can be handled as file registers, and written to and read from the program memory area.

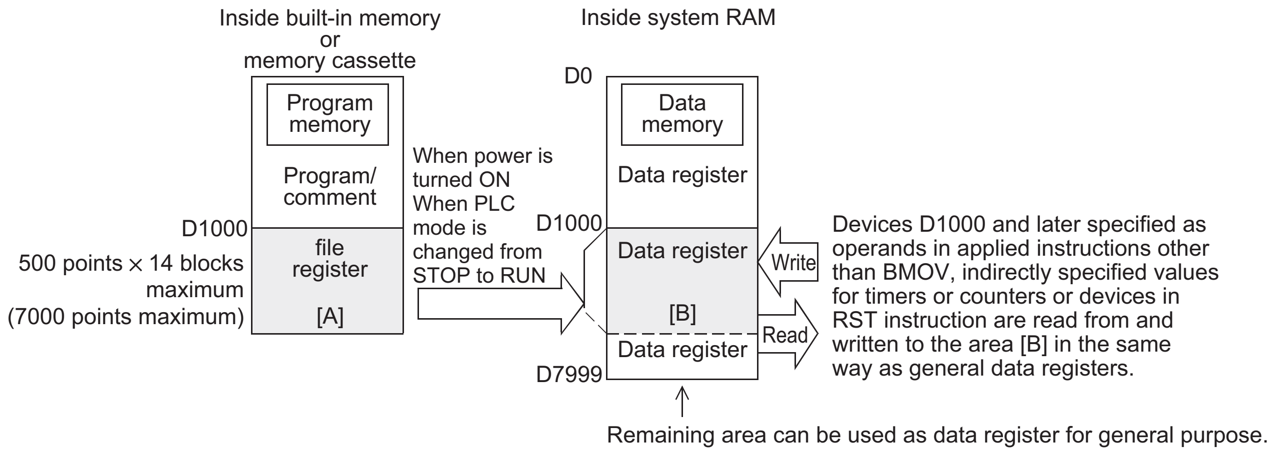
1) Outline of setting
   File registers (D1000 to D7999) do not exist in the initial status. They are valid only when some number of file registers are secured by parameter setting in a programming tool.

2) Number of file registers
   In parameter setting, set 500 file registers as 1 block.
   1 to 14 blocks (each of which has 500 file registers) can be set.
   1 block occupies 500 steps in the program memory area.

3) Difference between BMOV instruction and other instructions
   The table below shows the difference between BMOV instruction and other instructions with regard to file registers (D1000 and later).

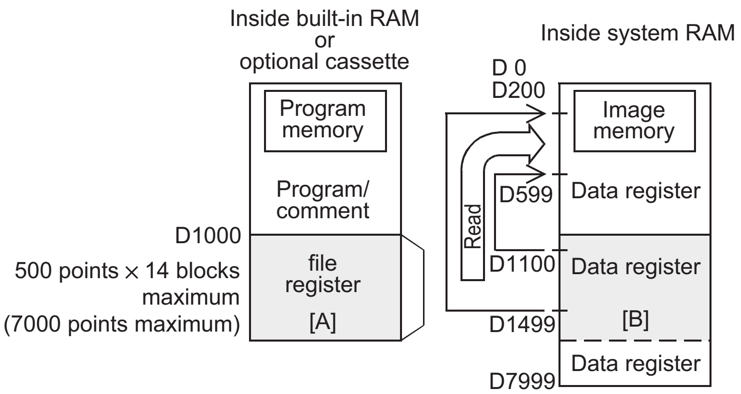
Instruction	Contents of transfer	Remarks
BMOV	Can read from and write to the file register area [A] inside the program memory.	-
Other applied instructions than BMOV	Can read from and write to the data register area [B] inside the image memory in the same way as general data registers.	Because the data register area [B] is provided inside the system RAM in PLCs, its contents can be arbitrarily changed without regard to the optional memory format.

When power is turned ON, data registers set as file registers are automatically copied from the file register area [A] to the data register area [B].



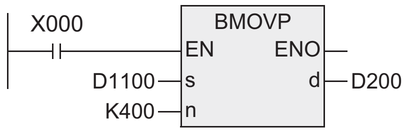
2. Cautions on use

1) When updating the contents of a file register with a same number (same-number update mode), make sure that the file register number is equivalent between (s) and (d).
2) When using file registers in the same-number update mode, make sure that the number of transfer points specified by "n" does not exceed the file register area.
3) If the file register area is exceeded while file registers are used in the same-number update mode, an operation error (M8067) is caused and the instruction is not executed.
4) In the case of indexing (in the same-number update mode)
   When (s) and (d) are modified with index, the instruction is executed if the actual device number is within the file register area and the number of transfer points does not exceed the file register area.
5) Handling of flash memory
   When changing the contents of file registers secured inside the flash memory, observe the following condition:
   - Set the protect switch to OFF in the optional memory.
   - When writing data using a continuous operation type instruction in a program, data is written to the flash memory in every operation cycle of the PLC.
     To prevent this, as the flash memory has a limit to the number of times of writing operations, be sure to use a pulse operation type instruction (BMOVP) so that the number of times of writing is reduced.
   - It takes 66 to 132 ms to write data of one serial block (500 points) to the flash memory.
     Execution of the program is paused during this period. Because the watchdog timer is not refreshed at this time, it is necessary to take proper countermeasures such as insertion of WDT instruction in a user program.
6) File register operation
   File registers are secured inside the built-in memory or memory cassette.
   Different from general data registers, file registers can be read and written directly only by peripheral equipment or BMOV instruction.
7) If a file register is not specified as the destination in BMOV instruction, the file register is not accessed.
   a) Outline of memory operation



   b) Program examples
   When X000 is set to ON, the data register area [B] is read.

[Structured ladder]



[ST]

```
BMOVP(X000, D1100, K400, D200);
```

A file register can be specified as (d). But if a same number with (s) is specified, the same-number register update mode is selected.
However, even if a file register having different number is specified for (s) and (d) respectively, data cannot be transferred from the file register area to another file register area. In such a case, read the contents of a file register specified as (s) in the same-number register update mode to the data register area [B] once, and then write the data.

7.2.7 FMOV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction transfers same data to specified number of devices.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
FMOV	16 bits	Continuous	FMOV EN ENO s d n	FMOV(EN,s,n,d);
FMOVP	16 bits	Pulse	FMOVP EN ENO s d n	FMOVP(EN,s,n,d);
DFMOV	32 bits	Continuous	DFMOV EN ENO s d n	DFMOV(EN,s,n,d);
DFMOVP	32 bits	Pulse	DFMOVP EN ENO s d n	DFMOVP(EN,s,n,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Transfer source data or device storing data	ANY16	ANY32
	(n)	Number of transfer points	ANY16	
Output variable	ENO	Execution state	Bit	
	(d)	Head word device of transfer destination (Same data is transferred from the transfer source at one time.)	ANY16	ANY32

3. Applicable devices

Operand type	Bit Devices (System User)							Word Devices (Digit Specification)				Word Devices (System User)				Word Devices (Special Unit)	Word Devices (Index)			Others (Constant)		Others (Real Number)	Others (Character String)	Others (Pointer)
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2			●					
(n)																				●	●			

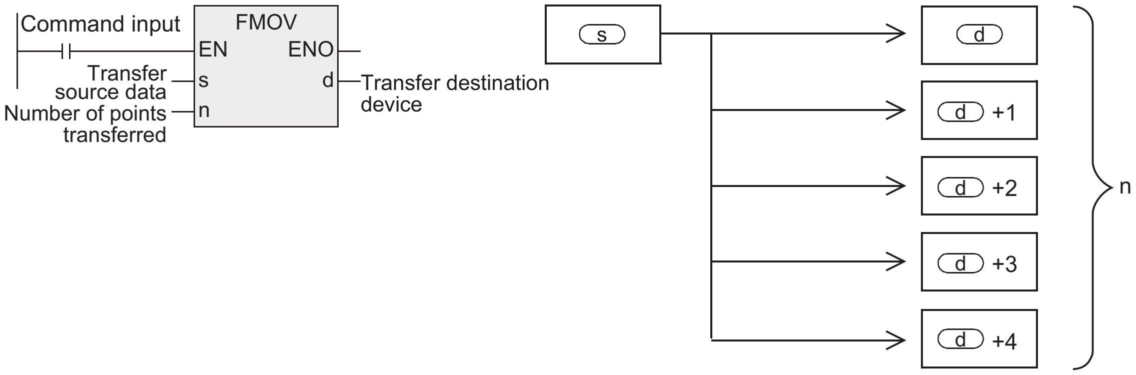
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(FMOV, FMOVP)

The data or contents of a device specified by (s) are transferred to "n" devices starting from a device specified by (d).

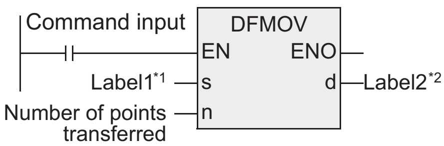
- The contents will be same among all of "n" devices.
- If the number of points specified by "n" exceeds the device number range, data is transferred within the possible range.
- While the command input is OFF, the transfer destination specified by (d) does not change.
- While the command input is ON, the data of the transfer source specified by (s) does not change.
- When a constant (K) is specified as the transfer source specified by (s), it is automatically converted into binary.



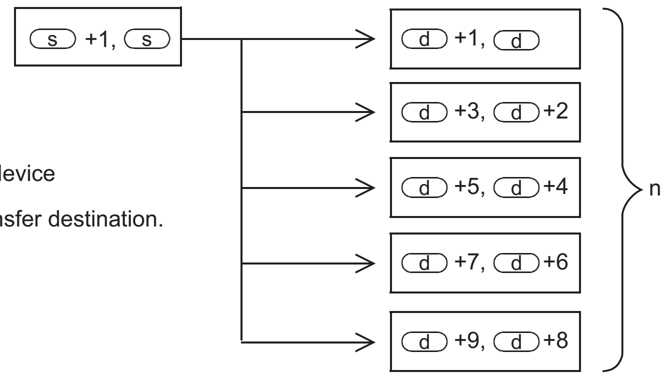
2. 32-bit operation(DFMOV, DFMOVP)

The contents of the transfer source specified by (s) are transferred to "n" 32-bit devices starting from the device specified by (d).

- The contents will be the same among all of "n" 32-bit devices.
- If the number of points specified by "n" exceeds the device number range, data is transferred within the possible range.
- While the command input is OFF, the transfer destination specified by (d) does not change.
- While the command input is ON, the data of the transfer source specified by (s) does not change.
- When a constant (K) is specified as the transfer source specified by (s), it is automatically converted into binary.



*1 This defines the transfer source data or the device that stores the transfer source data.
*2 This defines the head word device at the transfer destination.



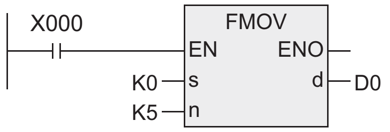
Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) The FXU PLC of V2.30 or earlier does not support 32-bit instructions.
3) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

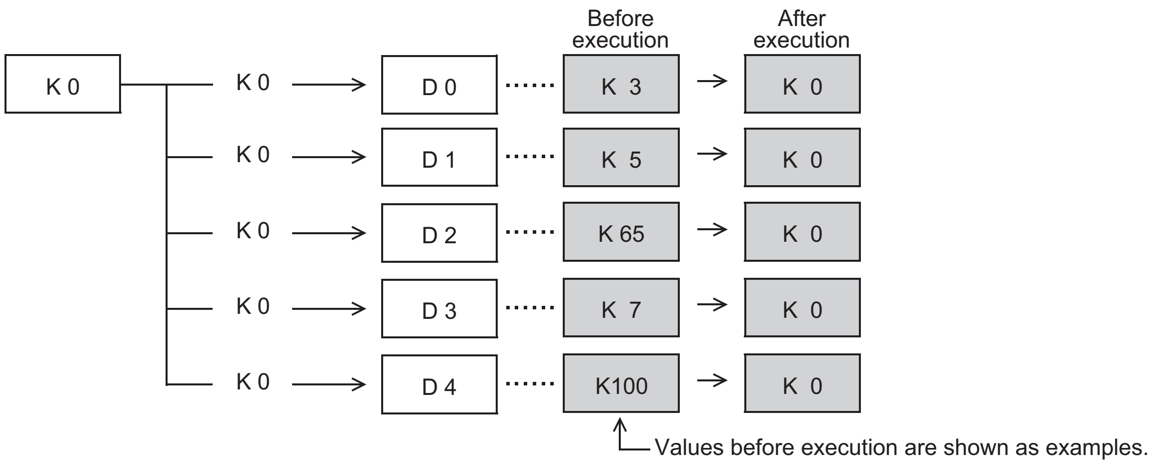
1. When writing specified data to two or more devices

[Structured ladder]



[ST]

```
FMOV(X000, K0, K5, D0);
```



7.2.8 XCH

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This instruction exchanges data between two devices.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
XCH	16 bits	Continuous	XCH EN ENO d1 d2	XCH(EN,d1,d2);
XCHP	16 bits	Pulse	XCHP EN ENO d1 d2	XCHP(EN,d1,d2);
DXCH	32 bits	Continuous	DXCH EN ENO d1 d2	DXCH(EN,d1,d2);
DXCHP	32 bits	Pulse	DXCHP EN ENO d1 d2	DXCHP(EN,d1,d2);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
Output variable	ENO	Execution state	Bit	
	(d1)	Device storing data to be exchanged.	ANY16	ANY32
	(d2)	Device storing data to be exchanged.	ANY16	ANY32

3. Applicable devices

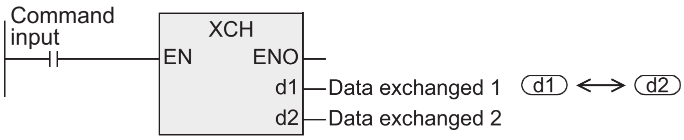
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d1)									●	●	●	●	●	●	▲1	▲1	●	●	●					
(d2)									●	●	●	●	●	●	▲1	▲1	●	●	●					

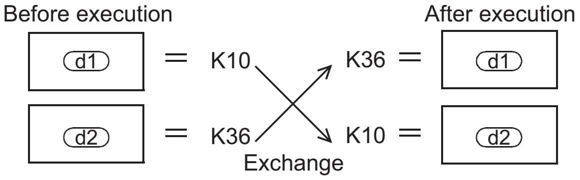
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(XCH, XCHP)

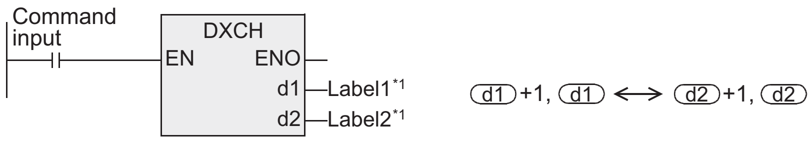
Data is exchanged between the device specified by d1 and the device specified by d2.



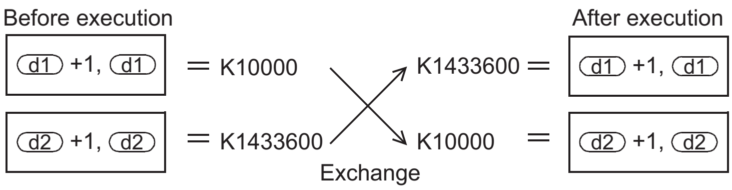


2. 32-bit operation(DXCH, DXCHP)

Data is exchanged between the device specified by d1 and the device specified by d2.

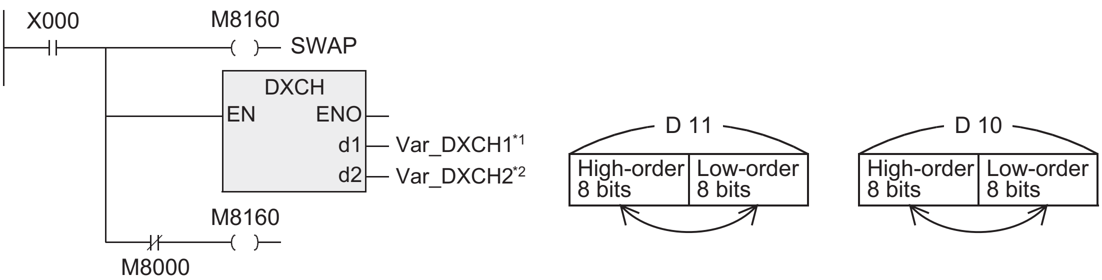


*1 This defines the device that stores the data to be exchanged.



Extension function

When the instruction is executed while M8160 is ON, high-order 8 bits (byte) and low-order 8 bits (byte) of a word device are exchanged each other.
(The FXU PLC of V2.30 or earlier does not support the extension function.)
This is the same operation as SWAP instruction, so use SWAP instruction for newly programming.
In the case of 32-bit operation, high-order 8 bits (byte) and low-order 8 bits (byte) of a word device are changed.



*1 Var_DXCH1 is a global label and is defined as D10.
*2 Var_DXCH2 is a global label and is defined as D10.

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) Some restrictions to applicable devices
   ▲1:The FX3U and FX3UC PLCs only are applicable.

Error

An operation error occurs in the following case. The error flag M8067 turns ON, and the error code is stored in D8067.

- When M8160 is ON, and the device numbers specified by d1 and d2 are different.

7.2.9 BCD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction converts binary (BIN) data into binary-coded decimal (BCD) data.
Binary data is used in operations in PLCs. Use this instruction to display numeric values on the seven-segment display unit equipped with BCD decoder.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BCD	16 bits	Continuous	BCD EN ENO s d	BCD(EN,s,d);
BCDP	16 bits	Pulse	BCDP EN ENO s d	BCDP(EN,s,d);
DBCD	32 bits	Continuous	DBCD EN ENO s d	DBCD(EN,s,d);
DBCDP	32 bits	Pulse	DBCDP EN ENO s d	DBCDP(EN,s,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Word device storing the conversion source (binary) data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Word device of the conversion destination (binary-coded decimal) data	ANY16	ANY32

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲1	▲2	●	●	●					
d									●	●	●	●	●	●	▲1	▲2	●	●	●					

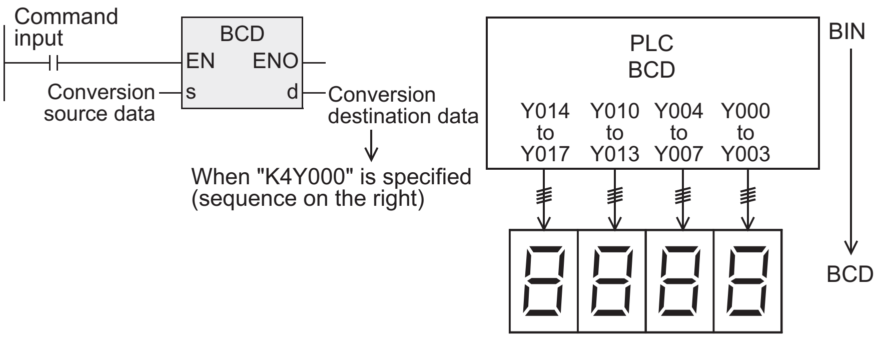
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(BCD, BCDP)

This instruction converts the binary (BIN) data specified by (s) into binary-coded decimal (BCD) data, and transfers the converted BCD data to the device specified by (d).

- The data of the device specified by (s) can be converted if it is within the range from K0 to K9999 (BCD).
- The table below shows digit specification for the devices specified by (s) and (d) respectively.

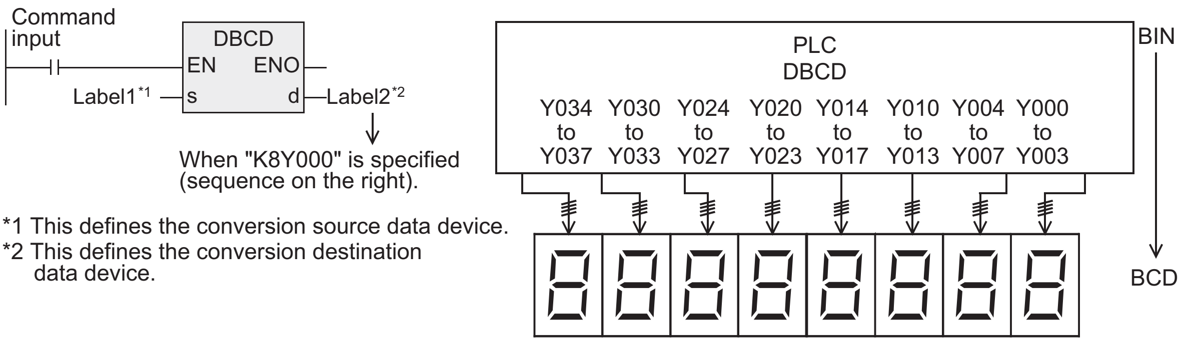


(d)	Number of digits	Data range
K1Y000	1	0 to 9
K2Y000	2	00 to 99
K3Y000	3	000 to 999
K4Y000	4	0000 to 9999

2. 32-bit operation(DBCD, DBCDP)

This instruction converts the binary (BIN) data specified by (s) into binary-coded decimal (BCD) data, and transfers the converted BCD data to the device specified by (d).

- The data of the device specified by (s) can be converted if it is within the range from K0 to K99999999 (BCD).
- The table below shows digit specification for the devices specified by (s) and (d) respectively.

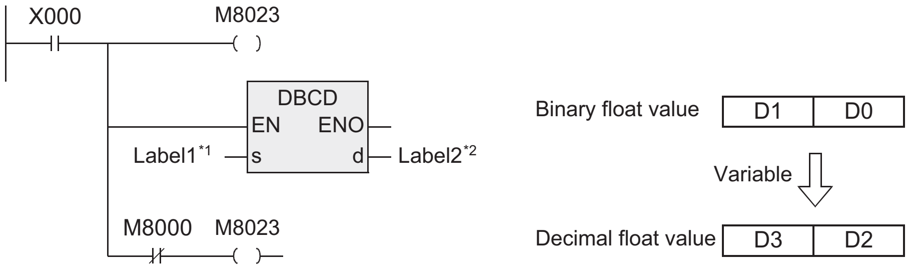


*1 This defines the conversion source data device.
*2 This defines the conversion destination data device.

[(d)+1, (d)]	Number of digits	Data range
K1Y000	1	0 to 9
K2Y000	2	00 to 99
K3Y000	3	000 to 999
K4Y000	4	0000 to 9999
K5Y000	5	00000 to 99999
K6Y000	6	000000 to 999999
K7Y000	7	0000000 to 9999999
K8Y000	8	00000000 to 99999999

Extension function(FXU and FX2C PLCs)

The FXU PLC of V2.30 or earlier does not support the extension function.
When executing the instruction with M8023 ON, conversion takes place from binary float to decimal float.



*1 This defines D0.
*2 This defines D2.

For the float conversion, only data register (D) is applicable as the device for s and d.

Related instruction

Instruction	Function
BIN	Converts binary-coded decimal (BCD) data into binary (BIN) data.

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
3) Because conversion between binary-coded decimal data and binary data is automatically executed in SEGL and ARWS instructions, BCD instruction is not required.
4) Binary data is used in all operations in PLCs including arithmetic operations (+, -, × and ÷), increment and decrement instructions.

- When receiving the digital switch information in the binary-coded decimal (BCD) format into a PLC, use BIN instruction for converting BCD data into binary data.
- When outputting data to the seven-segment display unit handling binary-coded decimal (BCD) data, use BCD instruction for converting binary data into BCD data.

5) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

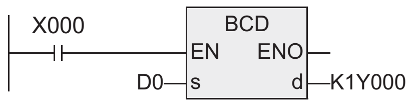
Error

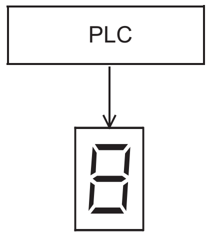
In BCD or BCDP (16-bit type) instructions, an operation error occurs when the s value is outside the range from 0 to 9,999.
In DBCD or DBCDP (32-bit type) instructions, an operation error occurs when the s value is outside the range from 0 to 99,999,999.

Program examples

1. When the seven-segment display unit has 1 digit

[Structured ladder]



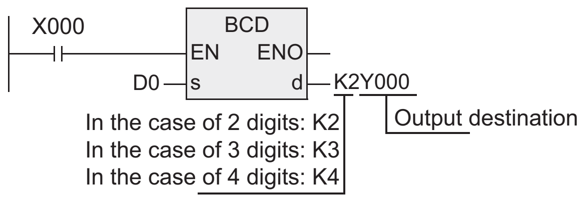


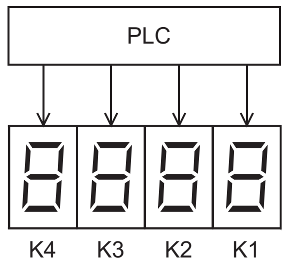
[ST]

BCD(X000, D0, K1Y000);

2. When the seven-segment display unit has 2 to 4 digits

[Structured ladder]



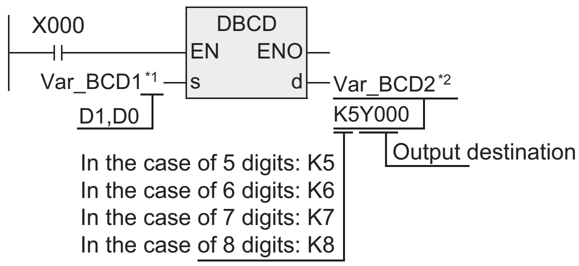


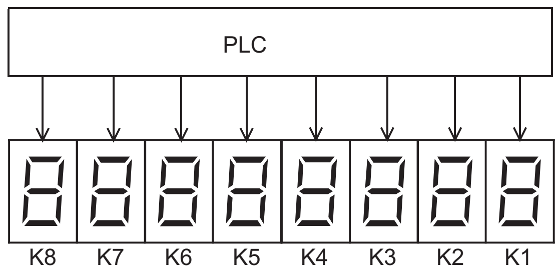
[ST]

BCD (X000, D0, K2Y000);

3. When the seven-segment display unit has 5 to 8 digits

[Structured ladder]





[ST]

DBCD (X000, Var_BCD1*1 , Var_BCD2*2);

*1 Var_BCD1 is a global label and is defined as D0.
*2 Var_BCD2 is a global label and is defined as K5Y000.

7.2.10 BIN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction converts binary-coded decimal (BCD) data into binary (BIN) data.
Use this instruction to convert a binary-coded decimal (BCD) value such as a value set by a digital switch into binary (BIN) data and to receive the converted binary data so that the data can be handled in operations in PLCs.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BIN	16 bits	Continuous	BIN —EN ENO— —s d—	BIN(EN,s,d);
BINP	16 bits	Pulse	BINP —EN ENO— —s d—	BINP(EN,s,d);
DBIN	32 bits	Continuous	DBIN —EN ENO— —s d—	DBIN(EN,s,d);
DBINP	32 bits	Pulse	DBINP —EN ENO— —s d—	DBINP(EN,s,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Word device storing the conversion source (binary-coded decimal) data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Word device of the conversion destination (binary)	ANY16	ANY32

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●					
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

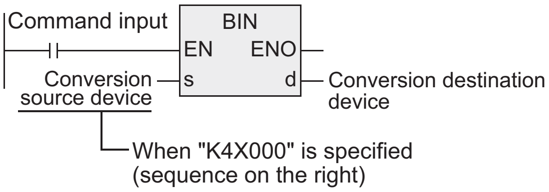
▲: Refer to "Cautions".

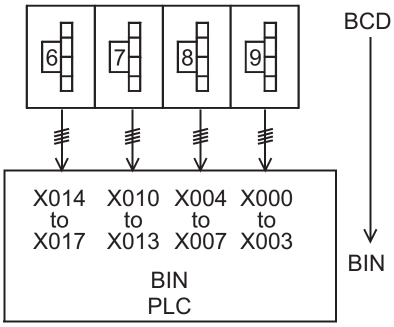
Function and operation explanation

1. 16-bit operation(BIN, BINP)

This instruction converts the binary-coded decimal (BCD) data specified by (s) into binary (BIN) data, and transfers the converted binary data to the device specified by (d).

- The data of the device specified by (s) can be converted if it is within the range from 0 to 9999 (BCD).
- The table below shows digit specification for the devices specified by (s) and (d).



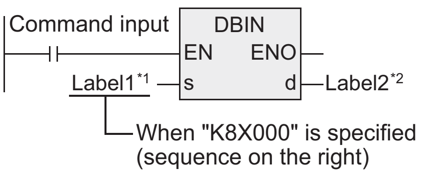


(s)	Number of digits	Data range
K1X000	1	0 to 9
K2X000	2	00 to 99
K3X000	3	000 to 999
K4X000	4	0000 to 9999

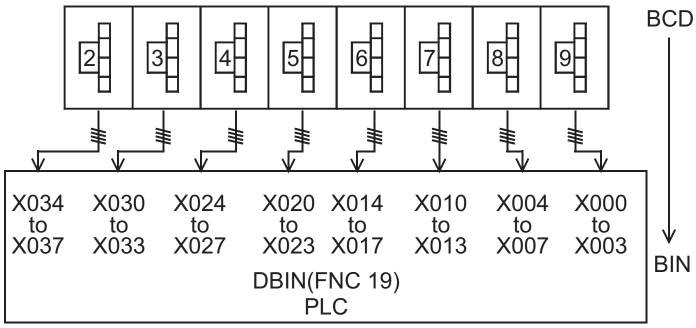
2. 32-bit operation(DBIN, DBINP)

This instruction converts the binary-coded decimal (BCD) data specified by (s) into binary (BIN) data, and transfers the converted binary data to the device specified by (d).

- The data of the device specified by (s) can be converted if it is within the range from 0 to 99999999 (BCD).
- The table below shows digit specification for the devices specified by (s) and (d).



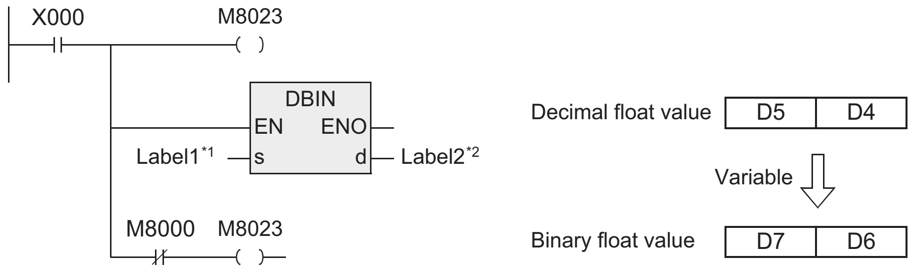
*1 This defines the transfer source device.
*2 This defines the transfer destination device.



(s)+1, (s)	Number of digits	Data range
K1X000	1	0 to 9
K2X000	2	00 to 99
K3X000	3	000 to 999
K4X000	4	0000 to 9999
K5X000	5	00000 to 99999
K6X000	6	000000 to 999999
K7X000	7	0000000 to 9999999
K8X000	8	00000000 to 99999999

Extension function(FXU and FX2C PLCs)

The FXU PLC of V2.30 or earlier does not support the extension function.
When executing the instruction with M8023 ON, conversion takes place from binary-coded decimal float to binary float.



*1 This defines D4.
*2 This defines D6.

For the float conversion, only data register (D) is applicable as the device for s and d.

Related instruction

Instruction	Function
BCD	Converts binary (BIN) data into binary-coded decimal (BCD) data.

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
3) Because conversion between binary-coded decimal data and binary data is automatically executed in DSW, BIN instruction is not required.
4) Binary data is used in all operations in PLCs including arithmetic operations (+, -, × and ÷), increment and decrement instructions.

- When receiving the digital switch information in the binary-coded decimal (BCD) format into a PLC, use BIN instruction for converting BCD data into binary data.
- When outputting data to the seven-segment display unit handling binary-coded decimal (BCD) data, use BCD instruction for converting binary data into BCD data.

5) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

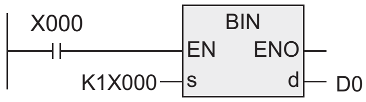
Error

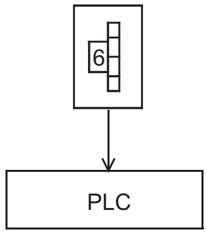
M8067 (operation error) turns ON when the source data is not binary-coded decimal (BCD).

Program examples

1. When the digital switch has 1 digit

[Structured ladder]



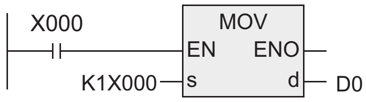


[ST]

BIN(X000, K1X000, D0);

MOV instruction can be used instead.

[Structured ladder]

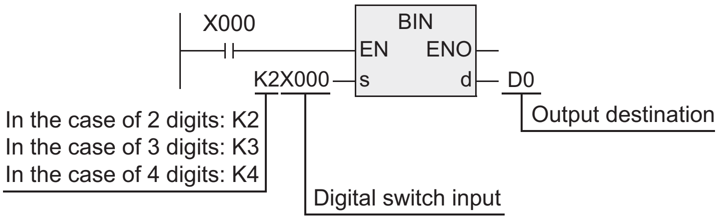


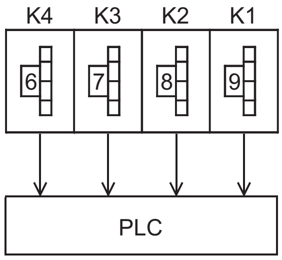
[ST]

MOV (X000, K1X000, D0);

2. When the digital switch has 2 to 4 digits

[Structured ladder]



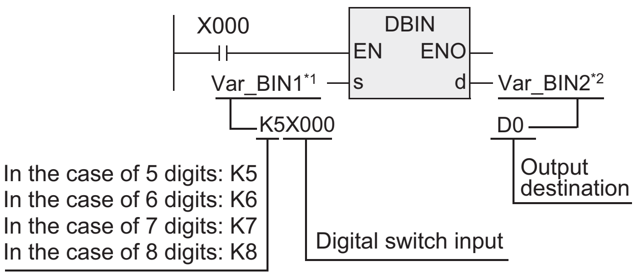


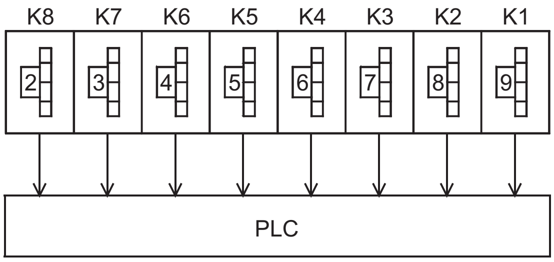
[ST]

BIN (X000, K2X000, D0);

3. When the digital switch has 5 to 8 digits

[Structured ladder]





[ST]

DBIN (X000, Var_BIN1*1 , Var_BIN2*2);

*1 Var_BIN1 is a global label and is defined as K5X000.
*2 Var_BIN2 is a global label and is defined as D0.

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

7.3 Arithmetic and Logical Operation

7.3.1 ADD

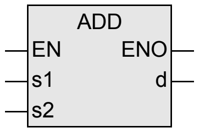
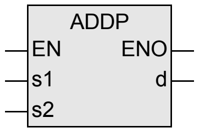
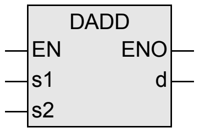
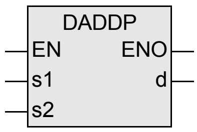
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction executes addition by two values to obtain the result (A + B = C)

→ **For the floating point addition instruction [DEADD], refer to Section 7.12.8.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
ADD	16 bits	Continuous		ADD(EN,s1,s2,d);
ADDP	16 bits	Pulse		ADDP(EN,s1,s2,d);
DADD	32 bits	Continuous		DADD(EN,s1,s2,d);
DADDP	32 bits	Pulse		DADDP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Data for addition or word device storing data	ANY16	ANY32
	s2	Data for addition or word device storing data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Word device storing the addition result	ANY16	ANY32

3. Applicable devices

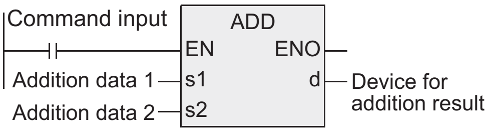
Operand type	Bit Devices: System User							Word Devices: Digit Specification				Word Devices: System User				Word Devices: Special Unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
s2								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
d									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(ADD, ADDP)

The data specified by (s1) and (s2) are added in the binary format, and the addition result is transferred to the device specified by (d).

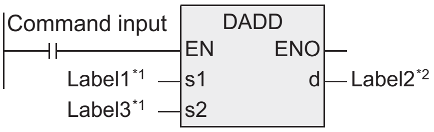


(s1) + (s2) → (d)

- The most significant bit of each data indicates the sign (positive: 0 or negative: 1), and data is added algebraically.
  5+(-8)=-3
- When a constant (K) is specified in (s1) or (s2), it is automatically converted into the binary format.

2. 32-bit operation(DADD, DADDP)

The data specified by (s1) and (s2) are added in the binary format, and the addition result is transferred to the device specified by (d).



[(s1)+1, (s1)] + [(s2)+1, (s2)] → [(d)+1, (d)]

*1 This defines the data to be added or the device that stores the data.
*2 This defines the device that stores the results of addition.

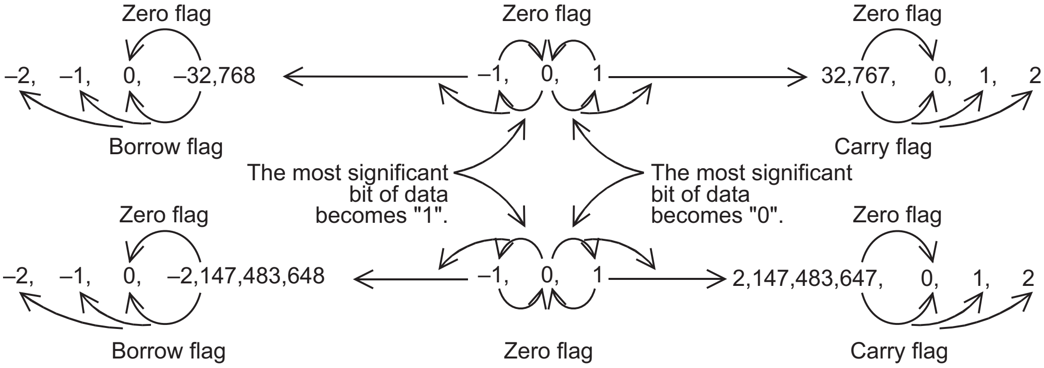
- The most significant bit of each data indicates the sign (positive: 0 or negative: 1), and data is added algebraically.
  5,500+(-8,540)=-3,040
- When a constant (K) is specified in (s1) or (s2), it is automatically converted into the binary format.

Related device

1. Relationship between the flag operation and the sign

→ For the flag operations, refer to Section 1.3.4.

Device	Name	Description
M8020	Zero	ON : When the operation result is 0 OFF : When the operation result is not 0
M8021	Borrow	ON : When the operation result is less than -32,768 (in 16-bit operation) or -2,147,483,648 (in 32-bit operation), the borrow flag operates. OFF : When the operation result is not less than -32,768 (in 16-bit operation) or -2,147,483,648 (in 32-bit operation)
M8022	Carry	ON : When the operation result is more than 32,767 (in 16-bit operation) or 2,147,483,647 (in 32-bit operation), the carry flag operates. OFF : When the operation result is not more than 32,767 (in 16-bit operation) or 2,147,483,647(in 32-bit operation)

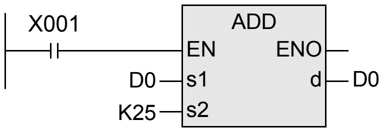


Extension function(FXU and FX2C PLCs)

The FXU PLC of V2.30 or earlier does not support the extension function.
When executing an instruction with M8023 ON, a binary float operation takes place.
In this case, K, H and D are valid as the object device for (s1) and (s2) and D is valid for (d).
The source data needs to be converted into binary float value in advance by FLT instruction.
Note, however, that constants K and H are automatically converted into binary float values.

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
3) When using a 32-bit operation instruction (DADD or DADDP) and specifying word devices, a 16-bit word device on the low-order side is specified first, and a word device with the subsequent device number is automatically set for the high-order 16 bits.
   To prevent number overlap, it is recommended to always specify an even number, for example.
4) The same device number can be specified for both the source and the destination.
   In this case, note that the addition result changes in every operation cycle if a continuous operation type instruction (ADD or DADD) is used.



(D 0) + 25 → (D 0)

5) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

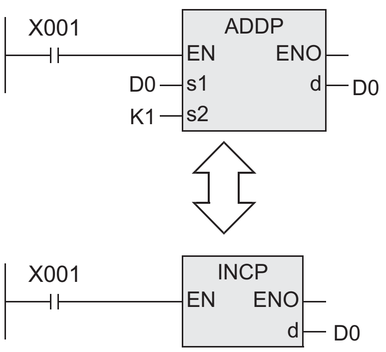
Program examples

1. Difference between ADD instruction and INC instruction caused by a program for adding "+1".

When ADD[P] instruction is executed, "1" is added to the contents of D0 every time X001 turns ON from OFF.
ADD[P] instruction is similar to INCP instruction described later except the contents shown in the table below.

			ADD/ADDP/DADD/DADDP	INC/INCP/DINC/DINCP
Flag (zero, borrow, carry)			Operates	Does not operate
Operation result	16-bit operation	(s)+(+1)=(d)	+32,767 → 0 → +1 → +2 →	+32,767 → -32,768 → -32,767
		(s)+(-1)=(d)	← -2 ← -1 ← 0 ← -32,768	-
	32-bit operation	(s)+(+1)=(d)	+2,147,483,647 → 0 → +1 → +2 →	+2,147,483,647 → -2,147,483,648 → -2,147,483,647
		(s)+(-1)=(d)	← -2 ← -1 ←0 ← -2,147,483,648	-

[Structured ladder]



(D 0) +1 →(D 0)

(D 0) +1 →(D 0)

[ST]

```
ADDP(X001, D0, K1, D0);
```

```
INCP(X001, D0);
```

7.3.2 SUB

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction executes subtraction using two values to obtain the result (A -B = C).

→ **For the floating point subtraction instruction [DESUB], refer to Section 7.12.9.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SUB	16 bits	Continuous	SUB EN ENO s1 d s2	SUB(EN,s1,s2,d);
SUBP	16 bits	Pulse	SUBP EN ENO s1 d s2	SUBP(EN,s1,s2,d);
DSUB	32 bits	Continuous	DSUB EN ENO s1 d s2	DSUB(EN,s1,s2,d);
DSUBP	32 bits	Pulse	DSUBP EN ENO s1 d s2	DSUBP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Data for subtraction or word device storing data	ANY16	ANY32
	s2	Data for subtraction or word device storing data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Word device storing the subtraction result	ANY16	ANY32

3. Applicable devices

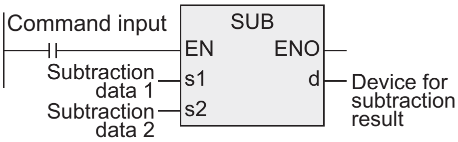
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(s2)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(SUB, SUBP)

The data specified by (s2) is subtracted from the data specified by (s1) in the binary format, and the subtraction result is transferred to (d).

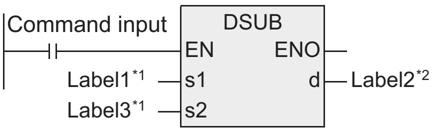


(s1) – (s2) → (d)

- The most significant bit of each piece of data indicates the sign (positive: 0 or negative: 1), and data is subtracted algebraically.
  (5-(-8)=13)
- When a constant (K) is specified in (s1) or (s2), it is automatically converted into the binary format.

2. 32-bit operation(DSUB, DSUBP)

The data specified by (s2) is subtracted from the data specified by (s1) in the binary format, and the subtraction result is transferred to (d).



[(s1)+1, (s1)] – [(s2)+1, (s2)] → [(d)+1, (d)]

*1 This defines the data or the device that stores the data.
*2 This defines the device that stores the subtraction result.

- The most significant bit of each piece of data indicates the sign (positive: 0 or negative: 1), and data is subtracted algebraically.
  (5,500-(-8,540)=14,040)
- When a constant (K) is specified in (s1) or (s2), it is automatically converted into the binary format.

Extension function(FXU and FX2C PLCs)

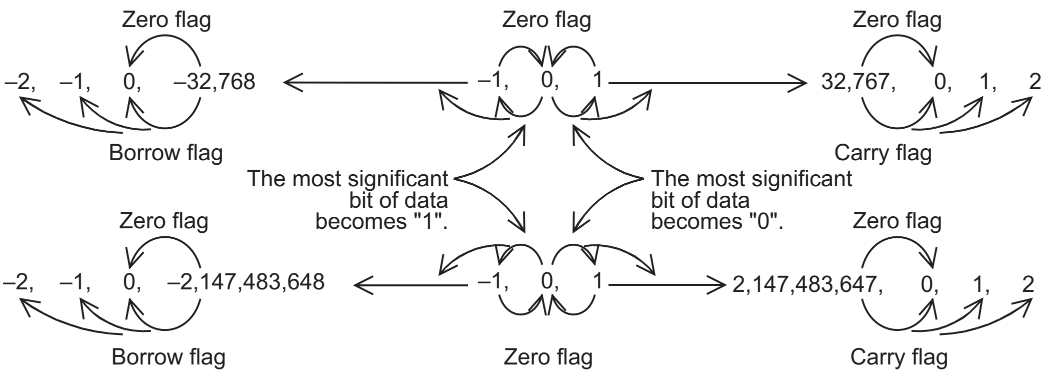
The FXU PLC of V2.30 or earlier does not support the extension function.
When executing an instruction with M8023 ON, a binary float operation takes place.
In this case, K, H and D are valid as the object device for (s1) and (s2) and D is valid for (d).
The source data needs to be converted into binary float value in advance by FLT instruction.
Note, however, that constants K and H are automatically converted into binary float values.

Related device

1. Relationship between the flag operation and the sign

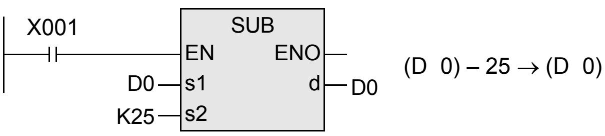
→ **For the flag operations, refer to Section 1.3.4.**

Device	Name	Description
M8020	Zero	ON : When the operation result is 0 OFF : When the operation result is not 0
M8021	Borrow	ON : When the operation result is less than -32,768 (in 16-bit operation) or -2,147,483,648 (in 32-bit operation), the borrow flag operates. OFF : When the operation result is not less than -32,768 (in 16-bit operation) or -2,147,483,648 (in 32-bit operation)
M8022	Carry	ON : When the operation result is more than 32,767 (in 16-bit operation) or 2,147,483,647 (in 32-bit operation), the carry flag operates. OFF : When the operation result is not more than 32,767 (in 16-bit operation) or 2,147,483,647(in 32-bit operation)



Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
To execute pulse operation, make the instruction execution condition pulse type.
3) When using a 32-bit operation instruction (DSUB or DSUBP) and specifying word devices, a 16-bit word device on the low-order side is specified first, and a word device with the subsequent device number is automatically set for the high-order 16 bits.
To prevent number overlap, it is recommended to always specify an even number, for example.
4) The same device number can be specified for both the source and the destination.
In this case, note that the addition result changes in every operation cycle if a continuous operation type instruction (SUB or DSUB) is used.



5) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

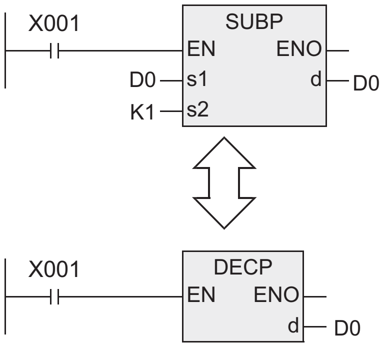
Program examples

1. Difference between SUB instruction and DEC instruction caused by a program for subtracting "1"

"1" is subtracted from the contents of D0 every time X001 turns ON from OFF.
SUB instruction is similar to DECP instruction described later except the contents shown in the table below.

			SUB/SUBP/DSUB/DSUBP instructions	DEC/DECP/DDEC/DDECP instructions
Flag (zero, borrow, carry)			Operates	Does not operate
Operation result	16-bit operation	(s)-(+1)=(d)	← -2 ← -1 ← 0 ← -32,768	-32,768 → +32,767 → +32,766
		(s)-(-1)=(d)	+32,767 → 0 → +1 → +2 →	-
	32-bit operation	(s)-(+1)=(d)	← -2 ← -1 ←0 ← -2,147,483,648	-2,147,483,648 → +2,147,483,647 → +2,147,483,646
		(s)-(-1)=(d)	+2,147,483,647 → 0 → +1 → +2 →	-

[Structured ladder]



(D 0) – 1 → (D 0)

(D 0) – 1 → (D 0)

[ST]

```
SUBP(X001, D0, K1, D0);
```

```
DECP(X001, D0);
```

7.3.3 MUL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction executes multiplication by two values to obtain the result (A × B = C).

→ **For the floating point multiplication instruction [DEMUL], refer to Section 7.12.10.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
MUL	16 bits	Continuous	MUL EN ENO s1 d s2	MUL(EN,s1,s2,d);
MULP	16 bits	Pulse	MULP EN ENO s1 d s2	MULP(EN,s1,s2,d);
DMUL	32 bits	Continuous	DMUL EN ENO s1 d s2	DMUL(EN,s1,s2,d);
DMULP	32 bits	Pulse	DMULP EN ENO s1 d s2	DMULP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s1)	Data for multiplication or word device storing the multiplication data	ANY16	ANY32
	(s2)	Data for multiplication or word device storing the multiplication data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Head word device storing the multiplication result	ANY32	ARRAY [1..2] OF ANY32

3. Applicable devices

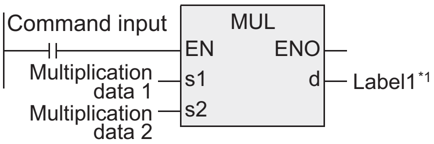
Operand type	Bit Devices							Word Devices												Others					
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲1	▲2		●	●	●	●			
(s2)								●	●	●	●	●	●	●	▲1	▲2		●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2		▲3	●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(MUL, MULP)

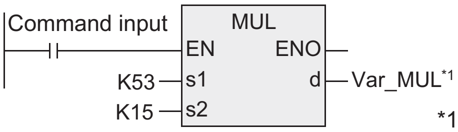
The data specified by (s1) is multiplied by data specified by (s2) in the binary format, and the multiplication result is transferred to 32-bit (double word) device specified by (d).



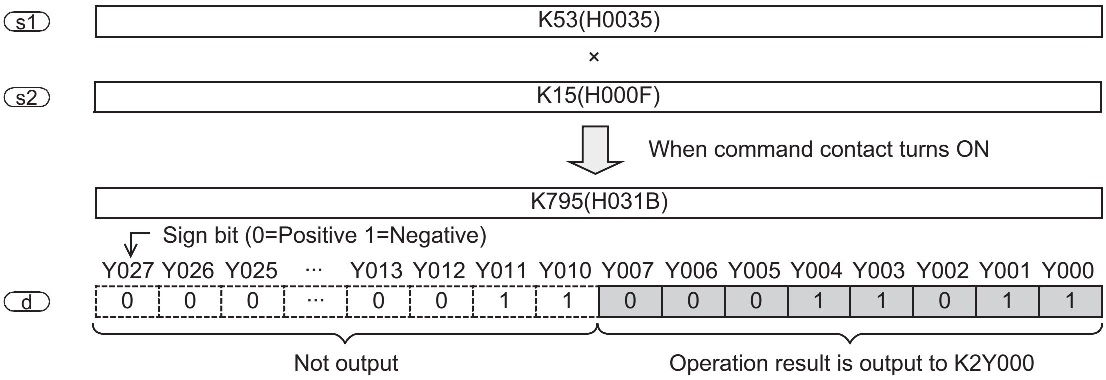
BIN BIN BIN
((s1)) × ((s2)) → ((d)+1, (d))
16 bits 16 bits 32 bits

*1 This defines the device that stores the multiplication result.

- The most significant bit of each piece of data indicates the sign (positive: 0 or negative: 1), and data is multiplied algebraically.
  (5×(-8)=-40)
- When a constant (K) is specified in (s1) or (s2), it is automatically converted into the binary format.
- When a digit (K1 to K8) is specified for the device specified by (d):
  A digit can be specified in the range from K1 to K8.
  For example, when K2 is specified, only low-order 8 bits can be obtained out of the product (32 bits).

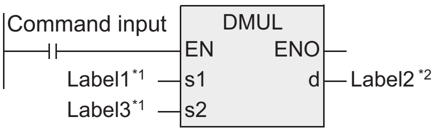


*1 Var_MUL is a global label and is defined as K2Y000.



2. 32-bit operation(DMUL, DMULP)

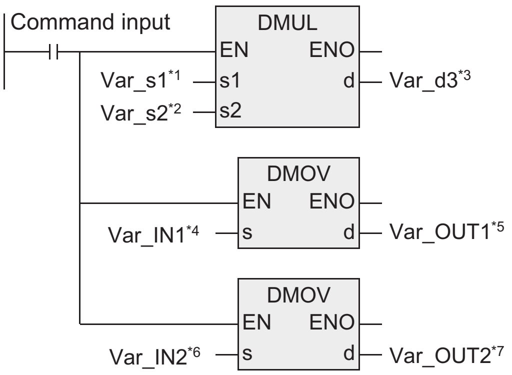
The data specified by (s1) is multiplied by the data specified by (s2) in the binary format, and the multiplication result is transferred to 64-bit (d) (four word devices).



*1 This defines the multiplication data or the device that stores the multiplication data.
*2 This defines the device that stores the multiplication result.

BIN [(s1)+1,(s1)] (32 bits) × BIN [(s2)+1,(s2)] (32 bits) → BIN [(d)+3,(d)+2,(d)+1,(d)] (64 bits)

- The most significant bit of each piece of data indicates the sign (positive: 0 or negative: 1), and data is multiplied algebraically.
  (5,500×(-8,540)=-46,970,000)
- When a constant (K) is specified in (s1) or (s2), it is automatically converted into the binary format.
- When a digit (K1 to K8) is specified for the device specified by (d), the result is obtained only for low-order 32 bits, and is not obtained for high-order 32 bits.
  Transfer the data to word devices once, then execute the operation.



(D51, D50) (D103, D102, D101, D100)
K100 × K150 → K15000

D100 is output to Y017 to Y000
D101 is output to Y037 to Y020

D102 is output to Y057 to Y040
D103 is output to Y077 to Y060

*1 Var_s1 is a global label and is defined as D50.
*2 Var_s2 is a global label and is defined as K150.
*3 Var_d3 is a global label and is defined as D100.
*4 Var_IN1 is a global label and is defined as D100.
*5 Var_OUT1 is a global label and is defined as K8Y000.
*6 Var_IN2 is a global label and is defined as D102.
*7 Var_OUT2 is a global label and is defined as K8Y040.

Extension function(FXU and FX2C PLCs)

The FXU PLC of V2.30 or earlier does not support the extension function.
When executing an instruction with M8023 ON, a binary float operation takes place, for example, (D1, D0) × (D3, D2) = (D5, D4).
In this case, K, H and D are valid as the object device for (s1) and (s2) and D is valid for (d).
The source data needs to be converted into binary float value in advance by FLT instruction.
Note, however, that constants K and H are automatically converted into binary float values.

Related device

1. Relationship between the flag operation and the sign

Device	Name	Description
M8304*1	Zero	ON : When the operation result is 0 OFF : When the operation result is not 0

*1. Available in the FX3U and FX3UC PLCs of Ver. 2.30 or later and FX3G PLC of Ver. 1.00 or later.

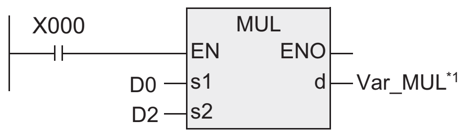
Cautions

1) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.
   ▲3:Available only for a 16-bit operation. Not available for a 32-bit operation.
2) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle array data or 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
3) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
4) When a 32-bit operation (DMUL or DMULP) is used, "Z" cannot be specified to the device specified by (d).
5) In monitoring the operation results by a programming tool, 64-bit data as the operation results cannot be monitored at one time even if a word device is used.
   In such a case, the FX3U, FX3UC and FX3G PLCs can use floating point operation.
   → **For the floating point operation, refer to Section 7.12.**

Program examples

1. 16-bit operation

[Structured ladder]



(D 0) × (D 2) × (D 5, D 4)
8 9 72

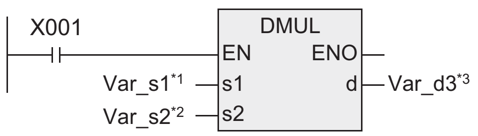
[ST]

MUL(X000, D0, D2, Var_MUL*1);

*1 Var_MUL is a global label and is defined as D4.

2. 32-bit operation

[Structured ladder]



(D 1,D 0) × (D 3,D 2) × (D 7,D 6,D 5,D 4)
1756 327 574,212

[ST]

DMUL(X001, Var_s1*1 , Var_s2*2 , Var_d3*3);

*1 Var_s1 is a global label and is defined as D0.
*2 Var_s2 is a global label and is defined as D2.
*3 Var_d3 is a global label and is defined as D4.

Function changes according to versions

Compatible versions			Item	Function summary
FX3U	FX3UC	FX3G		
Ver. 2.30 or later	Ver. 2.30 or later	Ver. 1.00 or later	Zero flag	Turns the special device M8304 ON when the operation result of MUL instruction is 0.

7.3.4 DIV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction executes division by two values to obtain the result [A ÷ B =C...(remainder)].

→ **For the floating point division instruction [DEDIV], refer to Section 7.12.11.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DIV	16 bits	Continuous	DIV EN ENO s1 d s2	DIV(EN,s1,s2,d);
DIVP	16 bits	Pulse	DIVP EN ENO s1 d s2	DIVP(EN,s1,s2,d);
DDIV	32 bits	Continuous	DDIV EN ENO s1 d s2	DDIV(EN,s1,s2,d);
DDIVP	32 bits	Pulse	DDIVP EN ENO s1 d s2	DDIVP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Data for division or word device storing the data (dividend).	ANY16	ANY32
	s2	Data for division or word device storing the data (divisor).	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head word device storing the division result (quotient and remainder)	ARRAY [0..1] OF ANY16	ARRAY [0..1] OF ANY32

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User			Special Unit	Index				Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲1	▲2		●	●	●	●			
s2								●	●	●	●	●	●	●	▲1	▲2		●	●	●	●			
d									●	●	●	●	●	●	▲1	▲2		▲3	●					

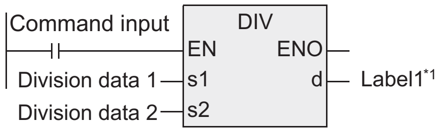
▲: Refer to "Cautions".

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

1. 16-bit operation(DIV, DIVP)

The contents specified by s1 indicates the dividend, the contents specified by s2 indicates the divisor, the quotient and remainder are transferred to the device specified by d.



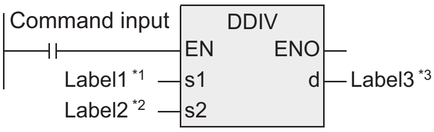
Dividend	Divisor	Quotient	Remainder
BIN	BIN	BIN	BIN
(s1) ÷	(s2) →	(d)···	(d+1)
16 bits	16 bits	16 bits	16 bits

*1 This defines the device that stores the division result.

- The most significant bit of each data indicates the sign (positive: 0 or negative: 1), and data is divided algebraically, for example, 36 ÷ (-5) = -7 (quotient) and 1 (remainder).
- Two devices in total starting from d are occupied to store the operation result (quotient and remainder). Make sure that these two devices are not used for other control.
- When a constant (K) is specified in s1 or s2, it is automatically converted into the binary format.

2. 32-bit operation(DDIV, DDIVP)

The contents specified by s1 indicates the dividend, the contents specified by s2 indicates the divisor, the quotient and remainder are transferred to the device specified by d.



*1 This defines the dividend data or the device that stores the dividend data.
*2 This defines the divisor data or the device that stores the divisor data.
*3 This defines the device that stores the division result.

Dividend	Divisor	Quotient	Remainder
BIN	BIN	BIN	BIN
[s1+1, s1] ÷	[s2+1, s2] →	[d+1, d] ···	[d+3, d+2]
32 bits	32 bits	32 bits	32 bits

- Four devices in total starting from d are occupied to store the operation result (quotient and remainder). Make sure that these four devices are not used for other control.
- The most significant bit of each data indicates the sign (positive: 0 or negative: 1), and data is divided algebraically, for example, 5,500 ÷ (-540) = -10 (quotient) and 100 (remainder).
- When a constant (K) is specified in s1 or s2, it is automatically converted into the binary format.

Extension function(FXU and FX2C PLCs)

The FXU PLC of V2.30 or earlier does not support the extension function.
When executing an instruction with M8023 ON, a binary float operation takes place, for example, (D1, D0) ÷ (D3, D2) = (D5, D4).
In this case, K, H and D are valid as the object device for s1 and s2 and D is valid for d.
The source data needs to be converted into binary float value in advance by FLT instruction.
Note, however, that constants K and H are automatically converted into binary float values.

Related device

Device	Name	Description
M8304*1	Zero	ON : When the operation result is 0 OFF : When the operation result is not 0
M8306*1	Carry	ON : When the operation result is more than 32,767 (in 16-bit operation) or 2,147,483,647 (in 32-bit operation), the carry flag operates. OFF : When the operation result is not more than 32,767 (in 16-bit operation) or 2,147,483,647(in 32-bit operation)

*1. Available in the FX3U and FX3UC PLCs of Ver. 2.30 or later and FX3G PLC of Ver. 1.00 or later.

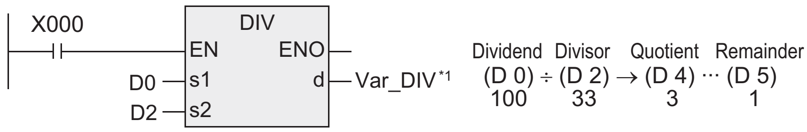
Cautions

1) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.
   ▲3:Available only for a 16-bit operation. Not available for a 32-bit operation.
2) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle array data or 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
3) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
4) The most significant bit of the quotient and remainder indicates the sign (positive: 0 or negative: 1).
5) The quotient is negative when either the dividend or divisor is negative.
   The remainder is negative when the dividend is negative.
6) The remainder is not obtained when a bit device is specified with digit specification for the device specified by (d).
7) In a 32-bit operation (by DDIV or DDIVP), Z cannot be specified as the device specified by (d).

Program examples

1. 16-bit operation

[Structured ladder]



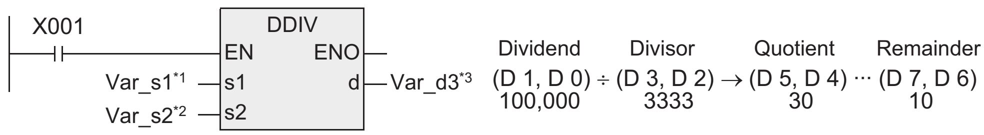
[ST]

```
MUL(X000,D0,D2,Var_DIV*1);
```

*1 Var_DIV is a global label and is defined as D4.

2. 32-bit operation

[Structured ladder]



[ST]

```
DDIV(X001, Var_s1*1 , Var_s2*2 , Var_d3*3 );
```

*1 Var_s1 is a global label and is defined as D0.
*2 Var_s2 is a global label and is defined as D2.
*3 Var_d3 is a global label and is defined as D4.

Function changes according to versions

Compatible versions			Item	Function summary
FX3U	FX3UC	FX3G		
Ver. 2.30 or later	Ver. 2.30 or later	Ver. 1.00 or later	Zero flag	Turns M8304 ON when the operation result of DIV instruction is 0.
			Carry flag	Turns M8306 ON when the operation result of DIV instruction overflows. 16-bit operation : Only when the maximum negative value (-32,768) is divided by "-1". 32-bit operation : Only when the maximum negative value (-2,147,483,648) is divided by "-1".

7.3.5 INC

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction increments the data of a specified device by "1" (+1 addition).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
INC	16 bits	Continuous	INC EN ENO d	INC(EN,d);
INCP	16 bits	Pulse	INCP EN ENO d	INCP(EN,d);
DINC	32 bits	Continuous	DINC EN ENO d	DINC(EN,d);
DINCP	32 bits	Pulse	DINCP EN ENO d	DINCP(EN,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
Output variable	ENO	Execution state	Bit	
	(d)	Word device storing data to be incremented by "1"	ANY16	ANY32

3. Applicable devices

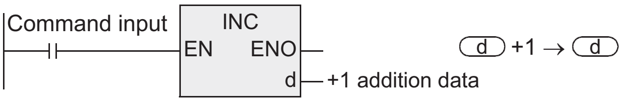
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation

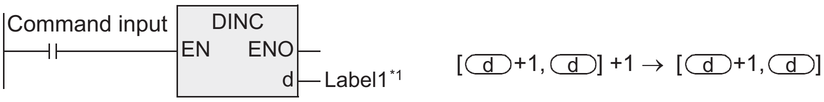
1. 16-bit operation(INC, INCP)

The contents of the device specified by (d) is incremented by "1", and the increment result is transferred to (d).



2. 32-bit operation(DINC, DINCP)

The contents of the device specified by (d) is incremented by "1", and the increment result is transferred to (d).



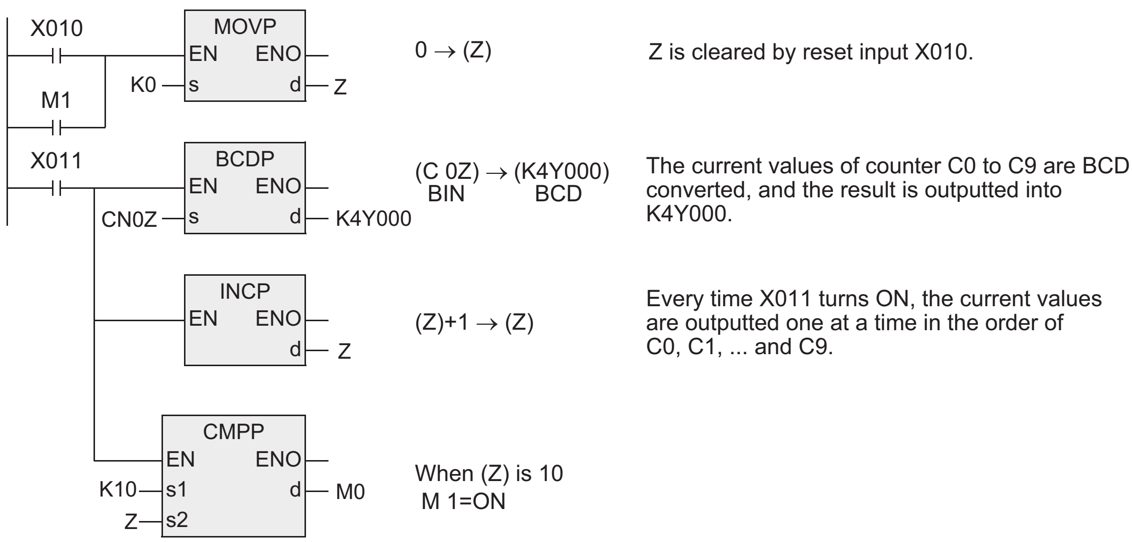
*1 This defines the device that stores the data to be added by "1".

Cautions

1) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle array data or 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
3) Note that data is incremented in every operation cycle in a continuous operation type instruction.
4) In a 16-bit operation, when "+32,767" is incremented by "1", the result is "-32,768". Flags (zero, borrow and carry) are not activated at this time.
5) In a 32-bit operation, when "+2,147,483,647" is incremented by "1", the result is "-2,147,483,648". Flags (zero, borrow and carry) are not activated at this time.
6) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

[Structured ladder]



[ST]

```
MOVP(X010 OR M1, K0, Z);
BCDP(X011, CN0Z, K4Y000);
INCP(X011, Z);
CMPP(X011, K10, Z, M0);
```

7.3.6 DEC

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction decrements the data of a specified device by "1" (-1 addition).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DEC	16 bits	Continuous	DEC EN ENO d	DEC(EN,d);
DECP	16 bits	Pulse	DECP EN ENO d	DECP(EN,d);
DDEC	32 bits	Continuous	DDEC EN ENO d	DDEC(EN,d);
DDECP	32 bits	Pulse	DDECP EN ENO d	DDECP(EN,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
Output variable	ENO	Execution state	Bit	
	(d)	Device storing data to be decremented by "1"	ANY16	ANY32

3. Applicable devices

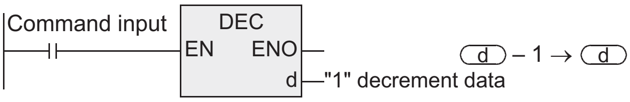
Operand type	Bit Devices							Word Devices													Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation

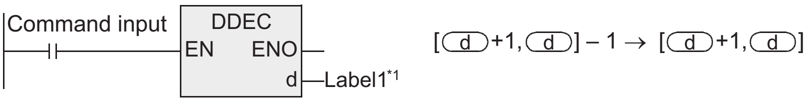
1. 16-bit operation(DEC, DECP)

The contents of the device specified by (d) are decremented by "1", and the decremented result is transferred to the device specified by (d).



2. 32-bit operation(DDEC, DDECP)

The contents of the device specified by (d) are decremented by "1", and the decremented result is transferred to the device specified by (d).



*1 This defines the device that stores the data to be decremented by "1".

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
3) In a 16-bit operation, when "+32,767" is incremented by "1", the result is "-32,768". Flags (zero, borrow and carry) are not activated at this time.
4) In a 32-bit operation, when "+2,147,483,647" is incremented by "1", the result is "-2,147,483,648". Flags (zero, borrow and carry) are not activated at this time.
5) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

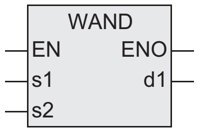
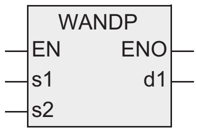
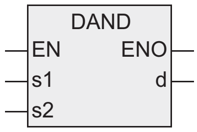
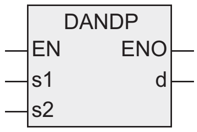
7.3.7 WAND

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction executes the logical product (AND) operation of two numeric values.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
WAND	16 bits	Continuous		WAND(EN,s1,s2,d1);
WANDP	16 bits	Pulse		WANDP(EN,s1,s2,d1);
DAND	32 bits	Continuous		DAND(EN,s1,s2,d1);
DANDP	32 bits	Pulse		DANDP(EN,s1,s2,d1);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Data used for logical product or word device storing data	ANY16	ANY32
	s2	Data used for logical product or word device storing data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Word device storing the logical product result	ANY16	ANY32

3. Applicable devices

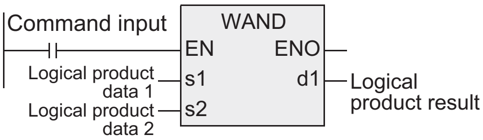
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User			Special Unit		Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
s2								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
d									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(WAND, WANDP)

The logical product (AND) operation is executed to the contents specified by (s1) and (s2) in units of bit, and the result is transferred to the device specified by (d).



(s1) ∧ (s2) → (d)

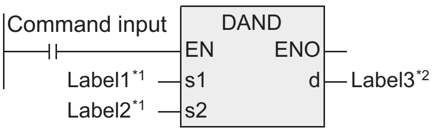
- While the command input is OFF, the data of the transfer destination specified by (d) does not change.
- While the command input is ON, the data of the transfer sources specified by (s1) and (s2) do not change.
- When a constant (K) is specified in the transfer sources specified by (s1) and (s2), it is automatically converted into the binary format.
- The logical product operation is executed in units of bit as shown in the table below (1 ∧ 1 = 1, 0 ∧ 1 = 0, 1 ∧ 0 = 0 and 0 ∧ 0 = 0).

In the table : 1=ON, 0=OFF

	(s1)	(s2)	(d) WAND
Logical operation (unit: bit)	0	0	0
	1	0	0
	0	1	0
	1	1	1

2. 32-bit operation(DAND, DANDP)

The logical product (AND) operation is executed to the contents specified by (s1) and (s2) in units of bit, and the result is transferred to the device specified by (d).



(s1)+1, (s1) ∧ (s2)+1, (s2) → (d)+1, (d)

*1 This defines the logical product data or the device that stores the logical product data.
*2 This defines the device that stores the logical product operation result.

- While the command input is OFF, the data of the transfer destination specified by (d) does not change.
- While the command input is ON, the data of the transfer sources specified by (s1) and (s2) do not change.
- When a constant (K) is specified in the transfer sources specified by (s1) and (s2), it is automatically converted into the binary format.
- The logical product operation is executed in units of bit as shown in the table below (1∧1 = 1, 0∧1 = 0, 1∧0 = 0 and 0∧0 = 0).

In the table : 1=ON, 0=OFF

	(s1)+1, (s1)	(s2)+1, (s2)	(d)+1, (d) DAND instruction
Logical operation (unit: bit)	0	0	0
	1	0	0
	0	1	0
	1	1	1

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
To execute pulse operation, make the instruction execution condition pulse type.
3) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

7.3.8 WOR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction executes the logical sum (OR) operation of two numeric values.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
WOR	16 bits	Continuous	WOR EN ENO s1 d1 s2	WOR(EN,s1,s2,d1);
WORP	16 bits	Pulse	WORP EN ENO s1 d1 s2	WORP(EN,s1,s2,d1);
DOR	32 bits	Continuous	DOR EN ENO s1 d s2	DOR(EN,s1,s2,d1);
DORP	32 bits	Pulse	DORP EN ENO s1 d s2	DORP(EN,s1,s2,d1);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Data used for logical sum or word device storing data	ANY16	ANY32
	s2	Data used for logical sum or word device storing data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Word device storing the logical sum result.	ANY16	ANY32

3. Applicable devices

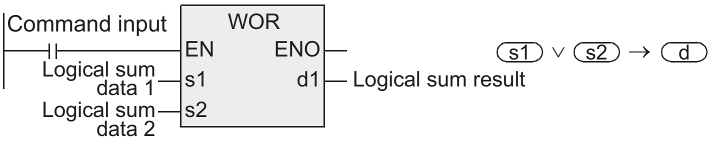
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
s2								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
d									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(WOR, WORP)

The logical sum (OR) operation is executed to the contents specified by (s1) and (s2) in units of bit, and the result is transferred to the device specified by (d).



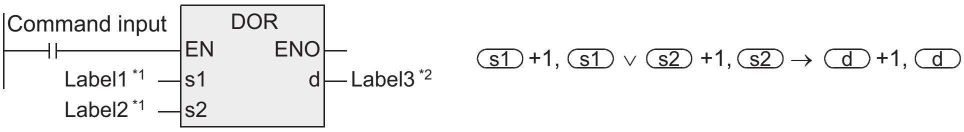
- While the command input is OFF, the data of the transfer destination specified by (d) does not change.
- While the command input is ON, the data of the transfer sources specified by (s1) and (s2) do not change.
- When a constant (K) is specified in the transfer sources specified by (s1) and (s2), it is automatically converted into the binary format.
- The logical sum operation is executed in units of bit as shown in the table below (1 ∨ 1 = 1, 0 ∨ 1 = 1, 0 ∨ 0 = 0 and 1 ∨ 0 = 1).

In the table : 1=ON, 0=OFF

	(s1)	(s2)	(d) WOR
Logical operation (unit: bit)	0	0	0
	1	0	1
	0	1	1
	1	1	1

2. 32-bit operation(DOR, DORP)

The logical sum (OR) operation is executed to the contents specified by (s1) and (s2) in units of bit, and the result is transferred to the device specified by (d).



*1 This defines the logical sum data or the device that stores the logical sum data.
*2 This defines the device that stores the logical sum result.

- While the command input is OFF, the data of the transfer destination specified by (d) does not change.
- While the command input is ON, the data of the transfer sources specified by (s1) and (s2) do not change.
- When a constant (K) is specified in the transfer sources specified by (s1) and (s2), it is automatically converted into the binary format.
- The logical sum operation is executed in units of bit as shown in the table below (1 ∨ 1 = 1, 0 ∨ 1 = 1, 0 ∨ 0 = 0 and 1 ∨ 0 = 1).

In the table : 1=ON, 0=OFF

	(s1)+1, (s1)	(s2)+1, (s2)	(d)+1, (d) DOR instruction
Logical operation (unit: bit)	0	0	0
	1	0	1
	0	1	1
	1	1	1

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
To execute pulse operation, make the instruction execution condition pulse type.
3) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

7.3.9 WXOR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction executes the exclusive logical sum (XOR) operation of two numeric values.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
WXOR	16 bits	Continuous	WXOR EN ENO s1 d1 s2	WXOR(EN,s1,s2,d1);
WXORP	16 bits	Pulse	WXORP EN ENO s1 d1 s2	WXORP(EN,s1,s2,d1);
DXOR	32 bits	Continuous	DXOR EN ENO s1 d s2	DXOR(EN,s1,s2,d1);
DXORP	32 bits	Pulse	DXORP EN ENO s1 d s2	DXORP(EN,s1,s2,d1);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s1)	Data used for exclusive logical sum or word device storing data	ANY16	ANY32
	(s2)	Data used for exclusive logical sum or word device storing data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Word device storing the exclusive logical sum result	ANY16	ANY32

3. Applicable devices

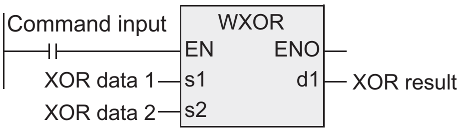
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(s2)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(WXOR, WXORP)

The exclusive logical sum (XOR) operation is executed to the contents specified by (s1) and (s2) in units of bit, and the result is transferred to the device specified by (d).



(s1) ∀ (s2) → (d)

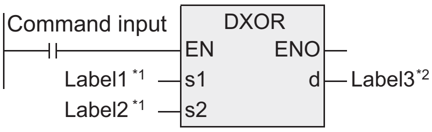
- While the command input is OFF, the data of the transfer destination specified by (d) does not change.
- While the command input is ON, the data of the transfer sources specified by (s1) and (s2) do not change.
- When a constant (K) is specified in the transfer sources specified by (s1) and (s2), it is automatically converted into the binary format.
- The logical exclusive sum operation is executed in units of bit as shown in the table below (1 ∀ 1 = 0, 0 ∀ 0 = 0, 1 ∀ 0 = 1 and 0 ∀ 1 = 1).

In the table : 1=ON, 0=OFF

	(s1)	(s2)	(d) WXOR
Logical operation (unit: bit)	0	0	0
	1	0	1
	0	1	1
	1	1	0

2. 32-bit operation(DXOR, DXORP)

The exclusive logical sum (XOR) operation is executed to the contents specified by (s1) and (s2) in units of bit, and the result is transferred to the device specified by (d).



(s1) +1, (s1) ∀ (s2) +1, (s2) → (d) +1, (d)

*1 This defines the exclusive logical sum data or the device that stores the exclusive logical sum data.
*2 This defines the devices that stores the exclusive logical sum result.

- While the command input is OFF, the data of the transfer destination specified by (d) does not change.
- While the command input is ON, the data of the transfer sources specified by (s1) and (s2) do not change.
- When a constant (K) is specified in the transfer sources specified by (s1) and (s2), it is automatically converted into the binary format.
- The logical exclusive sum operation is executed in units of bit as shown in the table below (1 ∀ 1 = 0, 0 ∀ 0 = 0, 1 ∀ 0 = 1 and 0 ∀ 1 = 1).

In the table : 1=ON, 0=OFF

	(s1) +1, (s1)	(s2) +1, (s2)	(d) +1, (d) DXOR instruction
Logical operation (unit: bit)	0	0	0
	1	0	1
	0	1	1
	1	1	0

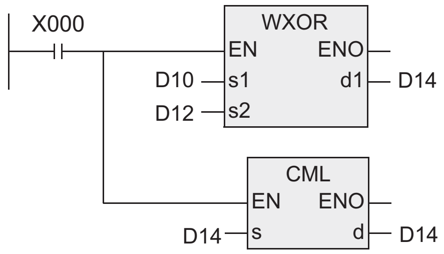
Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
To execute pulse operation, make the instruction execution condition pulse type.
3) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

By combining WXOR and CML instructions, the exclusive logical sum not (XORNOT) operation can be executed.

[Structured ladder]



[ST]

```
WXOR(X000, D10, D12, D14);
CML(X000, D14, D14);
```

7.3.10 NEG

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This instruction obtains the 2's complement of a numeric value (by inverting each bit and adding "1").
A sign of a numeric value can be converted by this instruction.

→ **For the floating point sign inversion instruction [DENEG], refer to Section 7.12.16.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	ST
NEG	16 bits	Continuous	NEG [EN → ENO, d]	NEG(EN,d);
NEGP	16 bits	Pulse	NEGP [EN → ENO, d]	NEGP(EN,d);
DNEG	32 bits	Continuous	DNEG [EN → ENO, d]	DNEG(EN,d);
DNEGP	32 bits	Pulse	DNEGP [EN → ENO, d]	DNEGP(EN,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
Output variable	ENO	Execution state	Bit	
	(d)	Word device which stores data for obtaining complement and will store the operation result. (The operation result will be stored in the same word device.)	ANY16	ANY32

3. Applicable devices

Operand type	Bit Devices: System User							Word Devices: Digit Specification				System User				Special Unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)									●	●	●	●	●	●	▲1	▲1	●	●	●					

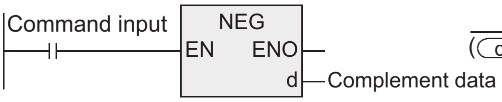
▲: Refer to "Cautions".

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

1. 16-bit operation(NEG, NEGP)

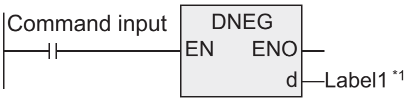
Each bit of the device specified by (d) is inverted (0 → 1, 1 → 0), "1" is added, and then the result is stored in the original device.



$$\overline{(d)} + 1 \rightarrow d$$

2. 32-bit operation(DNEG, DNEGP)

Each bit of the device specified by (d) is inverted (0 → 1, 1 → 0), "1" is added, and then the result is stored in the original device.



$$\overline{(d+1,\ d)}+1 = (d+1,\ d)$$

*1 This defines the device that stores the complement data.

Cautions

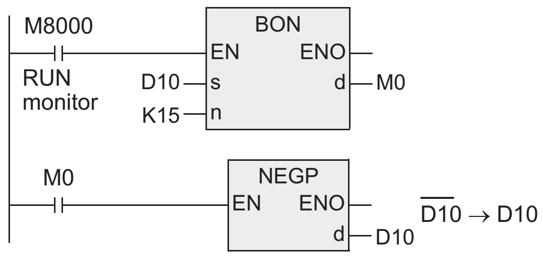
1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) Note that the complement is obtained in every scan time (operation cycle) in a continuous operation type instruction (NEG,DNEG).
3) Some restrictions to applicable devices
   ▲1:The FX3U and FX3UC PLCs only are applicable.

Program examples

The program examples below are provided to obtain the absolute value of a negative binary value.

1. Obtaining the absolute value of a negative value using NEG instruction

[Structured ladder]



In BON (ON bit check) instruction, M0 turns ON when the bit 15 (b15 among b0 to b15) of D10 is "1".

NEGP instruction is executed for D10 only when M0 turns ON.

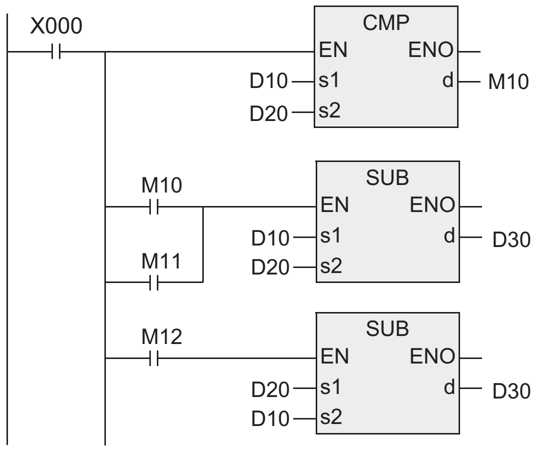
[ST]

```
BON(M8000, D0, K15, M0);
NEGP(M0, D10);
```

2. Obtaining the absolute value by SUB (subtraction) instruction

Even if NEG instruction (complement operation) is not used, D30 always stores the absolute value of the difference.

[Structured ladder]



(D 10) > (D 20) M 10=ON (D10) = (D 20) M 11=ON (D 10) < (D 20) M 12=ON

In the case of "D10 ≧ D20", D10 - D20 → D30

In the case of "D10 < D20", D20 - D10 → D30

[ST]

```
CMP(D10, D20, M10);
SUB(X000 AND (M10 OR M11), D10, D20, D30);
SUB(X000 AND M12, D20, D10, D30);
```

Negative value expression and absolute value (reference)

In PLCs, a negative value is expressed in 2's complement.
When the most significant bit is "1", it is a negative value, and its absolute value can be obtained by NEG instruction.

(D 10) = 2

0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

(D 10) = 1

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

(D 10) = 0

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(D 10) = –1

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

→ ($\overline{\text{D 10}}$) +1 = 1

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

(D 10) = –2

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

→ ($\overline{\text{D 10}}$) +1 = 2

0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

⋮

(D 10) = –32,767

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

→ ($\overline{\text{D 10}}$) +1 = 32,767

0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

(D 10) = –32,767

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

→ ($\overline{\text{D 10}}$)+1 = –32,768

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The absolute value can be obtained up to 32,767.

7.4 Rotation and Shift Operation

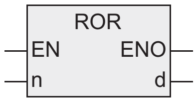
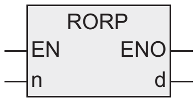
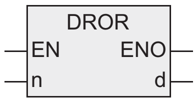
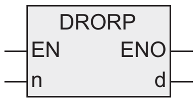
7.4.1 ROR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction shifts and rotates the bit information rightward by the specified number of bits without the carry flag.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	ST
ROR	16 bits	Continuous		ROR(EN,n,d);
RORP	16 bits	Pulse		RORP(EN,n,d);
DROR	32 bits	Continuous		DROR(EN,n,d);
DRORP	32 bits	Pulse		DRORP(EN,n,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	(n)	Number of bits to be rotated n≤16(16-bit operation), n≤32(32-bit operation)	ANY16	
Output variable	ENO	Execution state	Bit	
	(d)	Word device storing data to be rotated rightward	ANY16	ANY32

3. Applicable devices

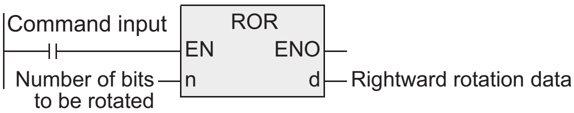
Operand type	Bit Devices: System User							Word Devices: Digit Specification				System User			Special Unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)									▲1	▲1	▲1	●	●	●	▲2	▲3	●	●	●					
(n)														●	▲2					●	●			

▲: Refer to "Cautions".

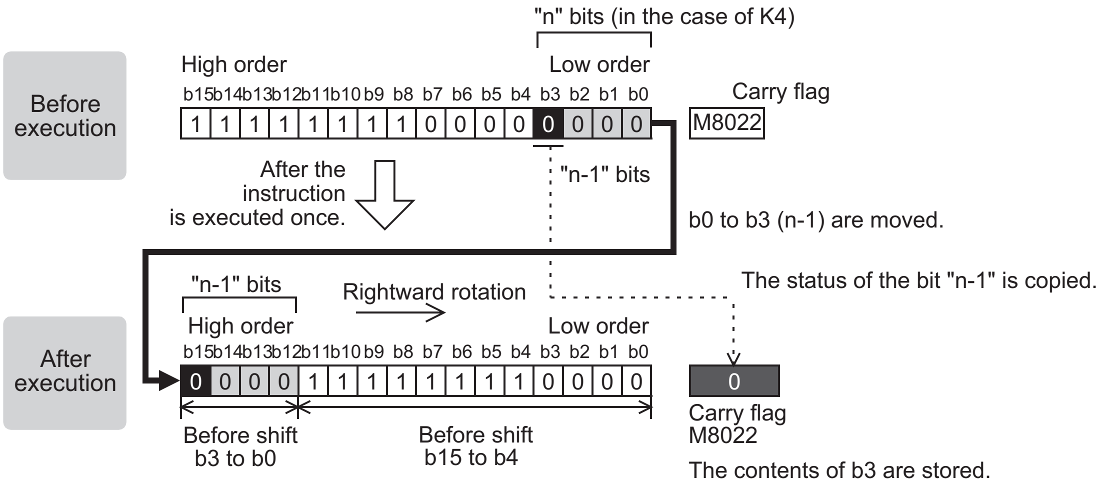
Function and operation explanation

1. 16-bit operation (ROR, RORP)

"n" bits out of 16 bits of the device specified by (d) are rotated rightward.

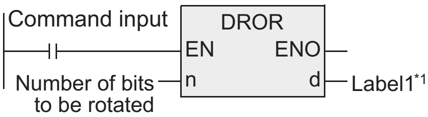


- The final bit is stored in the carry flag (M8022).
- In a device with digit specification, K4 (16-bit instruction) is valid.



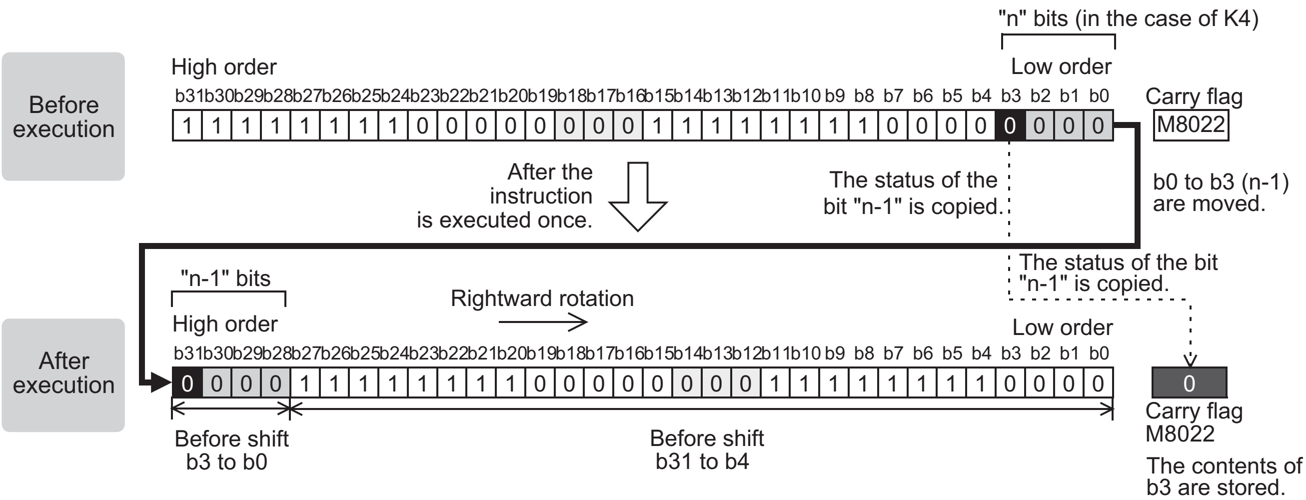
2. 32-bit operation (DROR, DRORP)

"n" bits out of 32 bits of the device specified by (d) are rotated rightward.



*1 This defines the device that stores the data to be rotated rightward.

- The final bit is stored in the carry flag (M8022).
- In a device with digit specification, K8 (32-bit instruction) is valid.



Related device

→ **For the carry flag use method, refer to Section 1.3.4.**

Device	Name	Description
M8022	Carry	Turns ON when the bit shifted last from the lowest position is "1".

Cautions

1) Some restrictions to applicable devices
▲1:K4Y○○○, K4M○○○ and K4S○○○ are valid for a 16-bit operation.
K8Y○○○, K8M○○○ and K8S○○○ are valid for a 32-bit operation.
▲2:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲3:The FX3U and FX3UC PLCs only are applicable.

2) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

3) In the case of continuous operation type instructions (ROR and DROR), note that shift and rotation are executed in every scan time (operation cycle).

4) When a device with digit specification is specified as (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid (examples: K4Y010 or K8M0).

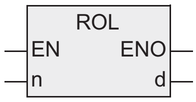
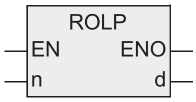
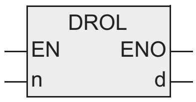
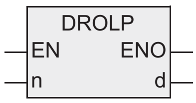
7.4.2 ROL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction shifts and rotates the bit information leftward by the specified number of bits without the carry flag.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ROL	16 bits	Continuous		ROL(EN,n,d);
ROLP	16 bits	Pulse		ROLP(EN,n,d);
DROL	32 bits	Continuous		DROL(EN,n,d);
DROLP	32 bits	Pulse		DROLP(EN,n,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(n)	Number of bits to be rotated n≤16(16-bit operation), n≤32(32-bit operation)	ANY16	
Output variable	ENO	Execution state	Bit	
	(d)	Word device storing data to be rotated leftward	ANY16	ANY32

3. Applicable devices

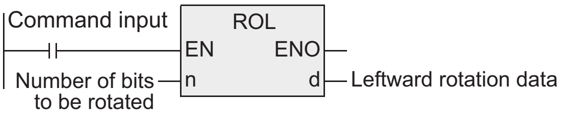
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)									▲1	▲1	▲1	●	●	●	▲2	▲3	●	●	●					
(n)														●	▲2					●	●			

▲: Refer to "Cautions".

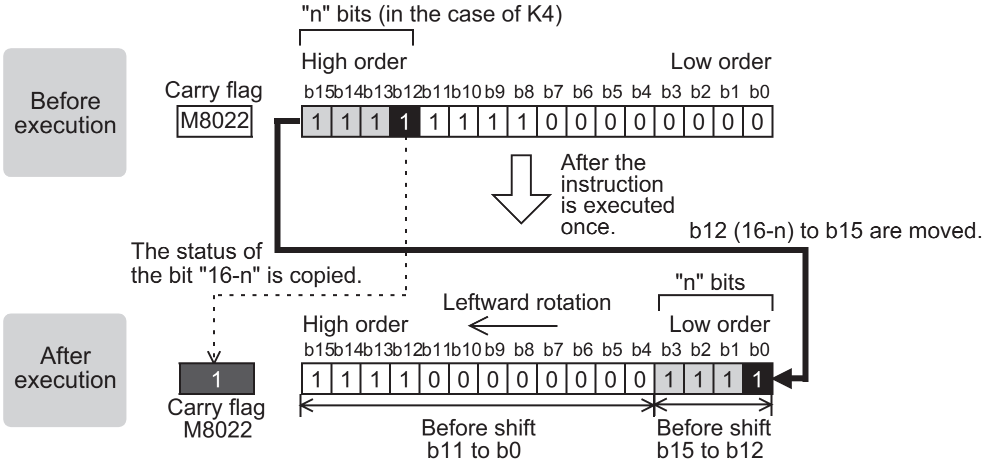
Function and operation explanation

1. 16-bit operation (ROL, ROLP)

"n" bits out of 16 bits of the device specified by (d) are rotated leftward.



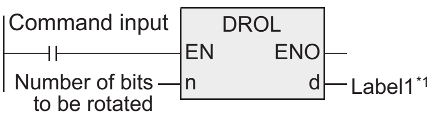
- The final bit is stored in the carry flag (M8022).
- In a device with digit specification, K4 (16-bit instruction) is valid.



The contents of b12 are stored.

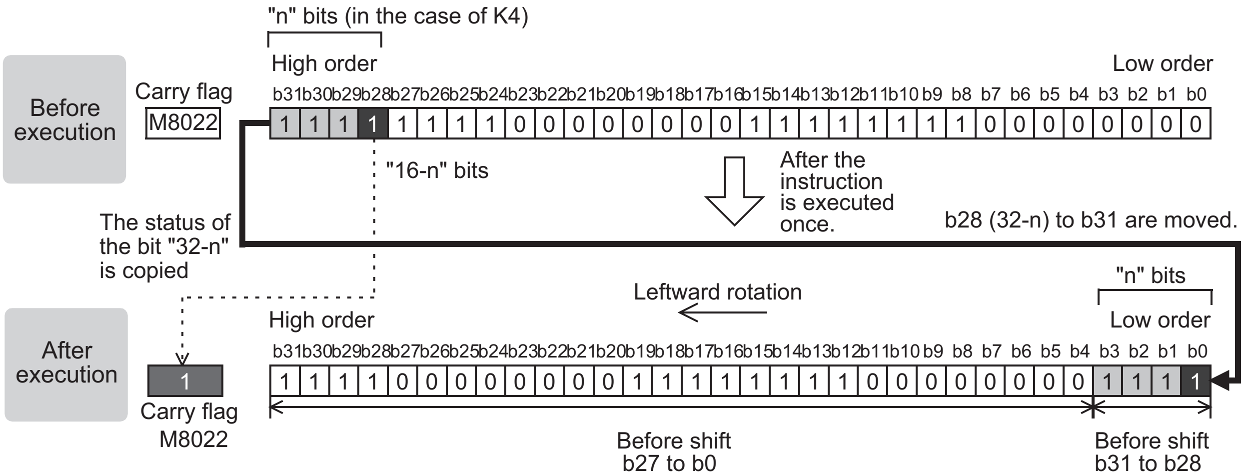
2. 32-bit operation (DROL, DROLP)

"n" bits out of 32 bits of the device specified by (d) are rotated leftward.



*1 This defines the device that stores the data to be rotated leftward.

- The final bit is stored in the carry flag (M8022).
- In a device with digit specification, K8 (32-bit instruction) is valid.



The contents of b28 are stored.

Related device

→ For the carry flag use method, refer to Section 1.3.4.

Device	Name	Description
M8022	Carry	Turns ON when the bit shifted last from the highest position is "1".

Cautions

1) Some restrictions to applicable devices
▲1:K4Y○○○, K4M○○○ and K4S○○○ are valid for a 16-bit operation.
K8Y○○○, K8M○○○ and K8S○○○ are valid for a 32-bit operation.
▲2:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲3:The FX3U and FX3UC PLCs only are applicable.
2) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
3) In the case of continuous operation type instructions (ROL and DROL), note that shift and rotation are executed in every scan time (operation cycle).
4) When a device with digit specification is specified as (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid (examples: K4Y010 or K8M0).

7.4.3 RCR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This instruction shifts and rotates the bit information rightward by the specified number of bits together with the carry flag.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
RCR	16 bits	Continuous	RCR (EN, ENO, n, d)	RCR(EN,n,d);
RCRP	16 bits	Pulse	RCRP (EN, ENO, n, d)	RCRP(EN,n,d);
DRCR	32 bits	Continuous	DRCR (EN, ENO, n, d)	DRCR(EN,n,d);
DRCRP	32 bits	Pulse	DRCRP (EN, ENO, n, d)	DRCRP(EN,n,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	(n)	Number of bits to be rotated n≤16(16-bit operation), n≤32(32-bit operation)	ANY16	
Output variable	ENO	Execution state	Bit	
	(d)	Word device storing data to be rotated rightward	ANY16	ANY32

3. Applicable devices

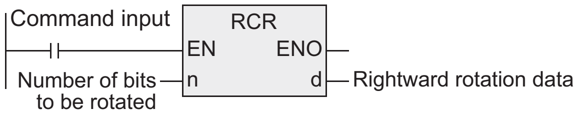
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)									▲1	▲1	▲1	●	●	●	▲2	▲2	●	●	●					
(n)														●	▲2					●	●			

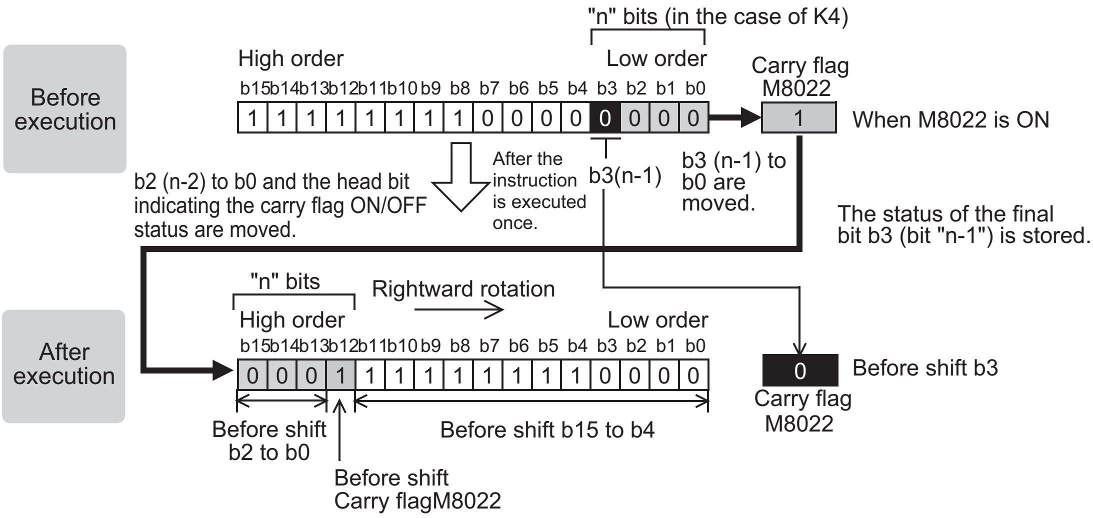
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (RCR, RCRP)

"n" bits out of 16 bits of the device specified by (d) and 1 bit (carry flag M8022) are rotated rightward.

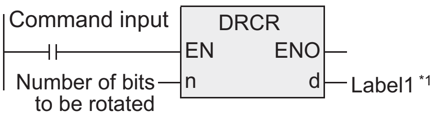




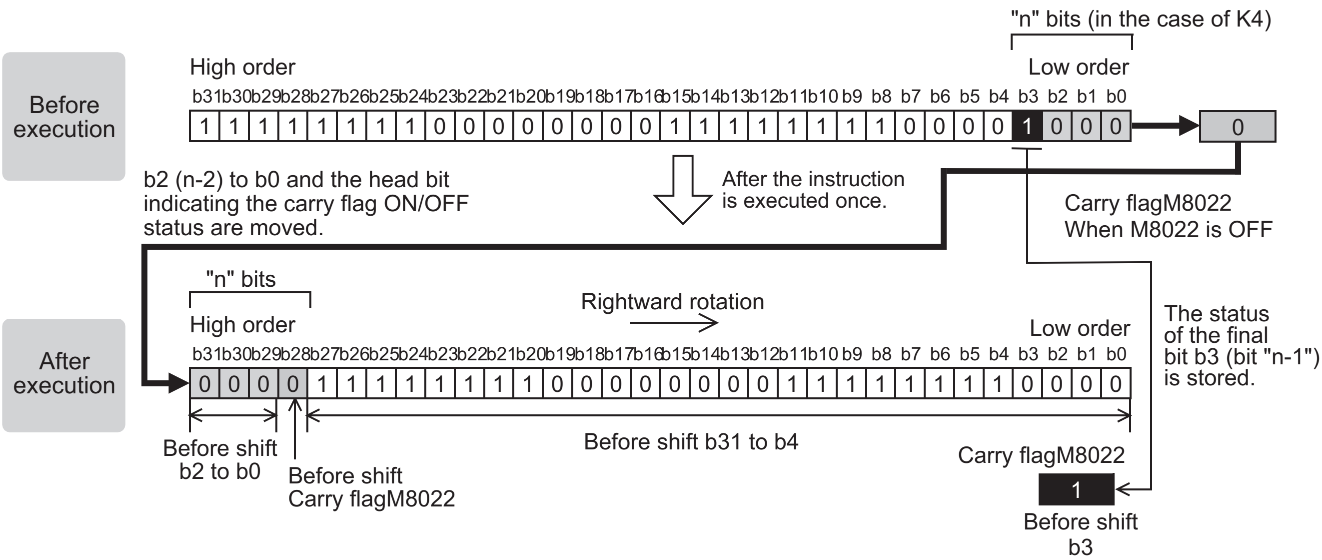
The carry flag is intervened in the rotation loop. If M8022 has been set to ON or OFF before the rotation instruction, the carry flag is transferred to the destination.

2. 32-bit operation (DRCR, DRCRP)

"n" bits out of 32 bits of the device specified by (d) and 1 bit (carry flag M8022) are rotated rightward.



*1 This defines the device that stores the data to be rotated rightward.



Related device

→ **For the carry flag use method, refer to Section 1.3.4.**

Device	Name	Description
M8022	Carry	Turns ON when the bit shifted last from the lowest position is "1".

Cautions

1) Some restrictions to applicable devices
▲1:K4Y○○○, K4M○○○ and K4S○○○ are valid for a 16-bit operation.
K8Y○○○, K8M○○○ and K8S○○○ are valid for a 32-bit operation.
▲2:The FX3U and FX3UC PLCs only are applicable.
2) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
3) In the case of continuous operation type instructions (RCR and DRCR), note that shift and rotation are executed in every scan time (operation cycle).
4) When a device with digit specification is specified as (S), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid (examples: K4Y010 or K8M0).

7.4.4 RCL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This instruction shifts and rotates the bit information leftward by the specified number of bits together with the carry flag.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RCL	16 bits	Continuous	RCL EN ENO n d	RCL(EN,n,d);
RCLP	16 bits	Pulse	RCLP EN ENO n d	RCLP(EN,n,d);
DRCL	32 bits	Continuous	DRCL EN ENO n d	DRCL(EN,n,d);
DRCLP	32 bits	Pulse	DRCLP EN ENO n d	DRCLP(EN,n,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(n)	Number of bits to be rotated n≤16(16-bit operation), n≤32(32-bit operation)	ANY16	
Output variable	ENO	Execution state	Bit	
	(d)	Word device storing data to be rotated leftward	ANY16	ANY32

3. Applicable devices

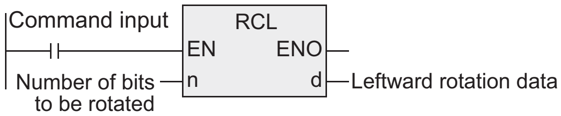
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User			Special Unit	Index				Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)									▲1	▲1	▲1	●	●	●	▲2	▲2	●	●	●					
(n)														●	▲2					●	●			

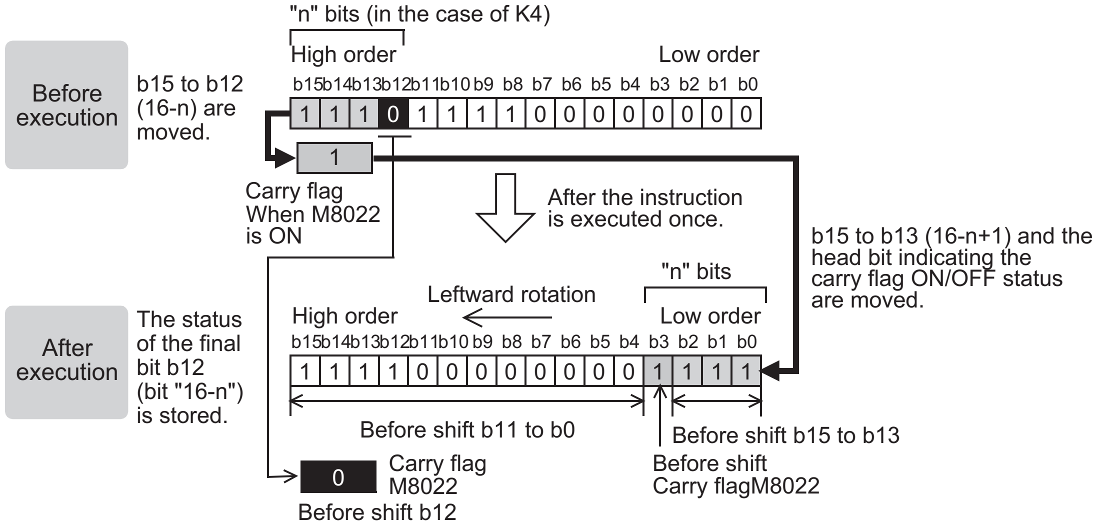
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (RCL, RCLP)

"n" bits out of 16 bits of the device specified by d and 1 bit (carry flag M8022) are rotated leftward.

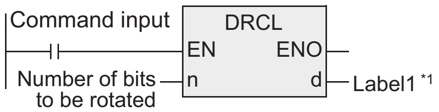




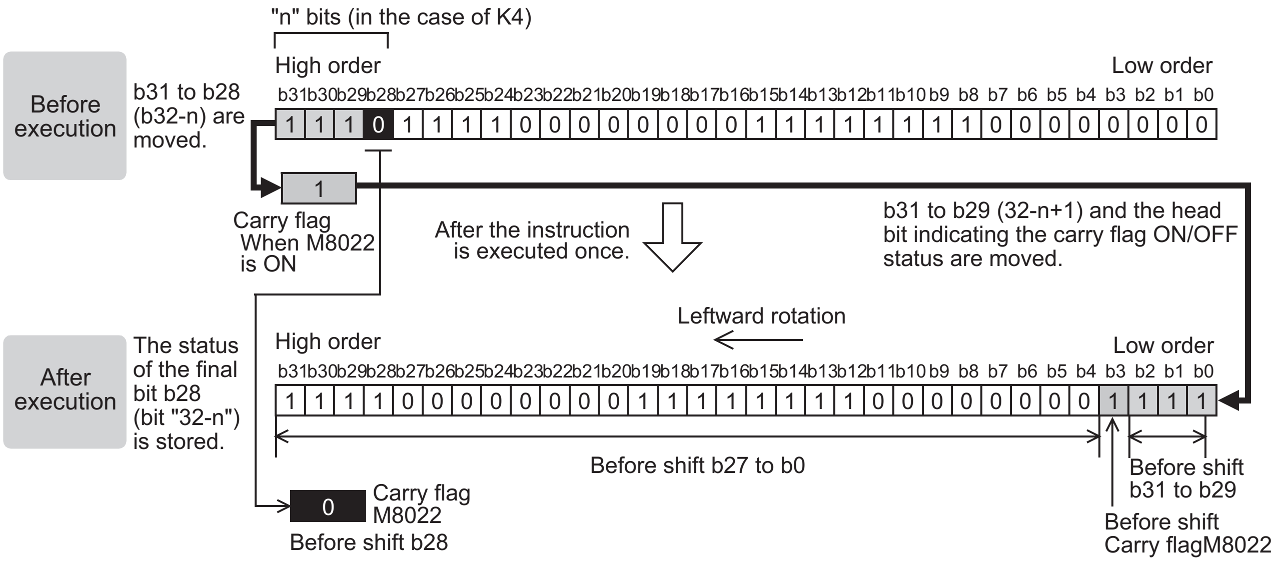
The carry flag is intervened in the rotation loop. If M8022 has been set to ON or OFF before the rotation instruction, the carry flag is transferred to the destination.

2. 32-bit operation (DRCL, DRCLP)

"n" bits out of 32 bits of the device specified by d and 1 bit (carry flag M8022) are rotated leftward.



*1 This defines the device that stores the data to be rotated leftward.



Related device

→ **For the carry flag use method, refer to Section 1.3.4.**

Device	Name	Description
M8022	Carry	Turns ON when the bit shifted last from the highest position is "1".

Cautions

1) Some restrictions to applicable devices
   ▲1:K4Y○○○, K4M○○○ and K4S○○○ are valid for a 16-bit operation.
   K8Y○○○, K8M○○○ and K8S○○○ are valid for a 32-bit operation.
   ▲2:The FX3U and FX3UC PLCs only are applicable.
2) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
3) In the case of continuous operation type instructions (RCL and DRCL), note that shift and rotation are executed in every scan time (operation cycle).
4) When a device with digit specification is specified as (S), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid (examples: K4Y010 or K8M0).

7.4.5 SFTR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction shifts bit devices of the specified bit length rightward by the specified number of bits.
After shift, the bit device specified by (s) is transferred by "n2" bits from the most significant bit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	ST
SFTR	16 bits	Continuous	SFTR block: EN, s, n1, n2 (inputs); ENO, d (outputs)	SFTR(EN,s,n1,n2,d);
SFTRP	32 bits	Pulse	SFTRP block: EN, s, n1, n2 (inputs); ENO, d (outputs)	SFTRP(EN,s,n1,n2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head bit device to be stored to the shift data after rightward shift.	Bit
	(n1)	Bit length of the shift data (see Caution).	ANY16
	(n2)	Number of bits to be shifted rightward (see Caution)	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Head bit device to be shifted rightward	Bit

3. Applicable devices

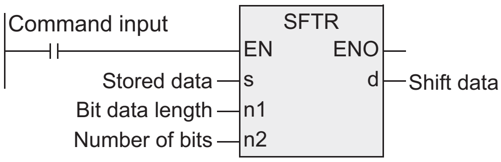
Operand type	Bit Devices: System User							Word Devices: Digit Specification				System User				Special Unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)	●	●	●			●	▲1												●					
(d)		●	●			●													●					
(n1)																				●	●			
(n2)														●	▲2					●	●			

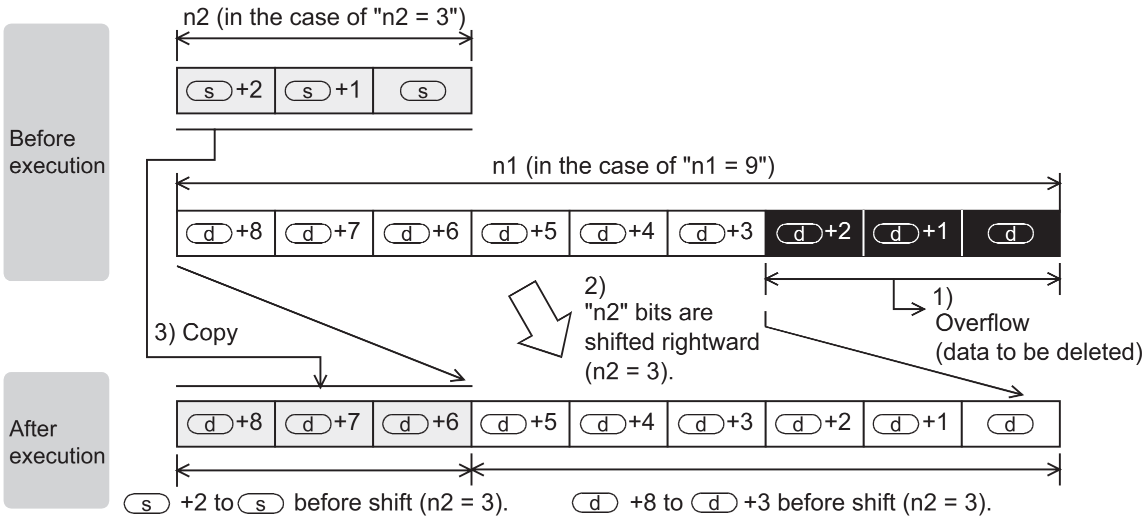
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (SFTR, SFTRP)

For "n1" bits (shift register length) starting from the bit device specified by d, "n2" bits are shifted rightward (1) and 2) shown below).
After shift, "n2" bits from the bit device specified by s are transferred to "n2" bits from d+n1-n2 (3) shown below).





Cautions

1) Some restrictions to applicable devices
▲1:Applicable only to the FX3U and FX3UC PLCs. Not indexed (V, Z).
▲2:The FX3U, FX3UC and FX3G PLCs only are applicable.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
To execute pulse operation, make the instruction execution condition pulse type.
3) Note that "n2" bits are shifted every time the command input turns ON from OFF in SFTRP instruction, but that "n2" bits are shifted in each scan time (operation cycle) in SFTR instruction.
4) Limitation to n1 and n2 differs from one PLC to another.

PLC	Limit
FX3U, FX3UC, FX3G, FX1N, FX2N, FX1NC, FX2NC, FX2, FX2C	n2≤n1≤1024
FX1S, FX0, FX0S, FX0N	n2≤n1≤512

Error

If the transfer source specified by s is equivalent to the shifted device specified by d, an operation error occurs (error code: K6710). (Applicable only to the FX3U and FX3UC PLCs)

7.4.6 SFTL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction shifts bit devices of the specified bit length leftward by the specified number of bits.
After shift, the bit device specified by (s) is transferred by "n2" bits from the least significant bit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
SFTL	16 bits	Continuous	SFTL EN ENO s d n1 n2	SFTL(EN,s,n1,n2,d);
SFTLP	16 bits	Pulse	SFTLP EN ENO s d n1 n2	SFTLP(EN,s,n1,n2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head bit device to be stored to the shift data after leftward shift	Bit
	(n1)	Bit length of the shift data (see Caution).	ANY16
	(n2)	Bit length of the shift data (see Caution).	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Head bit device to be shifted leftward	Bit

3. Applicable devices

Operand type	Bit Devices: System User							Word Devices: Digit Specification				Word Devices: System User				Word Devices: Special Unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)	●	●	●			●	▲1												●					
(d)		●	●			●													●					
(n1)																				●	●			
(n2)														●	▲2					●	●			

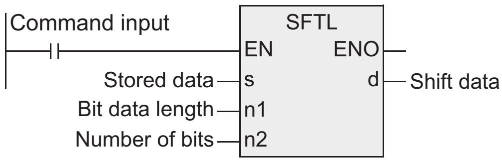
▲: Refer to "Cautions".

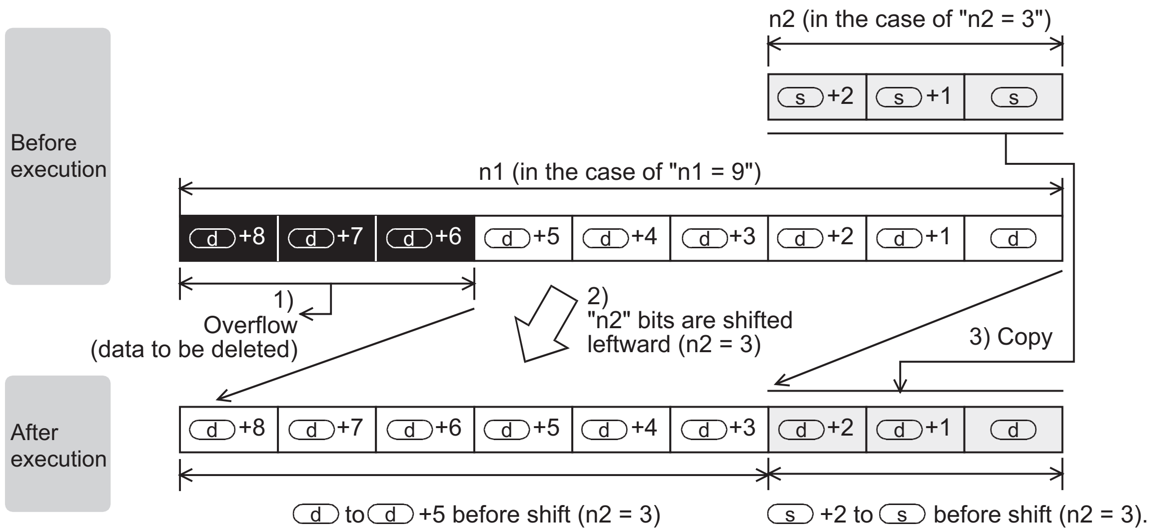
Function and operation explanation

1. 16-bit operation (SFTL, SFTLP)

For "n1" bits (shift register length) starting from the bit device specified by (d), "n2" bits are shifted leftward (1) and 2) shown below).

After shift, "n2" bits from the bit device specified by (s) are transferred to "n2" bits from the bit device specified by (d) (3) shown below).





Cautions

1) Some restrictions to applicable devices
▲1:Applicable only to the FX3U and FX3UC PLCs. Not indexed (V, Z).
▲2:The FX3U, FX3UC and FX3G PLCs only are applicable.
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
To execute pulse operation, make the instruction execution condition pulse type.
3) Note that "n2" bits are shifted every time the command input turns ON from OFF in SFTLP instruction, but that "n2" bits are shifted in each operation cycle in SFTL instruction.
4) Limitation to n1 and n2 differs from one PLC to another.

PLC	Limit
FX3U, FX3UC, FX3G, FX1N, FX2N, FX1NC, FX2NC, FXU, FX2C	n2≤n1≤1024
FX1S, FX0, FX0S, FX0N	n2≤n1≤512

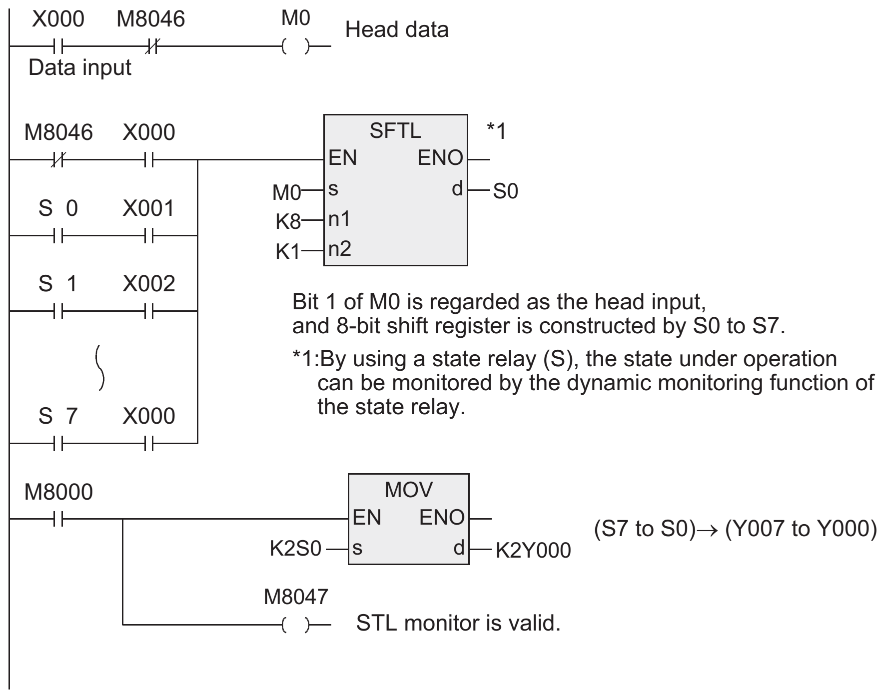
Error

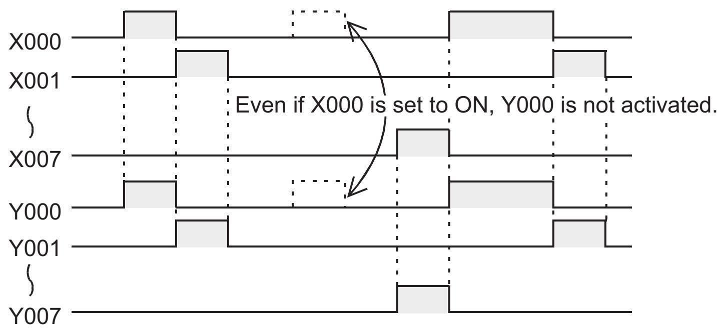
If the transfer source specified by (s) is equivalent to the shifted device specified by (d), an operation error occurs (error code: K6710). (Applicable only to the FX3U and FX3UC PLCs)

Program examples(Conditional stepping of 1-bit data)

By setting X000 to X007 to ON in turn, Y000 to Y007 are activated in turn.
If the order is wrong, activation is disabled.

[Structured ladder]





[ST]

```
M0:= X000 AND NOT M8046;
SFTL((NOT M8046 AND X000) OR (S0 AND X001) ··· OR (S1 AND X002) R(S7 AND X000), M0, K8, K1, S0);
MOV(M8000, K2S0, K2Y000);
M8047:= M8000;
```

7.4.7 WSFR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction shifts word devices with "n1" data length rightward by "n2" words.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
WSFR	16 bits	Continuous	WSFR: EN, s, n1, n2 → ENO, d	WSFR(EN,s,n1,n2,d);
WSFRP	16 bits	Pulse	WSFRP: EN, s, n1, n2 → ENO, d	WSFRP(EN,s,n1,n2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device to be stored to the shift data after rightward shift	ANY16
	n1	Word data length of the shift data (n2 ≤ n1 ≤ 512)	ANY16
	n2	Number of words to be shifted rightward (n2 ≤ n1 ≤ 512)	ANY16
Output variable	ENO	Execution state	Bit
	d	Head word device storing data to be shifted rightward	ANY16

3. Applicable devices

Operand type	Bit Devices: System User							Word Devices: Digit Specification				Word Devices: System User				Word Devices: Special Unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲1	▲2			●					
d									●	●	●	●	●	●	▲1	▲2			●					
n1																				●	●			
n2														●	▲1					●	●			

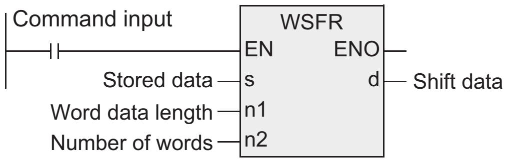
▲: Refer to "Cautions".

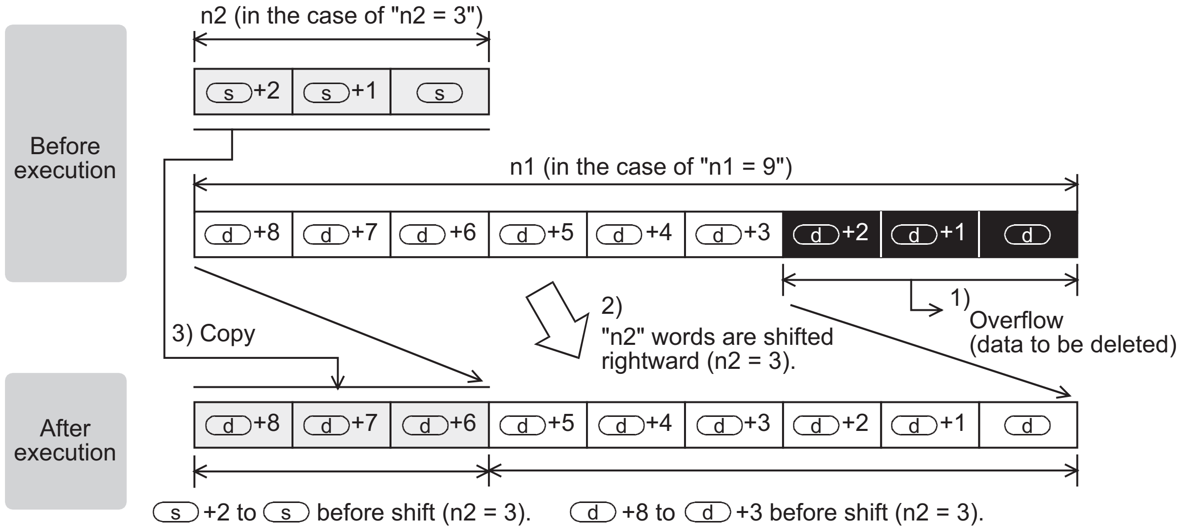
Function and operation explanation

1. 16-bit operation (WSFR, WSFRP)

For "n1" word devices starting from the device specified by (d), "n2" words are shifted rightward (1) and 2) shown below)

After shift, "n2" words starting from the device specified by (s) are transferred to "n2" words starting from the device specified by [(d)+n1-n2] (3) shown below).





Cautions

1) Note that "n2" words are shifted when the drive input turns ON in WSFRP instruction, but that "n2" words are shifted in each operation cycle in WSFR instruction.

2) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

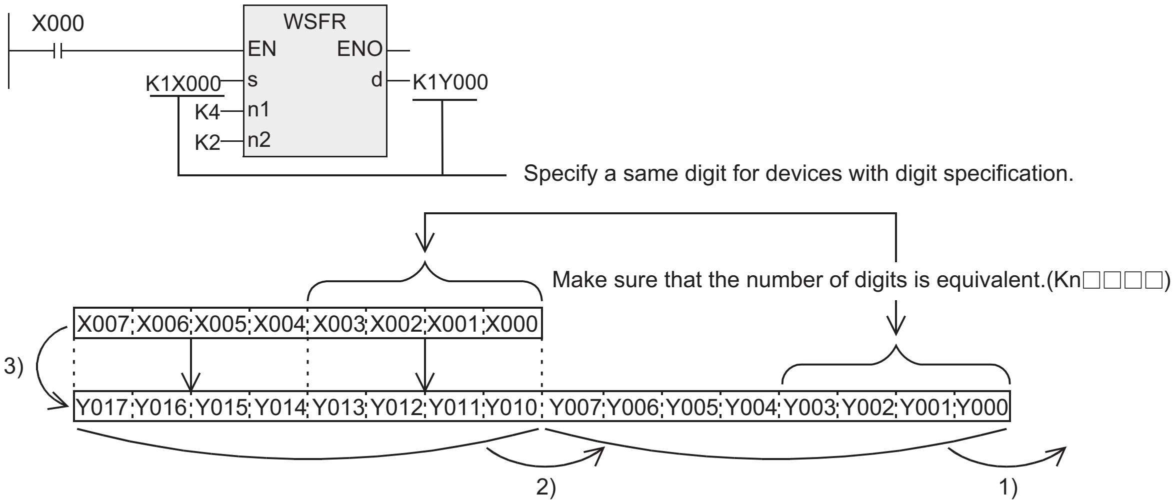
Error

If the transfer source device specified by (s) is equivalent to the shifted device specified by (d), an operation error occurs (error code: K6710).

Program examples

1. Shifting devices with digit specification

[Structured ladder]



[ST]

```
WSFR(X000, K1X000, K4, K2, K1Y000);
```

7.4.8 WSFL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction shifts the word data information leftward by the specified number of words.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
WSFL	16 bits	Continuous	WSFL: EN, ENO, s, d, n1, n2	WSFL(EN,s,n1,n2,d);
WSFLP	16 bits	Pulse	WSFLP: EN, ENO, s, d, n1, n2	WSFLP(EN,s,n1,n2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device to be stored to the shift data after leftward shift	ANY16
	n1	Word data length of the shift data (n2 ≤ n1 ≤ 512)	ANY16
	n2	Number of words to be shifted leftward (n2 ≤ n1 ≤ 512)	ANY16
Output variable	ENO	Execution state	Bit
	d	Head word device storing data to be shifted leftward	ANY16

3. Applicable devices

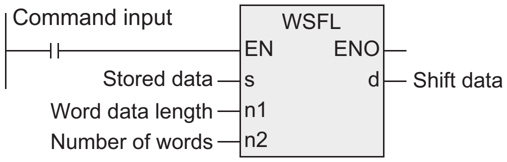
Operand type	Bit Devices							Word Devices												Others					
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲1	▲2			●					
d									●	●	●	●	●	●	▲1	▲2			●					
n1																				●	●			
n2														●	▲1					●	●			

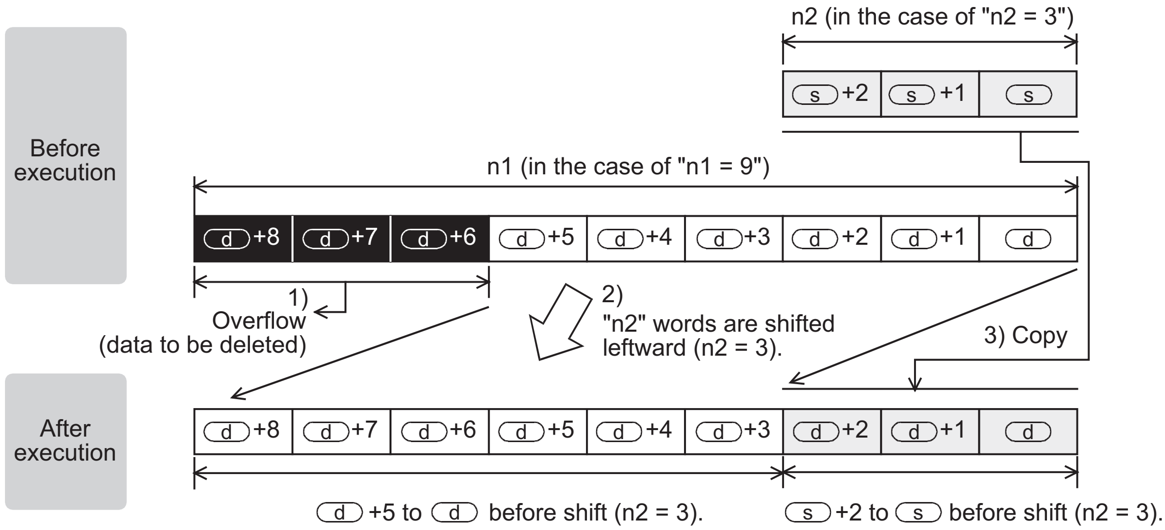
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (WSFL, WSFLP)

For "n1" word devices starting from the device specified by (d), "n2" words are shifted leftward (1) and 2) shown below).
After shift, "n2" words starting from the device specified by (s) are shifted to "n2" words starting from the device specified by (d).





Cautions

1) Note that "n2" words are shifted every time the drive input turns ON from OFF in WSFLP instruction, but that "n2" words are shifted in each operation cycle in WSFL instruction.
2) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

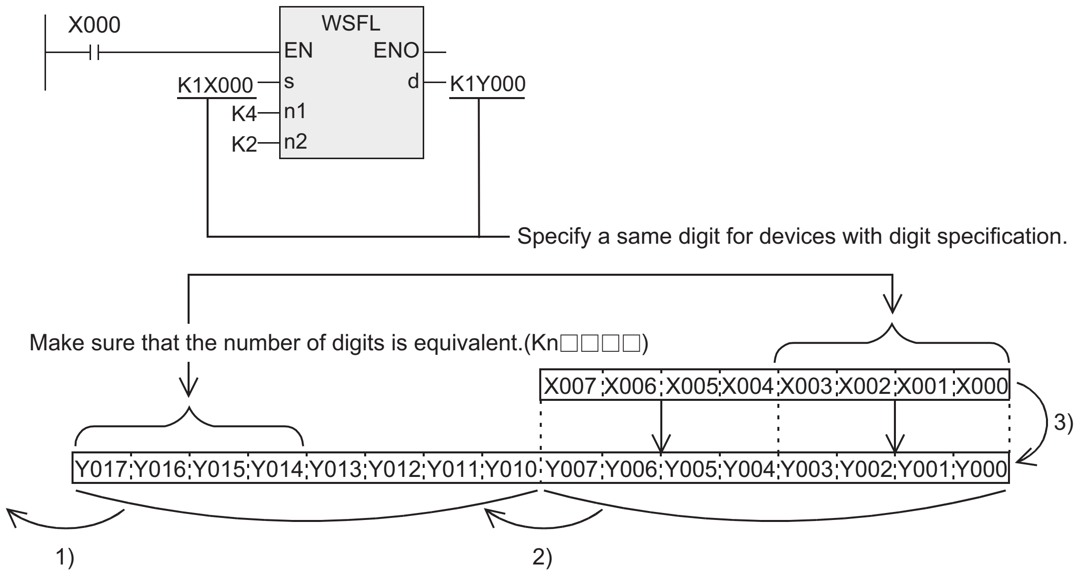
Error

If the transfer source specified by (s) is equivalent to the shifted device specified by (d), an operation error occurs (error code: K6710).

Program examples

1. Shifting devices with digit specification

[Structured ladder]



[ST]

```
WSFL(X000, K1X000, K4, K2, K1Y000);
```

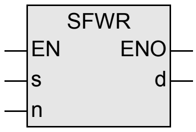
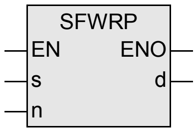
7.4.9 SFWR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This instruction writes data for first-in first-out (FIFO) and first-in last-out (FILO) control.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SFWR	16 bits	Continuous		SFWR(EN,s,n,d);
SFWRP	16 bits	Pulse		SFWRP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Word device storing data to be put in first	ANY16
	n	Number of store points (for pointer, value is added by "+1".) $2 \leq n \leq 512$	ANY16
Output variable	ENO	Execution state	Bit
	d	Head word device storing and shifting data.	ANY16

3. Applicable devices

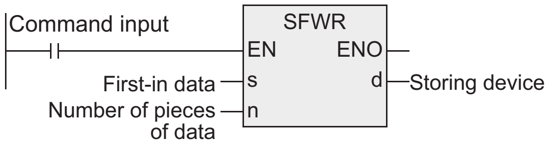
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
d									●	●	●	●	●	●	▲1	▲2			●					
n																				●	●			

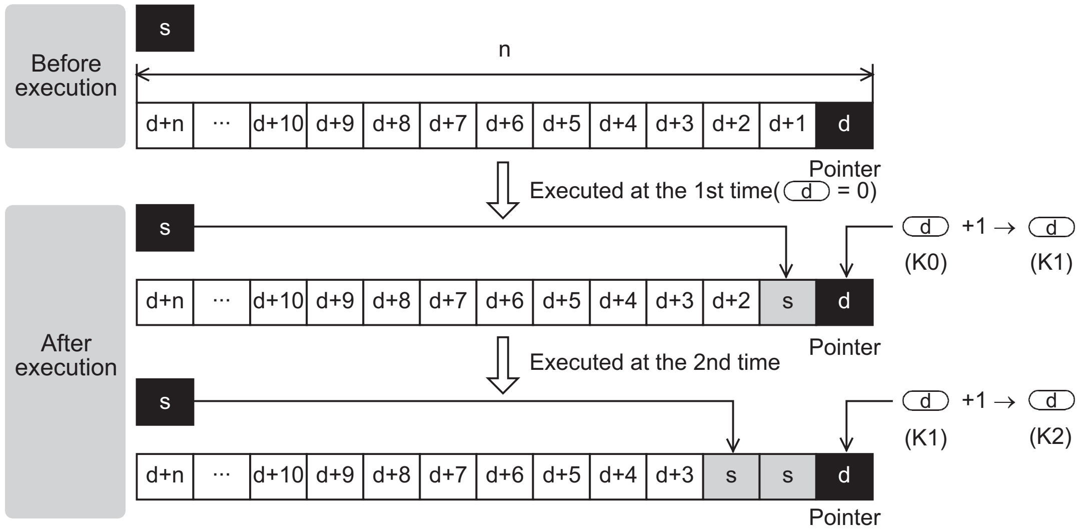
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (SFWR, SFWRP)

The contents of the device specified by s are written to "n-1" devices from the device specified by d +1, and "1" is added to the number of data stored in the device specified by d.
For example, when the device specified by d is "0", the contents of the device specified by s are written to the device specified by d +1. When the device specified by d is "1", the contents of the device specified by s are written to the device specified by d +2.





1) When X000 turns ON from OFF, the contents of the device specified by s are stored to the device specified by d +1. So the contents of the device specified by d +1 become equivalent to the contents of the device specified by s.
2) When the contents of the device specified by s are changed and then the command input is set to ON from OFF again, the new contents of the device specified by s are stored to the device specified by d +2. So the contents of the device specified by d +2 become equivalent to the contents of the device specified by s. (When the continuous operation type SFWR instruction is used, the contents are stored in each operation cycle. Use the pulse operation type SFWRP instruction in programming.)
3) Data are stored from the right end in the same way, and the number of stored data is indicated by the contents of the pointer specified by d.

Related device

→ **For the carry flag use method, refer to Section 1.3.4.**

Device	Name	Description
M8022	Carry	When the contents of the pointer specified by d exceeds "n-1", no operation is executed (data is not written) and the carry flag M8022 turns ON.

Related instructions

Instruction	Description
SFRD	Shift read (for FIFO control)
POP	Last-in data read (for FILO control)

Cautions

1. In the case of continuous operation type (SFWR) instruction.

Note that data are stored (overwritten) in each scan time (operation cycle).

2. Some restrictions to applicable devices

▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

1. Example of first-in first-out control

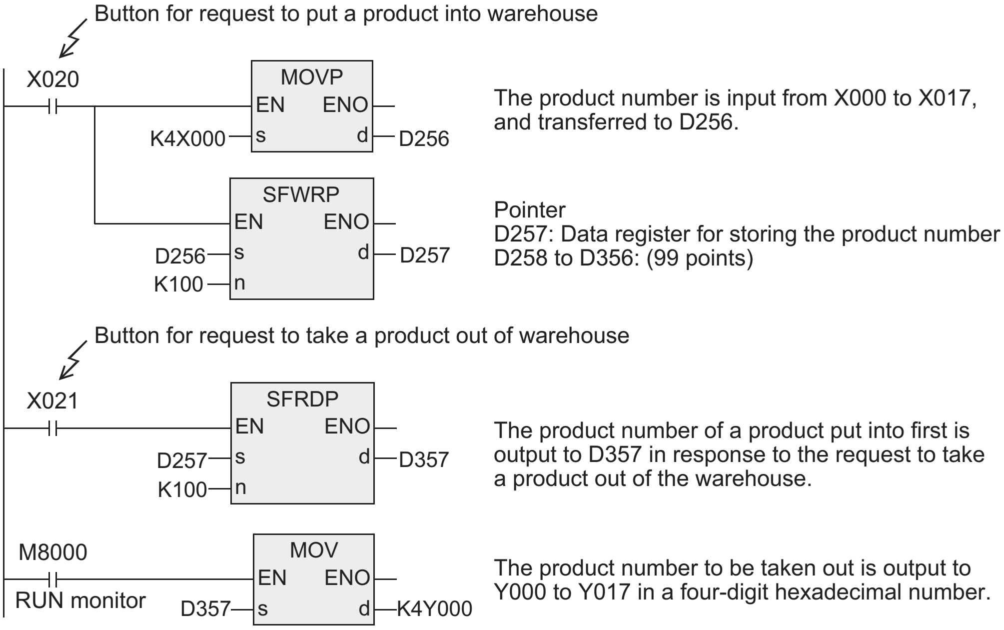
→ **For a program example of FILO, refer to Section 7.21.3.**

In the example below, the shift write (SFWR) and shift read (SFRD) instructions are used.

1) Contents of operation
   - In this circuit example, a product number to be taken out now is output according to "first-in first-out" rule while products which were put into a warehouse with their product numbers registered are taken out of the warehouse.
   - The product number is hexadecimal, and up to 4 digits. Up to 99 products can be stored in the warehouse.

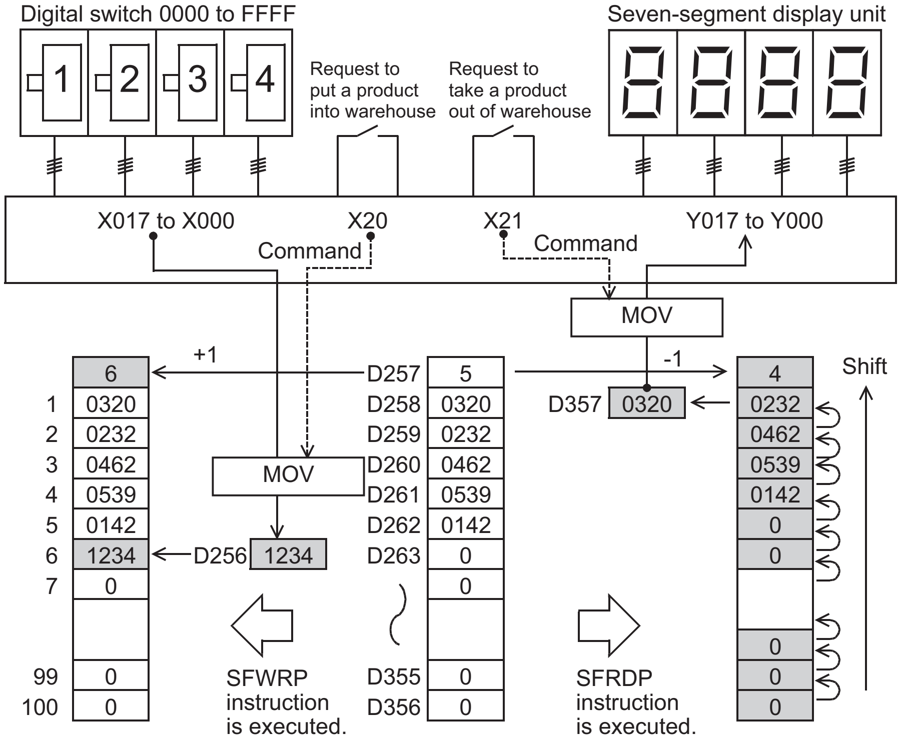
2) Program

[Structured ladder]



[ST]

```
MOVP(X020, K4X000, D256);
SFWRP(X020, D256, K100, D257);
SFRDP(X021, D257, K100, D357);
MOV(M8000, D357, K4Y000);
```



7.4.10 SFRD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This instruction reads data for first-in first-out control.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SFRD	16 bits	Continuous	SFRD EN ENO s d n	SFRD(EN,s,n,d);
SFRDP	16 bits	Pulse	SFRDP EN ENO s d n	SFRDP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head word device storing data	ANY16
	n	Number of store points (for pointer, value is added by "+1".) 2 ≤ n ≤ 512	ANY16
Output variable	ENO	Execution state	Bit
	d	Word device storing data taken out first	ANY16

3. Applicable devices

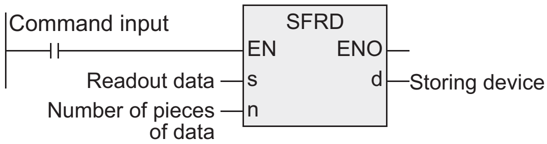
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s									●	●	●	●	●	●	▲1	▲2			●					
d									●	●	●	●	●	●	▲1	▲2	●	●	●					
n																				●	●			

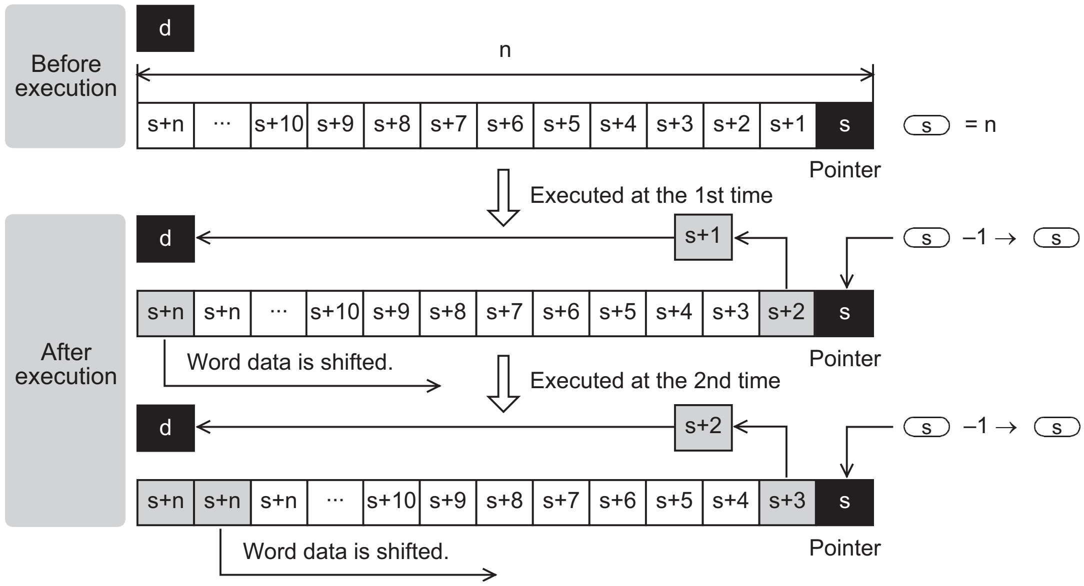
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (SFRD, SFRDP)

The data of the device specified by [s1+1] written in turn by SFWR instruction is transferred (read) to the device specified by d, and "n-1" words from the device specified by s1+1 are shifted rightward by 1 word. "1" is subtracted from the number of data stored in the device specified by s.





1) When the command contact turns ON, the contents of the device specified by [s+1] are transferred (read) to the device specified by d.
2) Accompanied by this transfer, the contents of the pointer specified by s decrease, and the data on the left side are shifted rightward by 1 word. (When the continuous operation type SFRD instruction is used, the contents are shifted in turn in each operation cycle. Use the pulse operation type SFRDP instruction in programming.)

Related device

→ **For the zero flag use method, refer to Section 1.3.4.**

Device	Name	Description
M8020	Zero	Data is always read from the device specified by [s+1]. When the contents of the pointer specified by s become "0", the zero flag M8020 turns ON.

Related instructions

Instruction	Description
SFWR	Shift write (for FIFO/FILO control)
POP	Last-in data read (for FILO control)

Cautions

1) The contents of the device specified by [s+n] do not change by reading.
2) In the case of continuous operation type (SFRD) instruction, data is read in turn in each scan time (operation cycle), but the contents of the device specified by [s+n] do not change.
3) When pointer specified by s is "0", data is not processed, and the contents of the device specified by d do not change.
4) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

Refer to the program example provided for SFWR instruction.

7.5 Data Operation

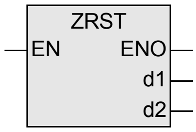
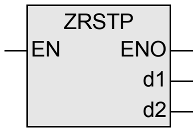
7.5.1 ZRST

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction resets devices located in a zone between two specified devices at one time.
Use this instruction for restarting operation from the beginning after pause or after resetting control data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ZRST	16 bits	Continuous		ZRST(EN,d1,d2);
ZRSTP	16 bits	Pulse		ZRSTP(EN,d1,d2);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	(d1)	Head device to be reset at one time	ANY_SIMPLE
	(d2)	Last device to be reset at one time	ANY_SIMPLE

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d1)		●	●			●						●	●	●	▲1	▲2			●					
(d2)		●	●			●						●	●	●	▲1	▲2			●					

▲: Refer to "Cautions".

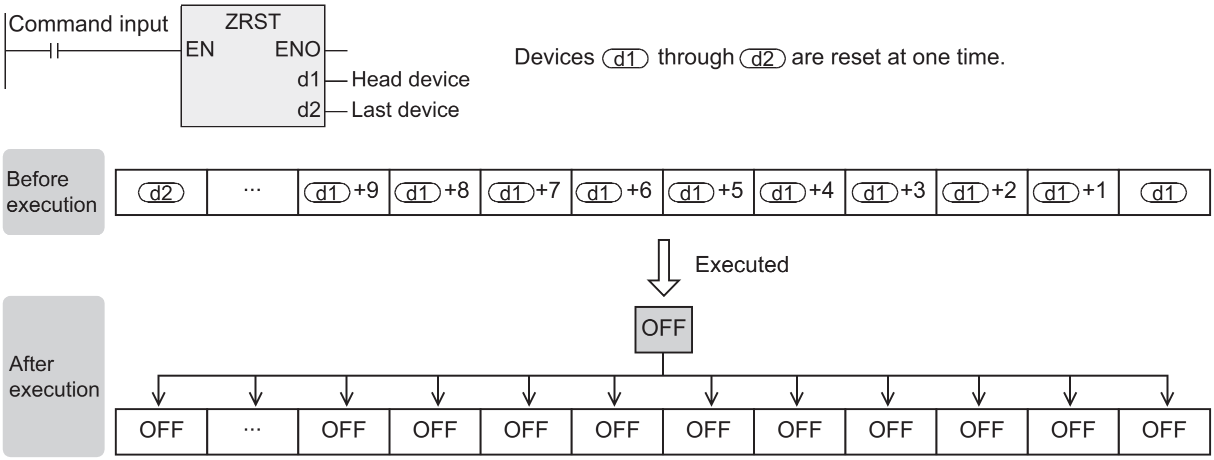
Function and operation explanation

1. 16-bit operation (ZRST, ZRSTP)

Same type of devices specified by d1 to d2 are reset at one time.

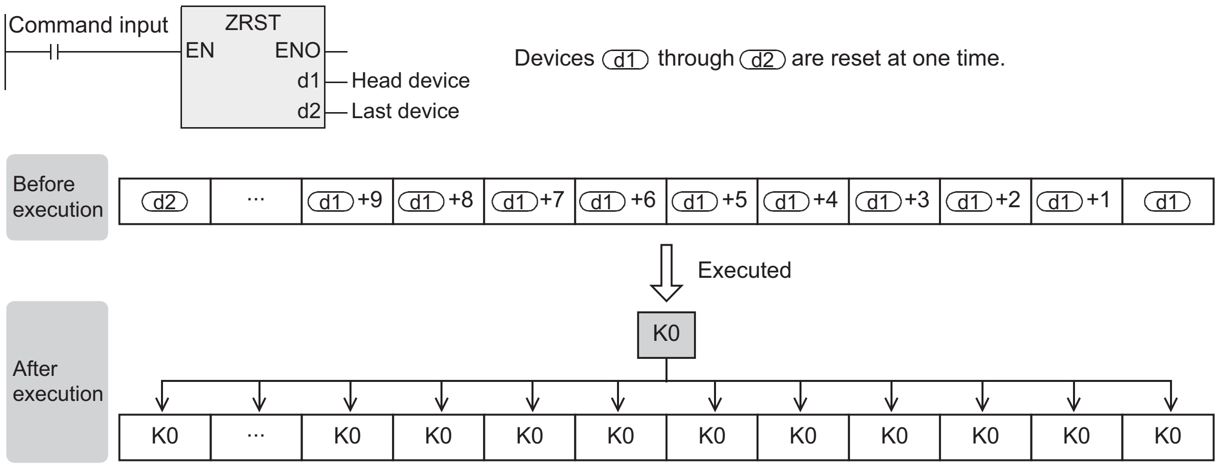
When the devices specified by d1 and d2 are bit devices

"OFF (reset)" is written to the entire range from the devices specified by d1 to d2 at one time.



When the devices specified by d1 and d2 are word devices

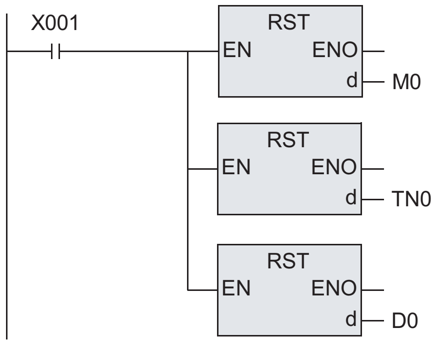
"K0" is written to the entire range from the devices specified by d1 to d2 at one time.



Related instructions

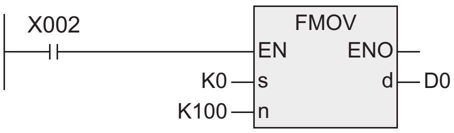
1. RST

As an independent reset instruction for devices, RST instruction can be used for bit devices (Y, M and S) and word devices (T, C, D and R).



2. FMOV

FMOV instruction is provided to write a constant (example: K0) at one time. By using this instruction, "0" can be written to word devices (KnY, KnM, KnS, T, C, D and R) at one time.



K0 is written to D0 to D99 at one time.

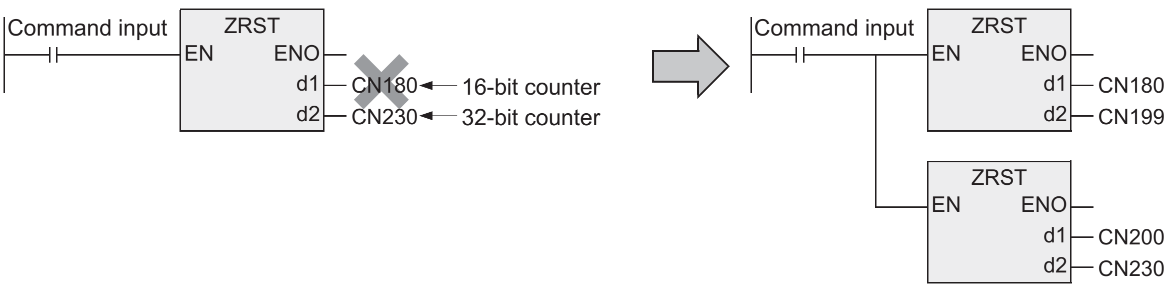
Cautions

1) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
To execute pulse operation, make the instruction execution condition pulse type.

2) Specify same type of devices in the devices specified by (d1) and (d2). The device number of the device specified by (d1) should be smaller than or equal to the device number of the device specified by (d2).
If the device number of the device specified by (d1) is larger than the device number of the device specified by (d2), only one device specified by (d1) is reset.

3) When specifying the high-speed counter, ZRST instruction is handled as the 16-bit type, but 32-bit counters can be specified in (d1) and (d2).
However, it is not possible to specify a 16-bit counter in the device specified by (d1) and specify a 32-bit counter in the device specified by (d2); (d1) and (d2) should be a same type.

Program example



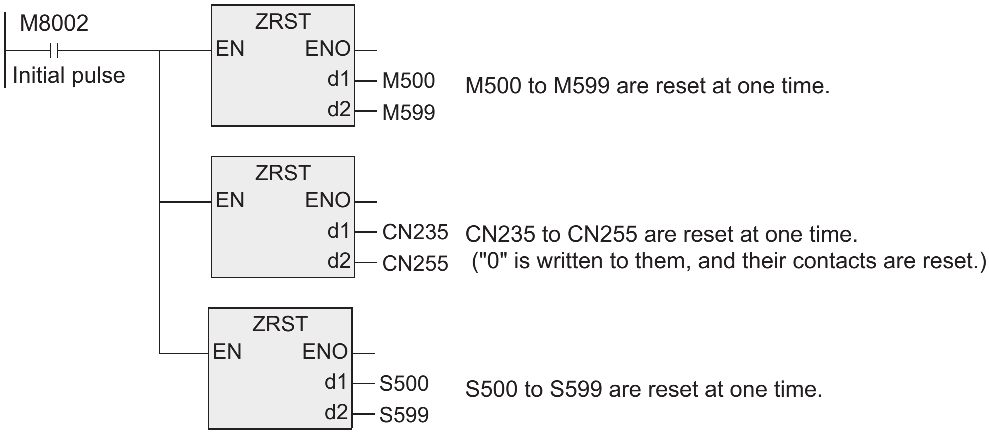
4) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

1. When using devices in the latch area as non-latch type devices

When the power of the PLC is turned ON or when the PLC mode is changed to RUN, the specified ranges of bit devices and word devices are reset at one time.

[Structured ladder]



[ST]

```
ZRST(M8002, M500, M599);
ZRST(M8002, CN235, CN255);
ZRST(M8002, S500, S599);
```

7.5.2 DECO

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction converts numeric data into ON bit.
A bit number which is set to ON by this instruction indicates a numeric value

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DECO	16 bits	Continuous	DECO EN ENO s d n	DECO(EN,s,n,d);
DECOP	16 bits	Pulse	DECOP EN ENO s d n	DECOP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Data to be decoded or word device storing data	ANY_SIMPLE
	n	Number of bits of device storing the decoding result (n = 1 to 8). (No processing is executed in the case of "n = 0".)	ANY16
Output variable	ENO	Execution state	Bit
	d	Device storing decoding result	ANY_SIMPLE

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●	●	●			●						●	●	●	▲1	▲2	●	●	●	●	●			
d		●	●			●						●	●	●	▲1	▲2			●					
n																				●	●			

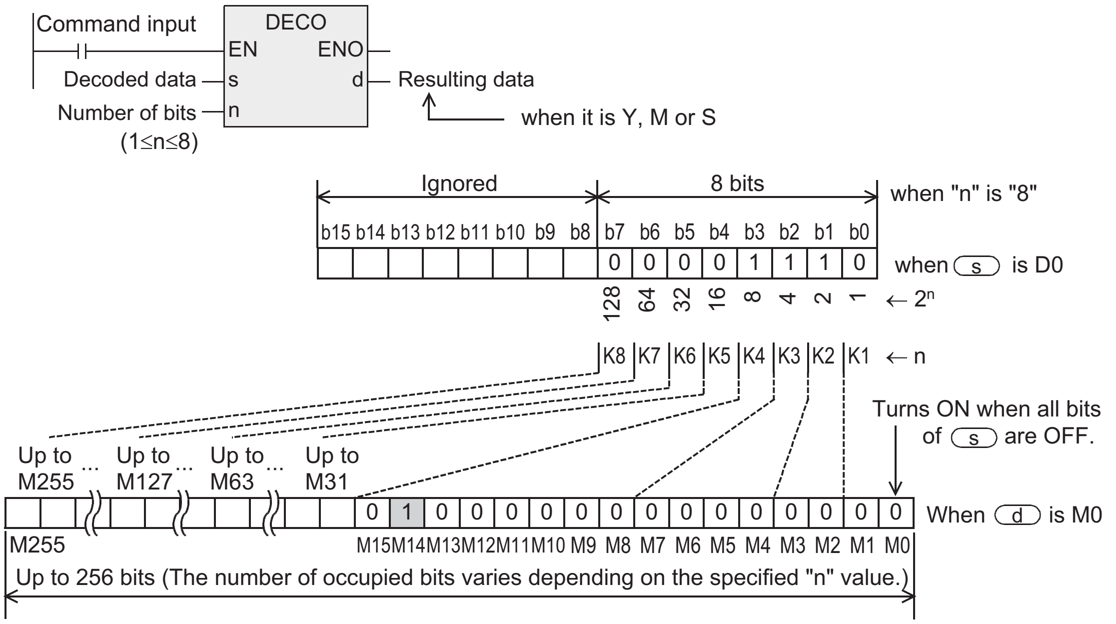
▲: Refer to "Cautions".

Function and operation explanation

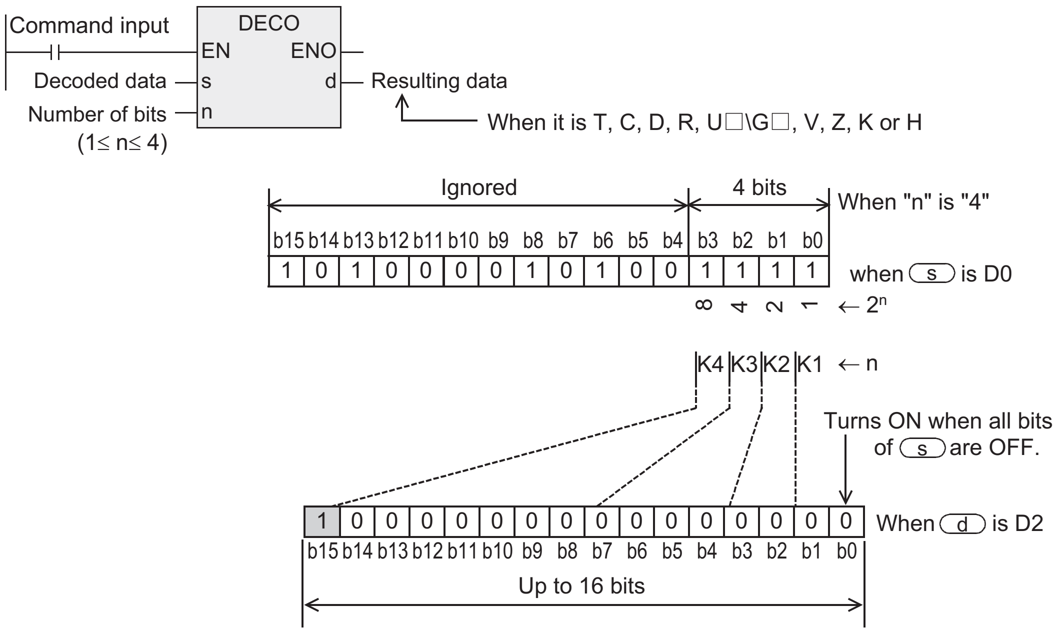
1. 16-bit operation (DECO, DECOP)

One bit among the devices specified by [(d) to (d)+2^n-1] is set to ON according to the value of the device specified by (s).

1) When the device specified by (d) is a bit device (1 ≤ n ≤ 8)
   n bits (1 ≤ n ≤ 8) of a device specified by (s) is decoded to the device specified by (d).
   - When all bits of the devices specified by (s) are "0", the bit device specified by (d) turns ON.
   - When "n" is "8", the bit device specified by (d) occupies maximum 2^8 = 256 bits.



2) When the device specified by (d) is word device (1 ≤ n ≤ 4)
   n bits on the low-order side of the device specified by (s) is decoded to the device specified by (d).
   - When all bits of the device specified by (s) are "0", b0 of the word device specified by (d) turns ON.
   - In the case of "n ≤ 3", all of high-order bits of the device specified by (d) become "0" (turn OFF).



Cautions

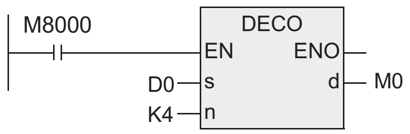
1) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
2) While the command input is OFF, the instruction is not executed. The activated decode output is held in the previous ON/OFF status.
3) When "n" is "0", the instruction executes no processing.
4) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

1. When setting bit devices to ON according to the value of a data register

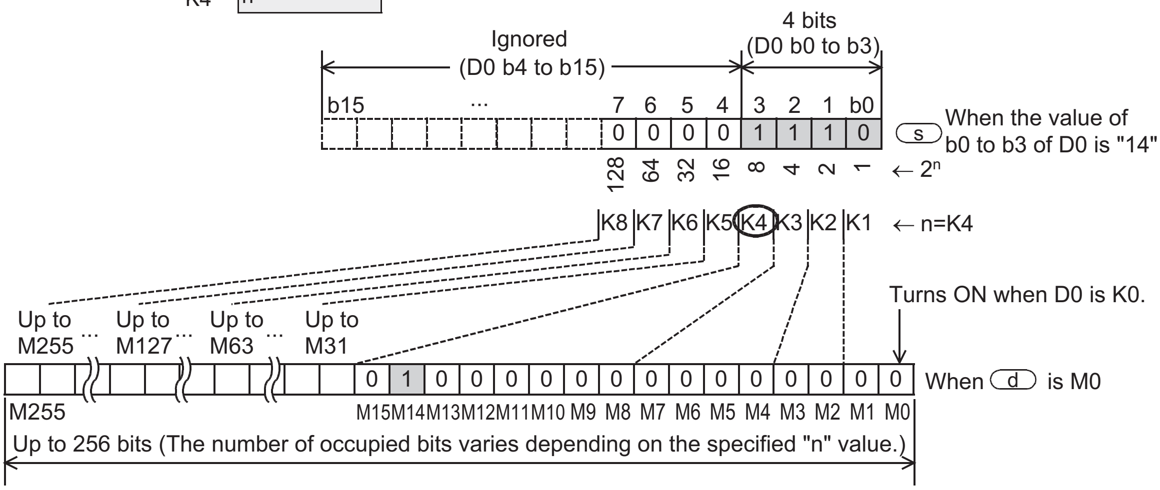
The value of D0 (whose current value is "14" in this example) is decoded to M0 to M15.

[Structured ladder]



[ST]

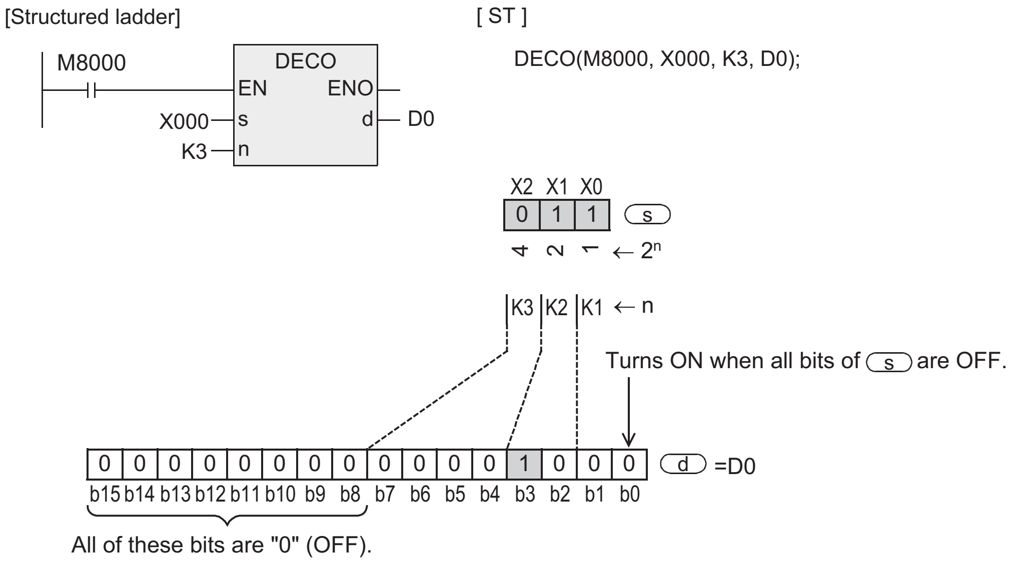
```
DECO(M8000, D0, K4, M0);
```



- When the value of b0 to b3 of D0 is "14 (=0+2+4+8)", M14 (which is the 15th from M0) becomes "1" (turns ON).
- When the value of D0 is "0", M0 becomes "1" (turns ON).
- When "n" is set to "K4", any one point among M0 to M15 turns ON according to the value of D0 (0 to 15).
- By changing "n" from K1 to K8, D0 can correspond to numeric values from 0 to 255.
  However, because the device range of the device specified by (d) is occupied for decoding accordingly, such device range should not be used for another control.

2. Turning ON the bit out of word devices according to the contents of bit devices

The value expressed by X000 to X002 is decoded to D0 (X000 and X001 are ON, and X002 is OFF in this example).



- When the values expressed by X000 to X002 are "3 (=1+2+0)", b3 (which is the 4th from b0) becomes "1" (turns ON).
- When all of X000 to X002 are "0" (OFF), b0 becomes "1" (turns ON).

7.5.3 ENCO

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction obtains positions in which bits are ON in data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
ENCO	16 bits	Continuous	ENCO (EN, s, n / ENO, d)	ENCO(EN,s,n,d);
ENCOP	16 bits	Pulse	ENCOP (EN, s, n / ENO, d)	ENCOP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Data to be encoded or word device storing data	ANY_SIMPLE
	n	Number of bits of device storing the encoding result (n = 1 to 8) (When "n" is "0", no processing is executed.)	ANY16
Output variable	ENO	Execution state	Bit
	d	Device storing the encoding result	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●	●	●		●							●	●	●	▲1	▲2	●	●	●					
d												●	●	●	▲1	▲2	●	●	●					
n																				●	●			

▲: Refer to "Cautions".

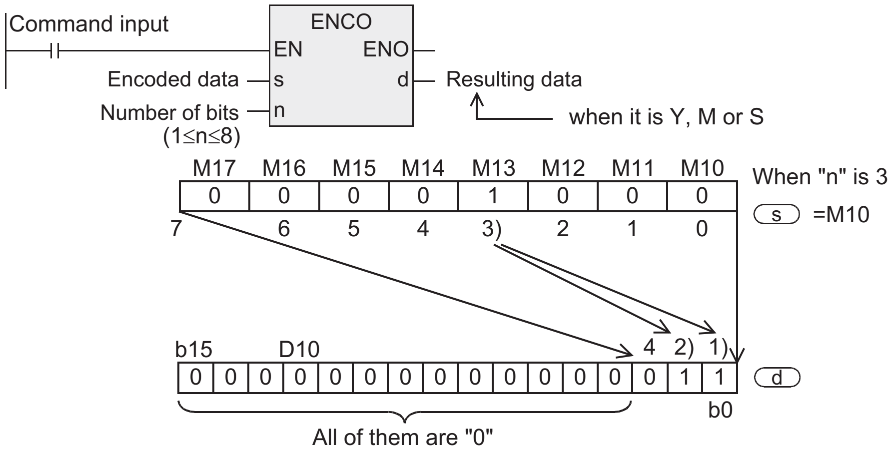
Function and operation explanation

1. 16-bit operation (ENCO, ENCOP)

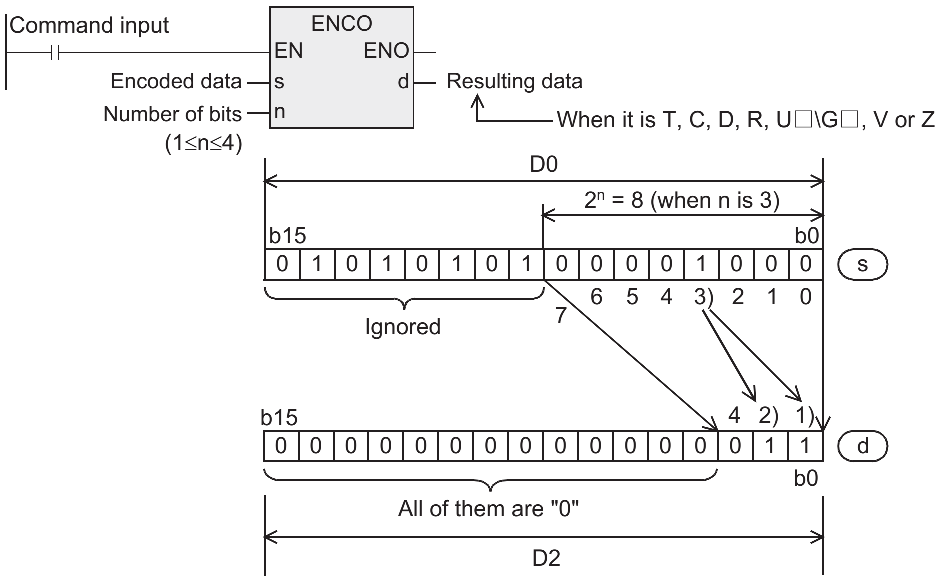
The 2^n bit of the data specified by (s) is encoded, and the result value is stored to the device specified by (d).

This instruction converts data into binary data according to a bit position in the ON status.

1) When the device specified by (s) is a bit device (1 ≤ n ≤ 8)
   ON bit positions among "2^n" bits (1 ≤ n ≤ 8) from the device specified by (s) are encoded to the device specified by (d).
   - When "n" is "8", the device specified by (s) occupies 2^8 = 256 bits (which is the maximum value).
   - The encoding result of the device specified by (d) is all "0" (OFF) from the most significant bit to the low-order bit "n".



2) When the device specified by (s) is a word device (1 ≤ n ≤ 4)
   ON bit positions among "2n" bits (1 ≤ n ≤ 4) from a device specified by (s) are encoded to the device specified by (d).
   - The encoding result of the device specified by (d) is all "0" (OFF) from the most significant bit to the low-order bit "n".



Cautions

1) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
2) When two or more bits are ON in the data specified by (S), the low-order side is ignored, and only the ON position on the high-order side is encoded.
3) When the command input is OFF, the instruction is not executed. Activated encode outputs are latched in the previous ON/OFF status.
4) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

7.5.4 SUM

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction counts the number of "1" (ON) bits in the data of a specified device.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
SUM	16 bits	Continuous	SUM — EN / ENO, s / d	SUM(EN,s,d);
SUMP	16 bits	Pulse	SUMP — EN / ENO, s / d	SUMP(EN,s,d);
DSUM	32 bits	Continuous	DSUM — EN / ENO, s / d	DSUM(EN,s,d);
DSUMP	32 bits	Pulse	DSUMP — EN / ENO, s / d	DSUMP(EN,s,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Word device storing the data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Word device storing the result data	ANY16	ANY32

3. Applicable devices

Operand type	Bit Devices: System User							Word Devices: Digit Specification				Word Devices: System User				Word Devices: Special Unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

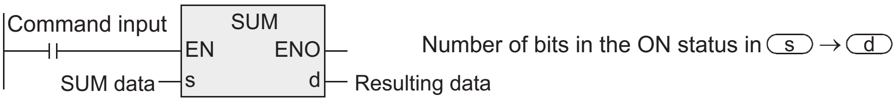
▲: Refer to "Cautions".

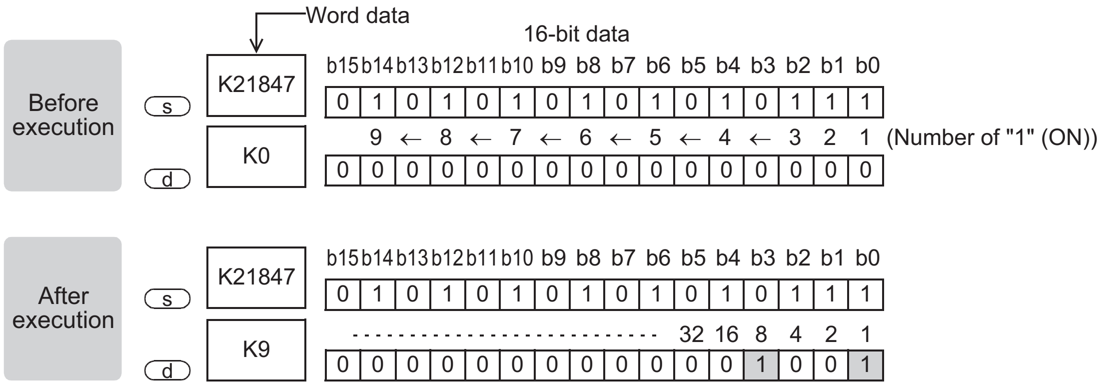
Function and operation explanation

1. 16-bit operation (SUM ,SUMP)

The number of bits in the ON status in the device specified by (s) is counted, and stored to the device specified by (d).

- When all bits are 0 (OFF) in the device specified by (s), the zero flag M8020 turns ON.

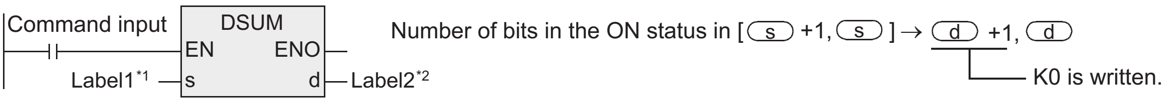




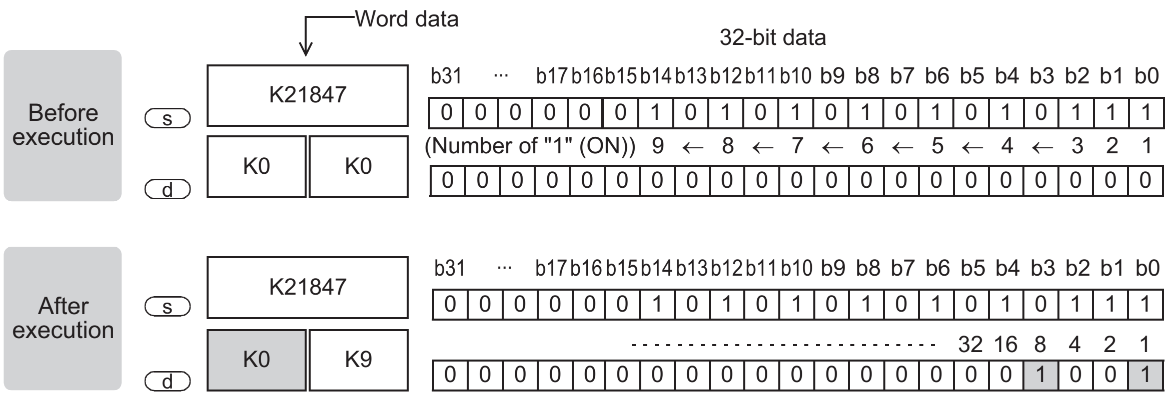
2. 32-bit operation (DSUM, DSUMP)

The number of bits in the ON status in the device specified by (s) is counted, and stored to the device specified by (d).

- The number of bits in the ON status is stored in the device specified by (d), and K0 is stored in (d)+1.
- When all bits are 0 (OFF) in the device specified by (s), the zero flag M8020 turns ON.



*1 This defines the data that executes SUM_2 or the device number storing the source data.
*2 This defines the device that stores the result of executing SUM_2.



3. Operation result of the device specified by (d) according to the value specified by (s) (in 16-bit operation)

(s)																		(d)	M8020 Zero flag
Bit device																Word device			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	Decimal	Hexadecimal		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000	0	ON
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0001	1	OFF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0002	1	OFF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	3	0003	2	OFF
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	4	0004	1	OFF
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	5	0005	2	OFF
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	6	0006	2	OFF
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	7	0007	3	OFF
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	8	0008	1	OFF
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	9	0009	2	OFF
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	10	000A	2	OFF
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	11	000B	3	OFF
⋮																⋮	⋮	⋮	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	-5	FFFB	15	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	-4	FFFC	14	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	-3	FFFD	15	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	-2	FFFE	15	OFF
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	FFFF	16	OFF

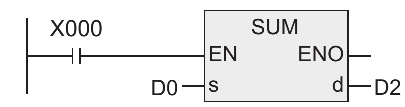
Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) While the command input is OFF, the instruction is not executed. The output of the number of bits in the ON status is latched in the previous status.
3) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

When X000 is ON, the number of bits in the ON status in D0 is counted, and stored to D2.

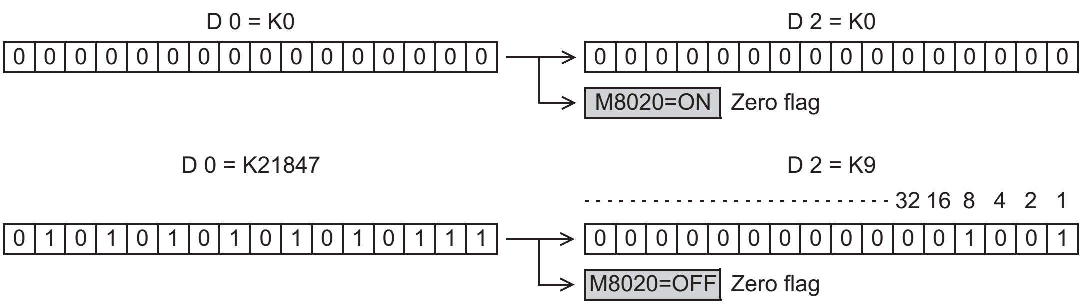
[Structured ladder]



The number of "1" in D0 is stored to D2.

[ST]

```
SUM_2(X000, D0, D2);
```



7.5.5 BON

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction checks whether a specified bit position in a device is ON or OFF.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BON	16 bits	Continuous	BON [EN, s, n → ENO, d]	BON(EN,s,n,d);
BONP	16 bits	Pulse	BONP [EN, s, n → ENO, d]	BONP(EN,s,n,d);
DBON	32 bits	Continuous	DBON [EN, s, n → ENO, d]	DBON(EN,s,n,d);
DBONP	32 bits	Pulse	DBONP [EN, s, n → ENO, d]	DBONP(EN,s,n,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Word device storing the data	ANY16	ANY32
	n	Bit position to be checked [n=0 to 15 (16-bit instruction), n=0 to 31 (32-bit instruction)]	ANY16	
Output variable	ENO	Execution state	Bit	
	d	Bit device to be driven	Bit	

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
d		●	●			●	▲1												●					
n														●	▲2					●	●			

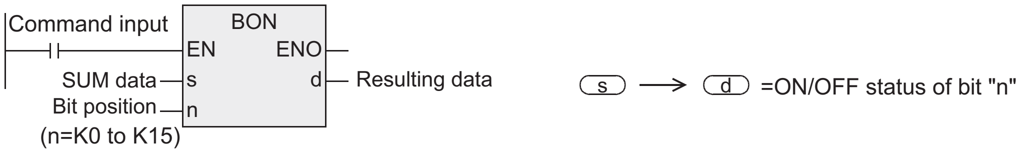
▲: Refer to "Cautions".

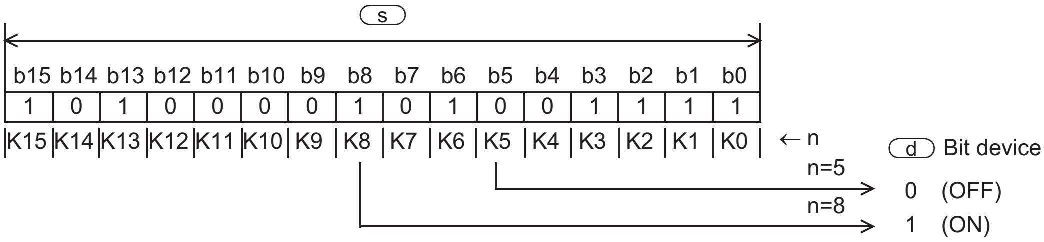
Function and operation explanation

1. 16-bit operation(BON, BONP)

The status (ON or OFF) of the bit "n" in the device specified by (s) is output to the device specified by (d). [When the bit "n" is ON, the device specified by (d) is set to ON. When the bit "n" is OFF, the device specified by (d) is set to OFF.]

- When a constant (K) is specified as (s), it is automatically converted into the binary format.



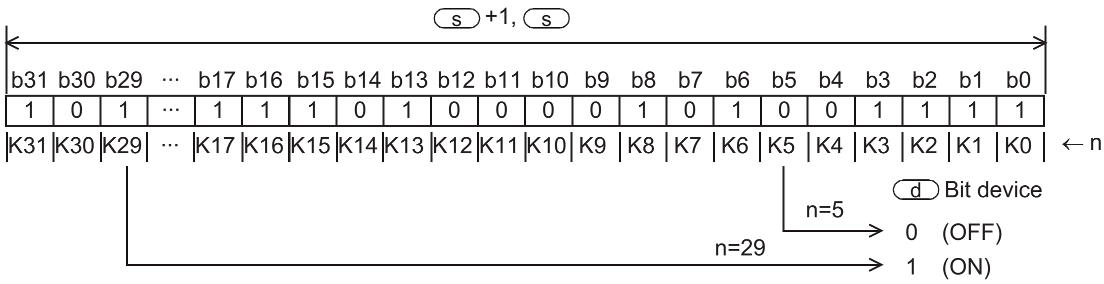


2. 32-bit operation(DBON, DBONP)

The status (ON or OFF) of the bit "n" in the device specified by (s) is output to the device specified by (d). [When the bit "n" is ON, the device specified by (d) is set to ON. When the bit "n" is OFF, the device specified by (d) is set to OFF.]

- When a constant (K) is specified as (s), it is automatically converted into the binary format.

*1 This defines the data that executes DBON or the device number storing the source data.



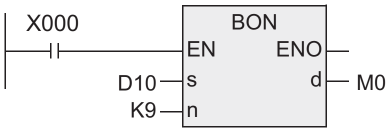
Cautions

1) Some restrictions to applicable devices
   ▲1:Applicable only to the FX3U and FX3UC PLCs. Not indexed (V, Z).
   ▲2:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲3:The FX3U and FX3UC PLCs only are applicable.
2) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.

Program examples

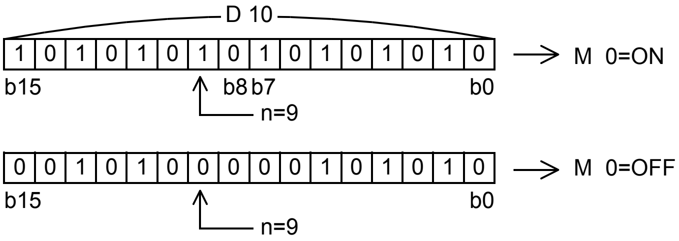
When the bit 9 (n=9) in D10 is "1" (ON), M0 is set to "1" (ON).

[Structured ladder]



[ST]

```
BON(X000, D10, K9, M0);
```



7.5.6 MEAN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction obtains the mean value of data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
MEAN	16 bits	Continuous	MEAN EN ENO s d n	MEAN(EN,s,n,d);
MEANP	16 bits	Pulse	MEANP EN ENO s d n	MEANP(EN,s,n,d);
DMEAN	32 bits	Continuous	DMEAN EN ENO s d n	DMEAN(EN,s,n,d);
DMEAN_P	32 bits	Pulse	DMEAN_P EN ENO s d n	DMEAN_P(EN,s,n,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Head word device storing data to be averaged	ANY16	ANY32
	n	Number of data to be averaged (n=1 to 64)	ANY16	
Output variable	ENO	Execution state	Bit	
	d	Word device storing the mean value result	ANY16	ANY32

3. Applicable devices

Operand type	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
	Bit Devices: System User							Word Devices: Digit Specification				System User				Special Unit	Index			Others: Constant		Real Number	Character String	Pointer
s								●	●	●	●	●	●	●	▲1	▲2			●					
d									●	●	●	●	●	●	▲1	▲2	●	●	●					
n														●	▲1					●	●			

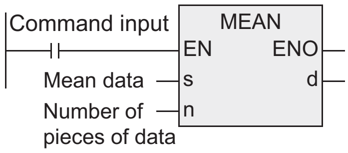
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (MEAN, MEANP)

The mean value of "n" 16-bit data from the device specified by (s) is stored to the device specified by (d).

- The sum is obtained as algebraic sum, and divided by "n".
- The remainder is ignored.

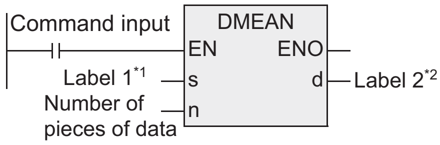


$$\frac{(s) + (s)+1 + \cdots + (s)+n-1}{n} \longrightarrow (d)$$

2. 32-bit operation (DMEAN, DMEANP)

The mean value of "n" 32-bit data from the device specified by (s) is stored to the device specified by (d).

- The sum is obtained as algebraic sum, and divided by "n".
- The remainder is ignored.



*1 This defines the head device number that stores the data to be averaged.
*2 This defines the device that stores the mean data.

$$\frac{[(s)+1, (s)] + [(s)+3, (s)+2] + \cdots + [\{(s)+n \times 2-1\}], [\{(s)+n \times 2-2\}]}{n} \longrightarrow [(d)+1, (d)]$$

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) The FXU PLC of V2.30 or earlier does not support 32-bit instructions.
3) When a device number is exceeded, "n" is handled as a smaller value in the possible range.
4) Some restrictions to applicable devices
   ▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2:The FX3U and FX3UC PLCs only are applicable.

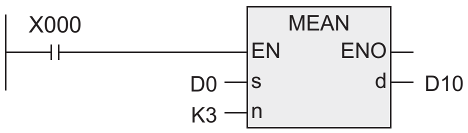
Error

When "n" is any value outside the range from "1" to "64", an operation error (M8067) is caused.

Program examples

The data of D0, D1 and D2 are summed, divided by "3", and then stored to D10.

[Structured ladder]



$$\frac{(D\ 0) + (D\ 1) + (D\ 2)}{3} \longrightarrow (D\ 10)$$

[ST]

```
MEAN(X000, D0, K3, D10);
```

7.5.7 ANS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction sets a state relay as an annunciator.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
ANS	16 bits	Continuous	ANS (EN, ENO, s, d, m)	ANS(EN,s,m,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Timer for evaluation time (100 ms timer)	ANY16
	(m)	Evaluation time m=1 to 32,767(unit: 100 ms)	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Annunciator device to be set	Bit

3. Applicable devices

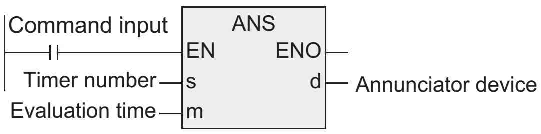
Operand type	Bit Devices: System User							Word Devices: Digit Specification				Word Devices: System User				Word Devices: Special Unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)												▲1							●					
(m)														●	▲3					●	●			
(d)						▲2													●					

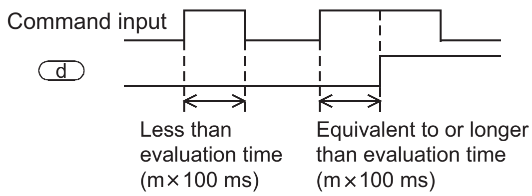
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation

When the command input remains ON for equivalent to or longer than the evaluation time [m × 100 ms, timer specified by (s)], the device specified by (d) is set.
When the command input remains ON for less than the evaluation time [m × 100 ms] and then turns OFF, the current value of the timer for evaluation specified by (s) is reset and the device specified by (d) is not set.
When the command input turns OFF, the timer for evaluation is reset.





Related device

Device	Name	Description
M8049	Enable annunciator	When M8049 is set to ON, M8048 and D8049 are valid.
M8048	Annunciator ON	When M8049 is ON and one of the state relays S900 to S999 is ON, M8048 turns ON.
D8049	Smallest state relay number in ON status	Among S900 to S999, the smallest state relay number in the ON status is stored.

Cautions

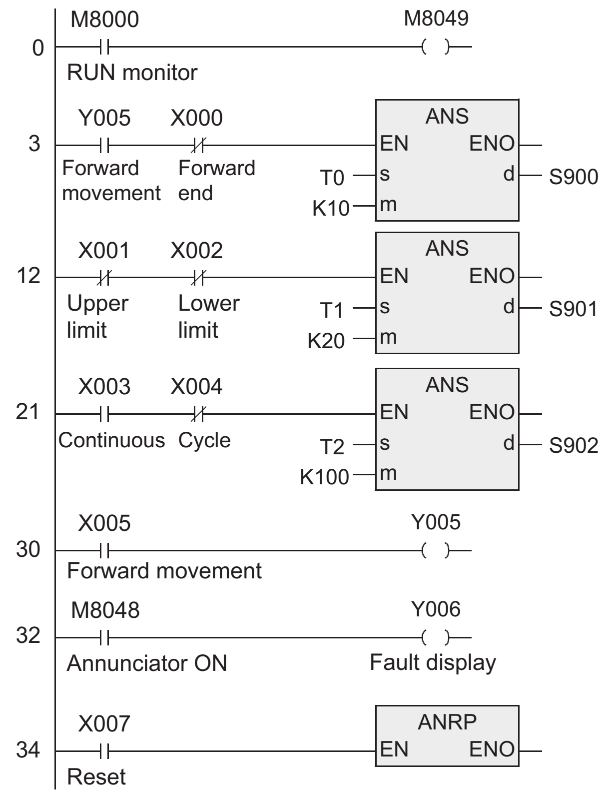
Some restrictions to applicable devices
▲1:T0 to T199
▲2:S900 to 999
▲3:The FX3U, FX3UC and FX3G PLCs only are applicable.

Program examples

1. Displaying a fault number using an annunciator

When the program for external fault diagnosis shown below is created and the content of D8049 (smallest state relay number in the ON status) is monitored, the smallest state relay number in the ON status from S900 to S999 is displayed.
If two or more faults are present at the same time, the next smallest fault number is displayed after the fault of the smallest fault number is cleared.

[Structured ladder]



When M8049 turns ON, monitoring becomes valid.

If the forward end detection input X000 does not turn ON within 1 second after the forward movement output Y005 is driven, S900 turns ON.

If both the upper limit input X001 and the lower limit input X002 are OFF for 2 seconds or more due to a DOG error, S901 turns ON.

The switch X004 is set to ON in one operation cycle of the machine. If the switch X004 is not set to ON while the continuous operation mode input X003 is ON in the machine whose tact time is less than 10 seconds, S902 turns ON.

When one among S900 to S999 turns ON, M8048 turns ON and the fault display output Y006 turns ON.

A state relay which was set to ON by the external fault diagnosis program is set to OFF by the reset button X007. Every time X007 is set to ON, an operation state relay in the ON status with the smallest device number is reset (set to OFF) in turn.

[ST]

```
M8049:= M8000;
ANS(Y005 AND NOT X000, T0, K10, S900);
ANS(NOT X001 AND NOT X002,T1, K20, S901);
ANS(X003 AND NOT X004, T2, K100, S902);
Y005:=X005;
Y006:=M8048;
ANRP(X007);
```

7.5.8 ANR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction resets an annunciator in the ON status with the smallest number.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ANR	16 bits	Continuous	[ANR: EN — ENO]	ANR(EN);
ANRP	32 bits	Pulse	[ANRP: EN — ENO]	ANRP(EN);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit

3. Applicable devices

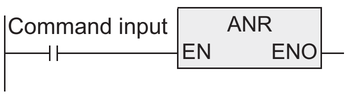
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
-	There are no applicable devices.																							

Function and operation explanation

1. 16-bit operation (ANR, ANRP)

When the command input turns ON, a state relay working as annunciator in the ON status is reset.

- If two or more state relays are ON, the state relay with the smallest number is reset.
  When the command input is set to ON again, the state relay with the next smallest number is reset among state relays working as annunciators in the ON status.



Related device

Device	Name	Description
M8049	Enable annunciator	When M8049 is set to ON, M8048 and D8049 are valid.
M8048	Annunciator ON	When M8049 is ON and one of the state relays S900 to S999 is ON, M8048 turns ON.
D8049	Smallest state relay number in ON status	Among S900 to S999, the smallest state relay number in the ON status is stored.

Cautions

1. Execution in each operation cycle

- When ANR instruction is used, annunciators in the ON status are reset in turn in each operation cycle.
- When ANRP instruction is used, an annunciator in the ON status is reset only in one operation cycle (only once).

Program examples

Refer to ANS instruction

→ **For a program example, refer to Section 7.5.7.**

7.5.9 SQR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This instruction obtains the square root.
The DESQR instruction obtains the square root in floating point operation.

→ **For DESQR instruction, refer to Section 7.12.15.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SQR	16 bits	Continuous	SQR [EN ENO, s d]	SQR(EN,s,d);
SQRP	16 bits	Pulse	SQRP [EN ENO, s d]	SQRP(EN,s,d);
DSQR	32 bits	Continuous	DSQR [EN ENO, s d]	DSQR(EN,s,d);
DSQRP	32 bits	Pulse	DSQRP [EN ENO, s d]	DSQRP(EN,s,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Word device storing data whose square root is obtained.	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Data register storing the square root operation result	ANY16	ANY32

3. Applicable devices

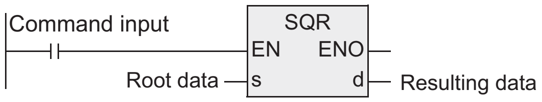
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	▲1	▲1			●	●	●			
(d)														●	▲1	▲1			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (SQR, SQRP)

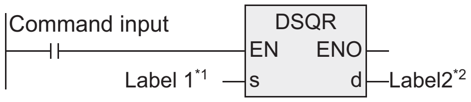
The square root of the data stored in the device specified by (s) is calculated, and stored to the device specified by (d).



$$\sqrt{(s)} \rightarrow (d)$$

2. 32-bit operation (DSQR, DSQRP)

The square root of the data stored in the device specified by (s) is calculated, and stored to the device specified by (d).



$$\sqrt{(s)+1, (s)} \rightarrow (d)+1, (d)$$

*1 This defines the device that stores the data whose square root is obtained.
*2 This defines the device that stores the square root operation result.

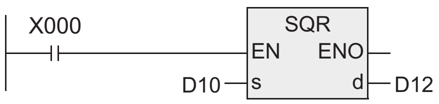
Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) The obtained square root is an integer because the decimal point is ignored.
   When the decimal point is ignored, M8021 (borrow flag) turns ON.
3) When the calculated value is true "0", M8020 (zero flag) turns ON.
4) Some restrictions to applicable devices
   ▲1:The FX3U and FX3UC PLCs only are applicable.

Program examples

The square root of D10 is stored to D12.
The value of D10 is "100".

[Structured ladder]



$$\sqrt{D\,10} \rightarrow D\,12 = \sqrt{100} \rightarrow 10$$

[ST]

```
SQR(X000, D10, D12);
```

7.5.10 FLT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	○	×	×	△	×	×

Outline

This instruction converts a binary integer into a binary floating point (real number).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
FLT	16 bits	Continuous	FLT EN ENO s d	FLT(EN,s,d);
FLTP	16 bits	Pulse	FLTP EN ENO s d	FLTP(EN,s,d);
DFLT	32 bits	Continuous	DFLT EN ENO s d	DFLT(EN,s,d);
DFLTP	32 bits	Pulse	DFLTP EN ENO s d	DFLTP(EN,s,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Data register storing binary integer	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Data register storing binary floating point (real number)	ANY_SIMPLE	

3. Applicable devices

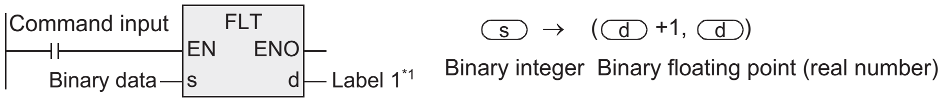
Operand type	Bit Devices							Word Devices												Others					
	System User							Digit Specification				System User				Special Unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	▲1	▲2			●					
(d)														●	▲1	▲2			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (FLT, FLTP)

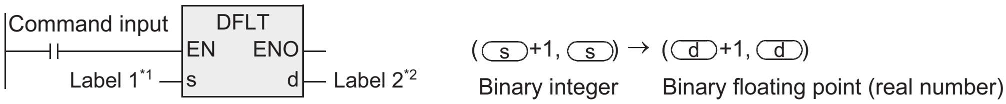
The binary integer data of the device specified by (s) is converted into binary floating point (real number), and stored to the device specified by (d).



*1 This defines the device that stores the binary floating point data.

2. 32-bit operation (DFLT, DFLTP)

The binary integer data of the device specified by (s) is converted into binary floating point (real number), and stored to the device specified by (d).



*1 This defines the device that stores the binary integer data.
*2 This defines the device that stores the binary floating point data.

Related instruction

Instruction	Description
INT	It is inverse of FLT instruction, and converts binary floating point into binary integer.

Related devices

→ For the method of the zero and borrow flags, refer to Section Section 1.3.4.

Device	Name	Description
M8020	Zero flag	Turns ON when the value is true "0".
M8021	Borrow flag	Turns ON when the floating point is rounded down..

Cautions

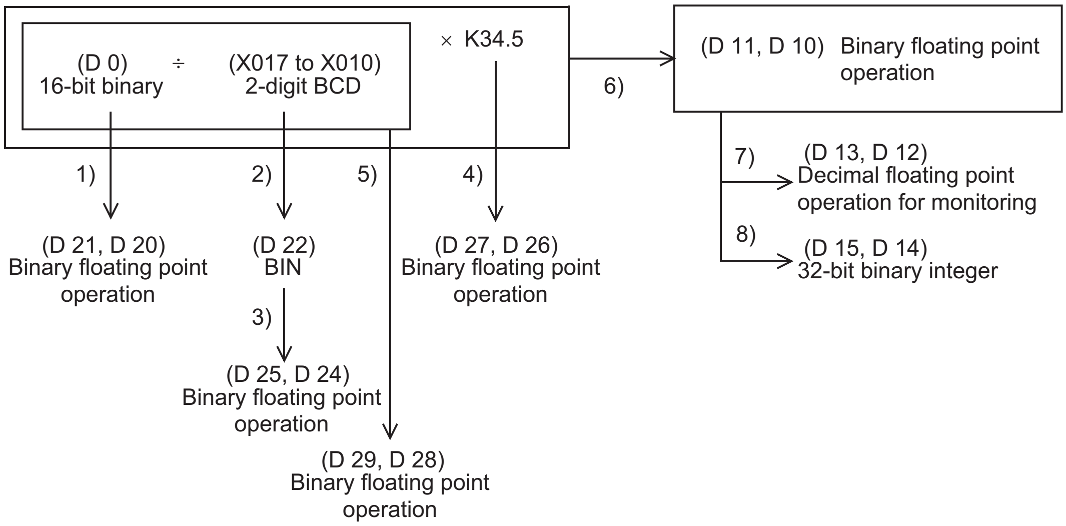
1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
2) The FX3G PLC of V1.10 or later supports the instruction.
3) The FXU PLC of V3.07 or later supports the instruction.
4) The value of K or H specified in each instruction for binary floating point (real number) operation is automatically converted into binary floating point (real number). It is not necessary to convert such a constant by FLT instruction.
(K and H cannot be specified in RAD, DEG, EXP and LOGE instructions.)
5) Some restrictions to applicable devices
▲1:The FX3U, FX3UC and FX3G PLCs only are applicable.
▲2:The FX3U and FX3UC PLCs only are applicable.

Program examples

1. Arithmetic operations by binary floating point operations

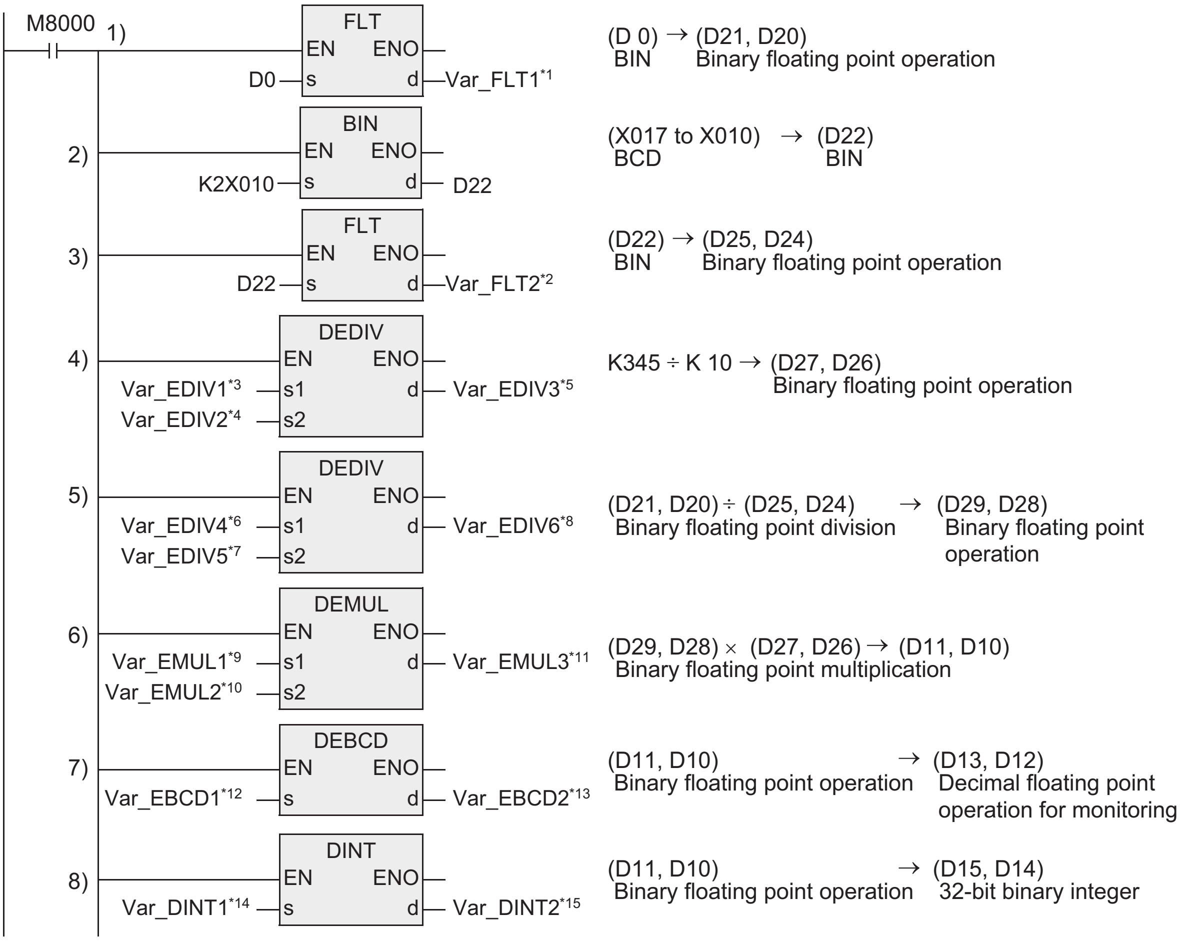
The sequence program shown below is constructed as follows:

1) Calculation example



2) Program

[Structured ladder]



[ST]

```
FLT(M8000, D0, Var_FLT1*1 );
BIN(M8000, K2X010, D22);
FLT(M8000, D22, Var_FLT2*2 );
DEDIV(M8000, Var_EDIV1*3 , Var_EDIV2*4 , Var_EDIV3*5 );
DEDIV(M8000, Var_EDIV4*6 , Var_EDIV5*7 , Var_EDIV6*8 );
DEMUL(M8000, Var_EMUL1*9 , Var_EMUL2*10 , Var_EMUL3*11 );
DEBCD(M8000, Var_EBCD1*12 , Var_EBCD2*13 );
DINT(M8000, Var_DINT1*14 , Var_DINT2*15 );
```

*1 Var_FLT1 is a global label and is defined as D20.
*2 Var_FLT2 is a global label and is defined as D24.
*3 Var_EDIV1 is a global label and is defined as K345.
*4 Var_EDIV2 is a global label and is defined as K10.
*5 Var_EDIV3 is a global label and is defined as D26.
*6 Var_EDIV4 is a global label and is defined as D20.
*7 Var_EDIV5 is a global label and is defined as D24.
*8 Var_EDIV6 is a global label and is defined as D28.
*9 Var_EMUL1 is a global label and is defined as D28.
*10 Var_EMUL2 is a global label and is defined as D26.
*11 Var_EMUL3 is a global label and is defined as D10.
*12 Var_EBCD1 is a global label and is defined as D10.
*13 Var_EBCD2 is a global label and is defined as D12.
*14 Var_DINT1 is a global label and is defined as D10.
*15 Var_DINT2 is a global label and is defined as D14.

7.6 High Speed Processing

7.6.1 REF

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction immediately outputs the latest input (X) information or the current output (Y) operation result in the middle of a sequence program operation.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
REF	16 bits	Continuous	REF EN ENO d n	REF(EN, d, n);
REFP	16 bits	Pulse	REFP EN ENO d n	REFP(EN, d, n);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	d	Bit device (X or Y) to be refreshed	Bit
	n	Number of bit devices to be refreshed (multiple of 8 in the range from 8 to 256)	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices													Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
d	▲1	▲2																						
n																				▲3	▲3			

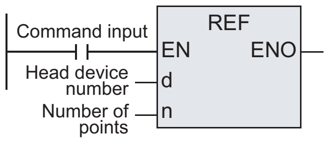
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(REF, REFP)

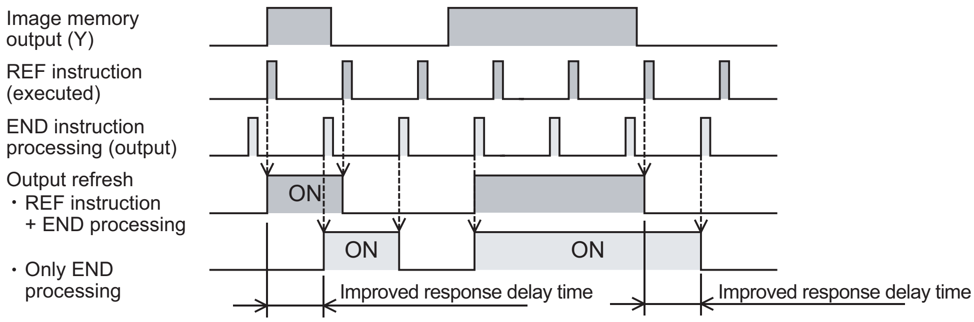
1) When refreshing outputs (Y)

"n" points are refreshed from the output of the device specified by (d). ("n" must be a multiple of 8.)



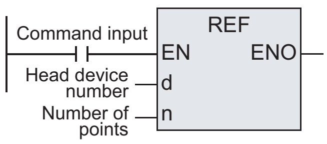
* Refer to "Caution" for the head device number and the number of points.

- When this instruction is executed, the output latch memory is refreshed to the output status in the specified range.



2) When refreshing inputs (X)

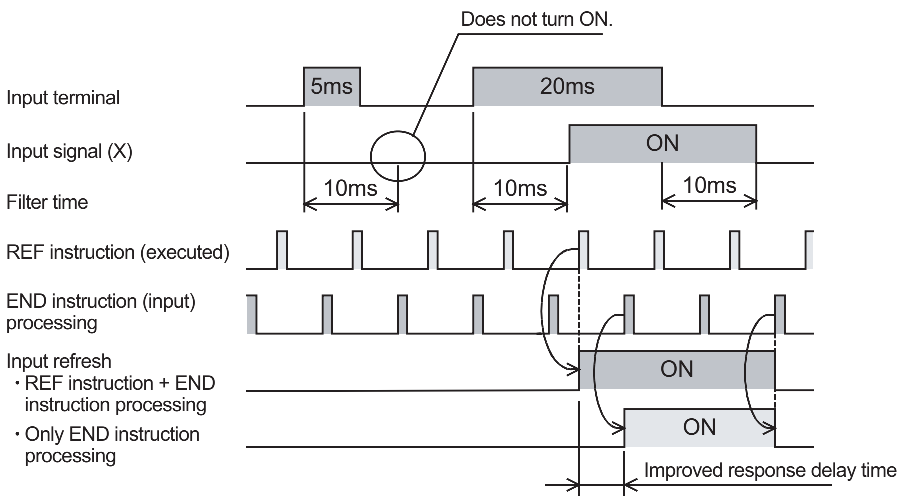
"n" points are refreshed from the input of the device specified by (d). ("n" must be a multiple of 8.)



* Refer to "Caution" for the head device number and the number of points.

- If the input information is turned ON approximately 10 ms (response delay time of the input filter) before the instruction is executed, the input image memory turns ON when the instruction is executed.
- The response delay time of the input filter can be changed.

→ **For details, refer to "what should be understood before using the REF instruction" described later.**



Cautions

1) When setting the specified head device number "d", make sure that the least significant digit number is "0" such as "X000, X010, X020, ..." or "Y000, Y010, Y020, ...".
2) The FX0, FX0S or FX0N PLC does not support the instructions of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
3) Some restrictions to applicable devices
   ▲1: X000, X010, X020 ....Up to the final input number (The last digit number must be "0".)
   ▲2: Y000, Y010, Y020 ....Up to the final output number (The last digit number must be "0".)
   ▲3: Set a multiple of "8" to the number of refresh points "n", such as K8(H8), K16(H10), ..., K256 (H100).
   Any number other than the above generates an error.

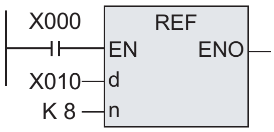
PLC	n
FX3U, FX3UC, FX3G	K8(H8), K16(H10), ..., K256(H100)
FX1S, FX1N, FX2N, FX1NC, FX2NC	K8(H8), K16(H10), ..., K256(H100)
FX0, FX0S	K8(H8) or K16(H10)
FX0N	K8(H8), K16(H10), ..., K128(H80)
FXU, FX2C	K8(H8), K16(H10), ..., K256(H100)

Program examples

1. When refreshing inputs

Only X010 to X017 (8 points in total) are refreshed.

[Structured ladder] [ST]

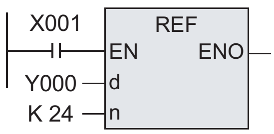


REF(X000,X010,K8);

2. When refreshing outputs

Y000 to Y007, Y010 to Y017 and Y020 to Y027 (24 points in total) are refreshed.

[Structured ladder] [ST]



REF(X001,Y000,K24);

What should be understood before using the REF instruction

1. Changing the input filter

The input filter value is determined by the contents of D8020 (initial value: 10 ms).
Use the MOV instruction, etc. to adjust the value in D8020, which represents the input filter value.

→ **For details, refer to "FX Structured Programming Manual (Device & Common)."**

2. Output response time

After the REF instruction is executed, the output (Y) sets the output signal to ON after the response time shown below.

→ **For details, refer to the respective PLC Hardware Edition manuals.**

1) Relay output type
   The output contact is activated after the response time of the output relay.
   - Y000 and higher: Approximately 10 ms
2) Transistor output type
   a) For FX3U and FX3UC (D, DSS) PLCs
      - Y000, Y001, Y002: 5 μs or less (load current = 10 mA or more, 5 to 24 V DC)
      - Y003 and higher: 0.2 ms or less (load current = 100 mA, 24 V DC)
   b) For FX3UC-32MT-LT (-2) PLC
      - Y000, Y001, Y002, Y003: 5 μs or less (load current = 10 mA or more, 5 to 24 V DC)
      - Y004 and higher: 0.2 ms or less (load current = 100 mA, 24 V DC)
   c) For FX3G PLC (14-point, 24-point types)
      - Y000, Y001: 5 μs or less (load current = 10 mA or more, 5 to 24 V DC)
      - Y002 and higher: 0.2 ms or less (load current = 100 mA, 24 V DC)
   d) For FX3G PLC (40-point, 60-point types)
      - Y000, Y001, Y002: 5 μs or less (load current = 10 mA or more, 5 to 24 V DC)
      - Y003 and higher: 0.2 ms or less (load current = 100 mA, 24 V DC)
   e) For FX1S, FX1N, FX2N, FX1NC and FX2NC PLCs
      - Y000, Y001: 15 μs to 30 μs or less
      - Y002 and higher: 0.2 ms or less
   f) For FX0, FX0S, FX0N, FXU and FX2C PLCs
      - Y000 and higher: 0.2 ms or less

3. When using the REF instruction between FOR and NEXT instructions or between a label (with a lower step number) and CJ instruction (with a higher step number)

Inputs and outputs can be refreshed in a routine program even when the input information or immediate output is required in the middle of a routine program during control.

4. When using the input interrupt (I) function

When executing interrupt processing accompanied by I/O operations, I/O refresh can be executed in the interrupt routine to receive the latest input (X) information and give the immediate output (Y) of the operation result so that dispersion caused by the operation time is improved.

7.6.2 REFF

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

The digital input filter time of the inputs can be changed using this instruction or D8020.
Using this instruction, the status of inputs can be refreshed at an arbitrary step in the program for the specified input filter time, and then transferred to the image memory.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
REF	16 bits	Continuous	REFF EN ENO n	REF(EN, n);
REFP	16 bits	Pulse	REFFP EN ENO n	REFP(EN, n);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(n)	Digital input filter time (1 ms increment)	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(n)														●	▲1					▲2	▲2			

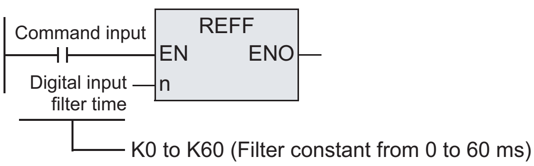
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(REFF, REFFP)

The image inputs below are refreshed at the digital input filter time [n × 1 ms].

PLC	Input
FX3U, FX3UC	X000 to X017 X000 to X007 in the FX3U-16M□, and FX3UC-16M□
FX2N, FX2NC	X000 to X017 X000 to X007 in the 16-input basic units
FXU, FX2C	X000 to X007



- When the input turns ON "n × 1 ms" before the instruction is executed, the input image memory is set to ON. When the input turns OFF "n × 1 ms" before the instruction is executed, the input image memory is set to OFF.
- When the command input is ON, the REFF instruction is executed in each operation cycle.
- When the command input is OFF, the REFF instruction is not executed, and the input filter uses the set value of D8020 (which is the value used during input processing).

Cautions

1. Function of the input filter

The filter time of the digital filter can be changed in 1 ms units within the range from 0 to 60 ms using instructions. When the filter time is set to "0", the input filter value is as follows.

1) For FX3U and FX3UC PLCs

Input number	Input filter value when set to "0"
X000 to X005	5μs*2
X006, X007	50μs
X010 to X017*3	200μs*3

*1. X000 to X007 in the FX3U-16M□, and FX3UC-16M□

*2. When setting the input filter time to "5 μs", perform the following actions.
- Make sure that the wiring length is 5 m or less.
- Connect a bleeder resistor of 1.5 kΩ (1 W or more) to the input terminal, and make sure that the load current in the open collector transistor output of the external equipment is 20 mA or more including the input current of the main unit.

*3. The filter time is fixed to 10 ms in X010 to X017 when the FX3U-16M□ or FX3UC-16M□ is used.

2) For FX2N and FX2NC PLCs

Input number	Input filter value when set to "0"
X000, X001	20μs
X002 to X017	50μs

3) For FXU and FX2C PLCs

Input number	Input filter value when set to "0"
X000 to X007	50μs

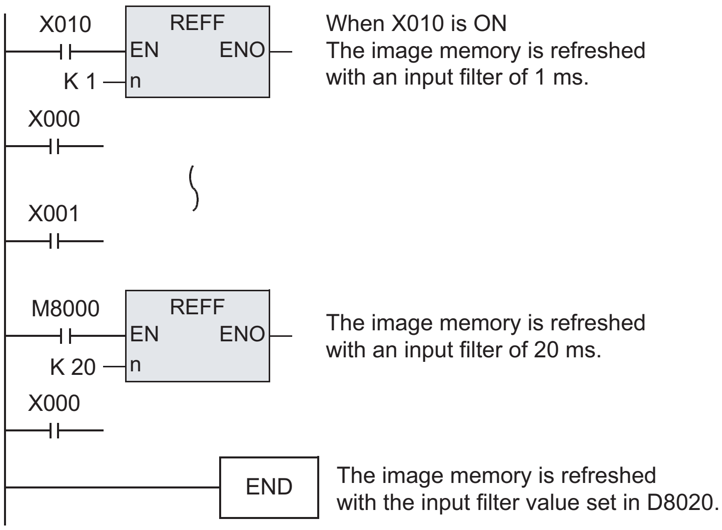
2. Some restrictions to applicable devices

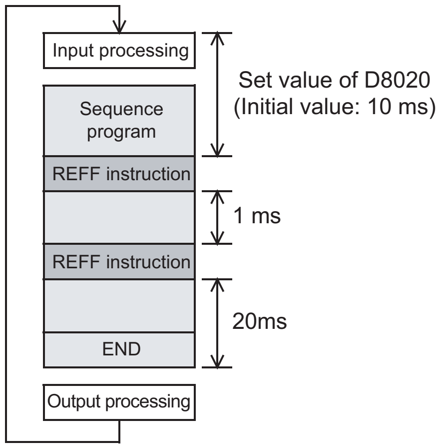
▲1: Applicable to the FX3U and FX3UC PLCs only.
▲2: Set the filter time within the range of K0(H0) to K60(H3C) [0 to 60 ms].

Program examples

1. Relationship between the program and the filter time

[Structured ladder]





[ST]

```
REFF(X010,K1);

    ⁀

REFF(M8000,K20);

    ⁀

END
```

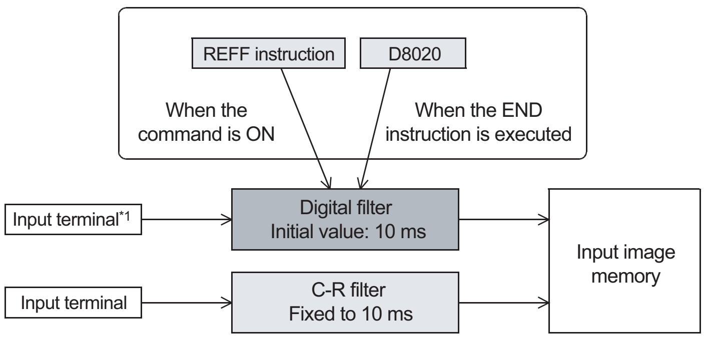
What should be understood before using REFF instruction

Generally, a C-R filter of approximately 10 ms is provided for inputs in PLCs as countermeasures against chattering and noise at the input contacts.
A digital filter is provided for some inputs (*1). The digital filter value can be changed within the range from 0 to 60 ms using instructions.

1. How to change the digital filter (executing END instruction)

The digital filter initial value (10 ms) is set in special data register D8020.

By changing this value using the MOV instruction, etc., the input filter value for X000 to X017[*2] which is used during execution of the END instruction can be changed.



*1. Where a digital filter is used on an input terminal, refer to the descriptions on the functions and operations.

2. Instruction in which the digital filter is automatically changed

Regardless of the change in the filter time executed by the REFF instruction, when the following functions and instructions are executed, the input filter value is automatically changed (as shown in the caution).
However, if the digital filter is used in any other functions or instructions than the ones listed, the digital filter uses the time set in D8020. As a result, the program will not run correctly if the ON or OFF duration of the corresponding input signal is less than the input filter time.

- Input of interrupt pointer specified in the input interrupt function
- Input used in a high speed counter
- Input used in the SPD instruction

7.6.3 MTR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This instruction reads matrix input as 8-point input × "n" output (transistor) in the time division method.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
MTR	16 bits	Continuous	MTR block: EN, s, n → ENO, d1, d2	MTR(EN, s, n, d1, d2);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device (X) number of matrix signal input X000, X010, X020, ..., final input X number (Only "0" is allowed in the least significant digit of device numbers)	Bit
	n	Number of columns in matrix input (K2 to K8/H2 to H8)	ANY16
Output variable	ENO	Execution state	Bit
	d1	Head device (Y) number of matrix signal output Y000, Y010, Y020, ..., final output Y number (Only "0" is allowed in the least significant digit of device numbers.)	Bit
	d2	Head bit device (Y, M or S) number of ON output destination Y000, Y010, Y020, ..., final Y number, M000, M010, M020, ... , final M number or S000, S010, S020, ..., final S number (Only "0" is allowed in the least significant digit of device numbers.)	Bit

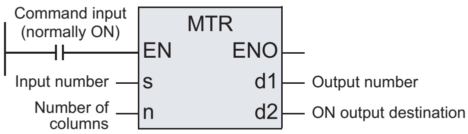
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●																							
n																				●	●			
d1		●																						
d2		●	●			●																		

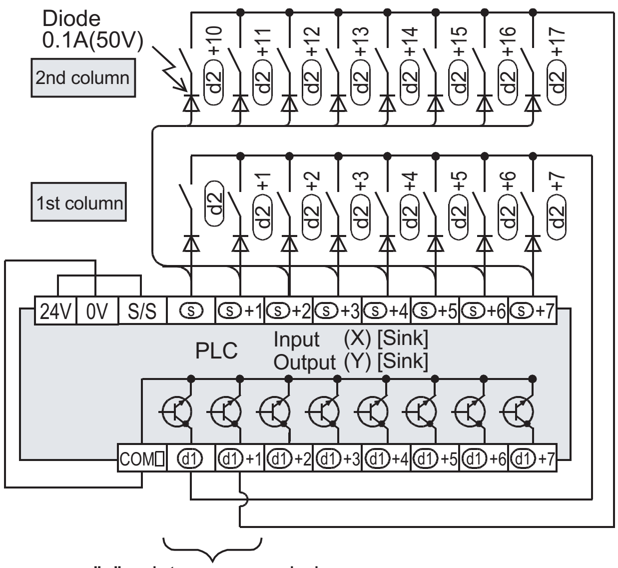
Function and operation explanation

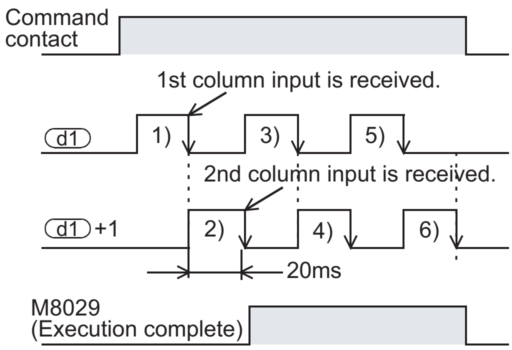
1. 16-bit operation(MTR)

An input signal of 8 points × "n" columns is controlled in the time division method using 8 inputs of the device specified by (s) and "n" transistor outputs of the device specified by (d1). Each column is read in turn, and then output to the device specified by (d2).



For each output, the I/O processing is executed immediately in turn in interrupt at every 20 ms under consideration of the input filter response delay of 10 ms.
The figure below shows an example of the FX3U series main unit (sink input / sink output). For the wiring, refer to the manual of the PLC used.





Related devices

Device	Name	Description
M8029	Instruction execution complete	Turns ON after the first cycle operation

Cautions

1. Number of occupied devices

1) Eight input points are occupied from the input device number specified in (s).

2) "n" output points are occupied from the output device number specified in (d1).
When specifying the output in (d2), make sure that "n" output numbers specified in (d1) does not overlap the output specified in (d2).

2. Wiring

One diode of 0.1A/50V is required for each switch.

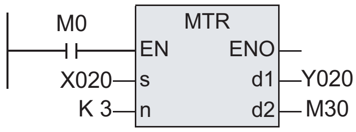
3. Output format

Use the transistor output format.

Program examples

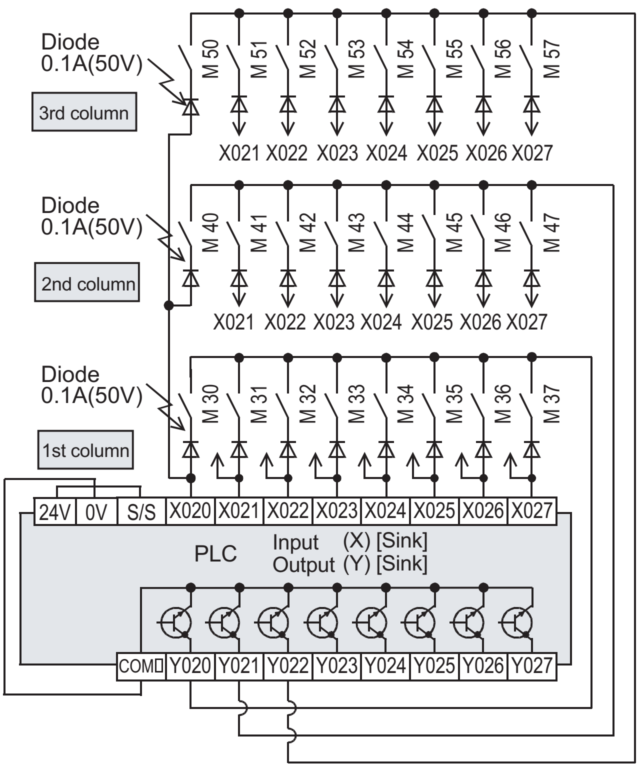
n = Three outputs (Y020, Y021 and Y022) are set to ON in turn repeatedly.
Every time an output is set to ON, eight inputs in the 1st, 2nd and 3rd columns are received in turn repeatedly, and stored to M30 to M37, M40 to M47, and M50 to M57 respectively.
In this program example, the FX3U series main unit (sink input / sink output) is used. For the wiring, refer to the manual of the PLC used.

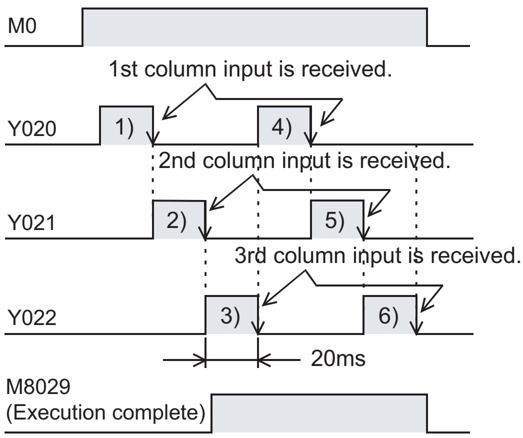
[Structured ladder]



[ST]

```
MTR(M0,X020,K3,Y020,M30);
```

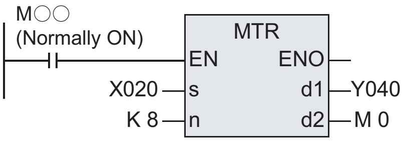




Operation and cautions for MTR instruction

1. Command input

1) Setting the command input to normally ON
For the MTR instruction, set the command input to normally ON.



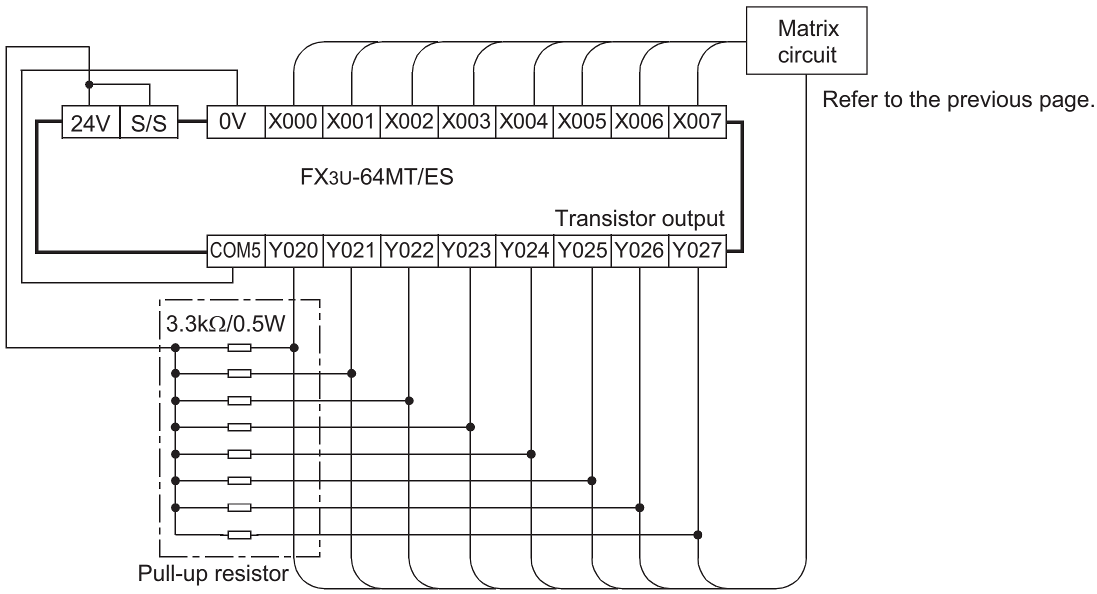
2. Input numbers used in MTR instruction

1) Inputs available in MTR instruction
Use inputs X020 and later under normal conditions.
(X010 and later for 16-point type main unit)

2) When using the inputs X000 to X017 (X000 to X007 for 16-point type main unit)
The receiving speed is higher. Because the output transistor recovery time is long and the input sensitivity is high, however, erroneous input pulses may be counted.
To prevent erroneous input pulses, connect pull-up resistors (3.3 kΩ / 0.5W) to transistor outputs used in MTR instruction.
For pull-up resistors, use the power supply shown in the table below.

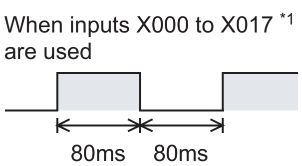
	Power supply used for pull-up resistors
AC power type PLC	Service power supply
DC power type PLC	Power supply for driving PLC

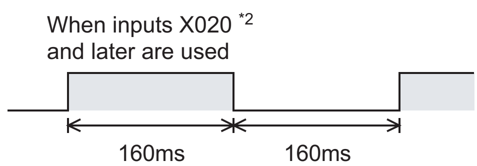
The figure below shows an example of the FX3U series main unit (sink input / sink output).



3. ON/OFF duration of input signals

Because 64 input points (8 rows × 8 columns) are received in a cycle of 80 or 160 ms, the ON/OFF duration of each input signal should be greater than or equal to the value shown below.





*1. X000 to X007 for 16-point type main unit

*2. X010 and later for 16-point type main unit

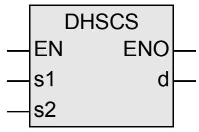
7.6.4 DHSCS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction compares a value counted by a high speed counter with a specified value at each count, and immediately sets an external output (Y) if the two values are equivalent each other.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DHSCS	32 bits	Continuous		DHSCS(EN, s1, s2, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s1)	Data to be compared with the current value of a high-speed counter or word device storing the data to be compared	ANY32
	(s2)	Device of a high speed counter	ANY32
Output variable	EN0	Execution state	Bit
	(d)	Bit device to be set to ON when the compared two values are equivalent to each other	Bit

3. Applicable devices

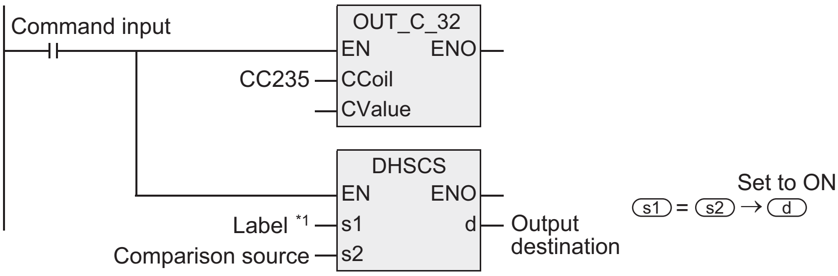
Operand type	Bit Devices							Word Devices												Others					
	System User							Digit Specification				System User				Special Unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲3	▲4		●	●	●	●			
(s2)													●						●					
(d)		●	●			●	▲1												●					▲2

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DHSCS)

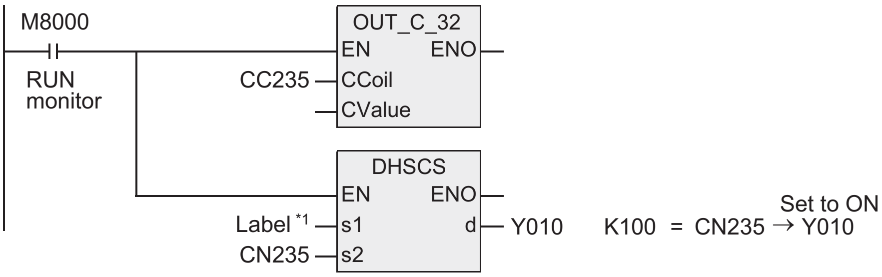
When the current value of a high speed counter of the device specified by s2 becomes the comparison value of the device specified by s1 (for example, when the current value changes from "199" to "200" or from "201" to "200" if the comparison value is K200), the bit device specified by d is set to ON without regard to the operation cycle. This instruction is executed after the counting processing in the high speed counter.



*1. This defines the comparison value.

Operation

When the current value of the high speed counter C235 changes from "99" to "100" or from "101" to "100", Y010 is set to ON (output refresh).



*1. This defines K100.

Related instruction

The following instructions can be combined with high speed counters.

Instruction	Instruction name
DHSCS	High speed counter set
DHSCR	High speed counter reset
DHSZ	High speed counter zone compare
DHCMOV	High speed counter move
DHSCT	High speed counter compare with data table

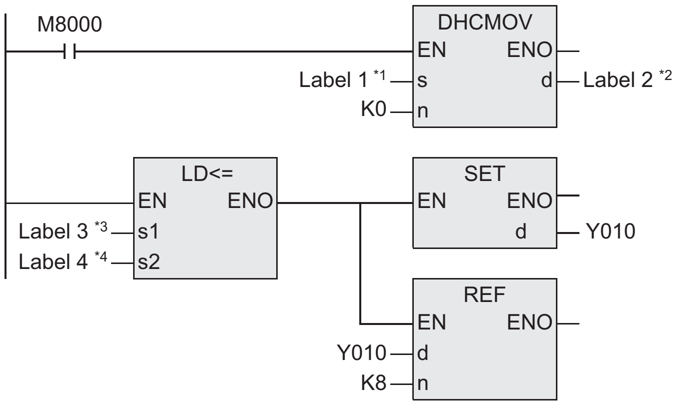
Cautions

1. Selection of the counter comparison method

When the DHSCS instruction is used, the maximum frequency and total frequency of the high speed counter are affected.
Refer to the counting operation described below, and select according to the contents of control whether to use this instruction or general-purpose comparison instruction.

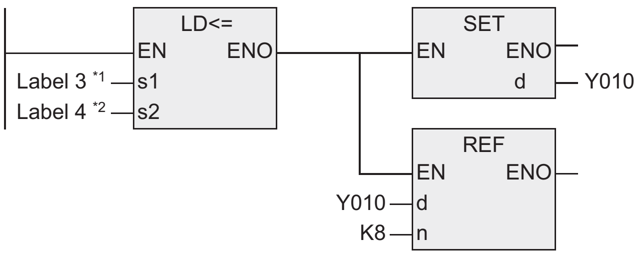
1) Case to select DHSCS instruction
   - When the output should be given when the counting result becomes equivalent to the comparison value without regard to the scan time of the PLC.
2) Cases to select a general-purpose comparison instruction
   - When the required frequency is beyond the counting performance.
   - When counting is regarded as important, but the effect of the scan time can be ignored in operations according to the counting result.
   - When the number of an instruction exceeds the allowable limit.

For FX3U and FX3UC PLCs



*1. This defines CN251.
*2. This defines D100.
*3. This defines K100.
*4. This defines D100.

For FX0, FX0S, FX0N, FXU, FX2C, FX1S, FX1N, FX1NC, FX2N, FX2NC and FX3G PLCs



*1. This defines K100.
*2. This defines CN251.

2. Device specification range

Only high speed counters can be specified as (s2).
For details, refer to the following manual.

→ **FX Structured Programming Manual (Device & Common)**

3. Some restrictions to applicable devices

▲1: Applicable to the FX3U and FX3UC PLCs only.
Not indexed (V, Z).
▲2: Use interrupt pointer when using counter interrupt.
(Not available for the FX0, FX0S, FX0N, FX1N, FX1S or FX3G PLC.)

→ **For the counter interrupt using this instruction, refer to Section 8.6.**

▲3: Applicable to the FX3U, FX3UC and FX3G PLCs only.
▲4: Applicable to the FX3U and FX3UC PLCs only.

4. Specifying input and output variables

When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

5. Precedence of DHSCS, DHSCR and DHSZ instructions to one particular high speed counter

→ **Refer to caution 6 in "Common cautions on using instructions for high speed counter" which is described later.**

6. Reset operation by an external terminal

→ **Refer to caution 5 in "Common cautions on using instructions for high speed counter" which is described later.**

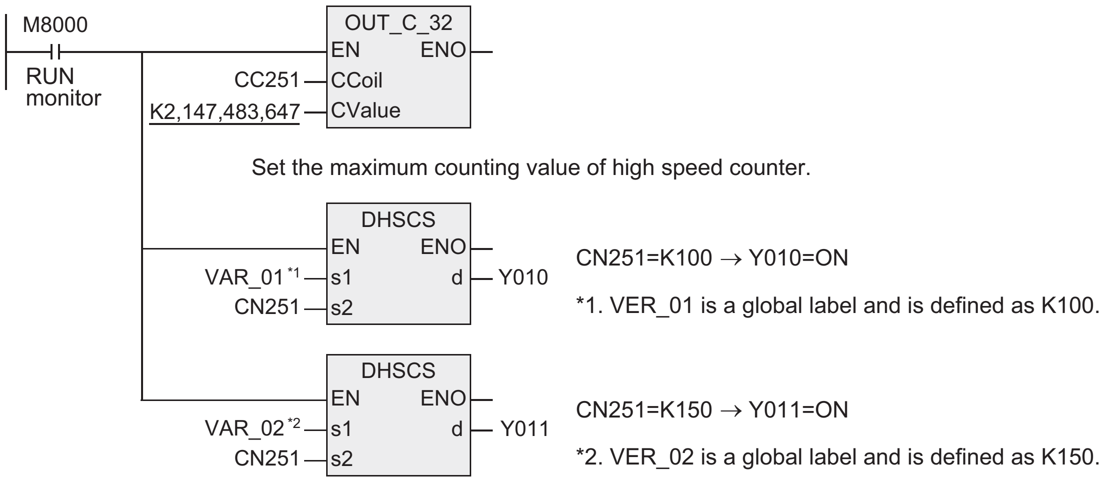
7. Other cautions on use

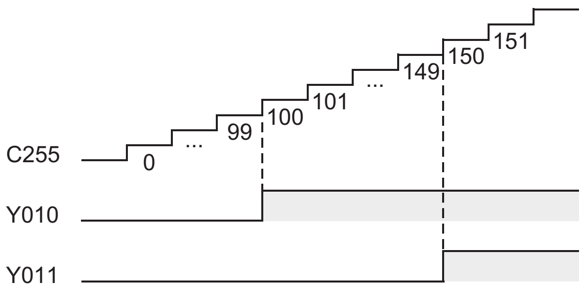
→ **Refer to caution in "Common cautions on using instructions for high speed counter" which is described later.**

Program examples

With regard to the current value of a counter, different outputs (Y) are arbitrarily set to ON by two values.

[Structured ladder]





[ST]

```
OUT_C_32(M8000,CC251,K2147483647);
DHSCS(M8000,VAR_01,CC251,Y010);
DHSCS(M8000,VAR_02,CC251,Y011);
```

Common cautions on using instructions for high speed counter

DHSCS, DHSCR and DHSZ instructions are provided for high speed counters. This section explains common cautions for these instructions.

1. Limitation in the number of an instruction in a program

DHSCS, DHSCR and DHSZ instructions can be used as many times as necessary in the same way as general instructions. However, the number of simultaneously driven instructions is limited.

→ **Refer to Section 1.3.7.**

2. Response frequency of high speed counters

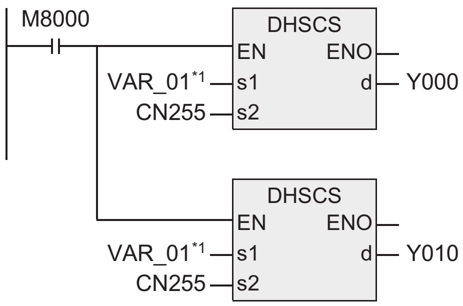
When DHSZ instruction is used, the maximum response frequency of every software counter and the total frequency are limited. For details, refer to the following manual.

→ **FX Structured Programming Manual (Device & Common)**

3. Specification of output numbers (Y)

When using the same instruction for high speed counter repeatedly or when driving two or more instructions for high speed counter at the same time, specify such output devices (Y) whose high-order two digits are the same (in units of 8 devices).

1) When using devices of the same number (in units of 8 devices)
   Example: when using Y000, specify Y000 to Y007. When using Y010, specify Y010 to Y017.
2) When using two or more instructions for high speed counter and non-consecutive output (Y) numbers
   A program example is shown below:



*1. VER_01 is a global label and is defined as K100.

When C255 reaches K100, the output Y000 is driven by interrupt. Y010 is driven when END processing is executed.
If interrupt drive is required, use an output number in the range from Y001 to Y007 whose high-order two digits are equivalent.

4. Caution on the counting operation when the current value is changed

An instruction for the high speed counter gives the comparison result when a pulse is input to the input (X) of the high speed counter.
However, the comparison result is not given when the current value of the high speed counter is changed in the following method.

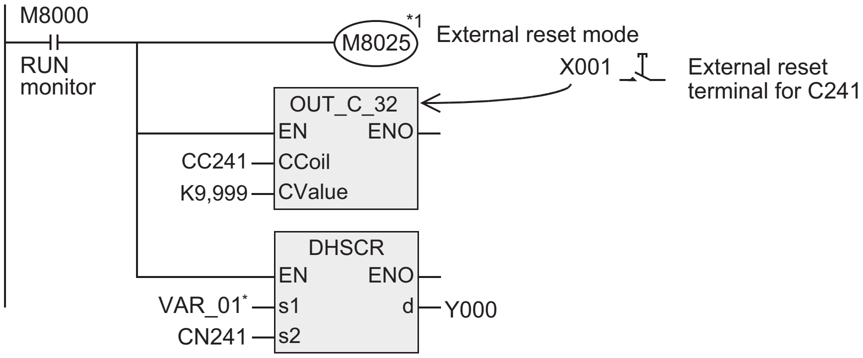
1) Change method (example)
   a) Overwriting the contents of a word device used as the comparison value using DMOV instruction, etc.
   b) Resetting the current value of a high speed counter in a program.
2) Operation
   Even if the condition for setting the output to ON or OFF is given as the comparison result, the comparison result does not change when an instruction is simply driven.

5. Reset operation by an external terminal [M8025*1: DHSC (external reset) mode]

For a high speed counter equipped with an external reset terminal (R) such as C241, an instruction is executed and the comparison result is output at the rising edge of the reset input signal.
(The FXU PLC of V2.1 or later and produced February 1990 or later are compatible with this function. The FX0, FX0S, FX0N, FX1S, FX1N, FX1NC or FX3G is not compatible.)

1) Program

If an instruction for the high speed counter is used while M8025*1 is driven, the instruction is executed again when the current value of the high speed counter C241 is cleared by an external reset terminal. And the comparison result is output even if a counting input is not given.



* VER_01 is a global label and is defined as K100.

*1. It is not necessary to drive M8025 for the FX0, FX0S, FX0N, FX1S, FX1N, FX1NC and FX3G PLCs.
The above reset operation takes place as a basic function.

2) Operation

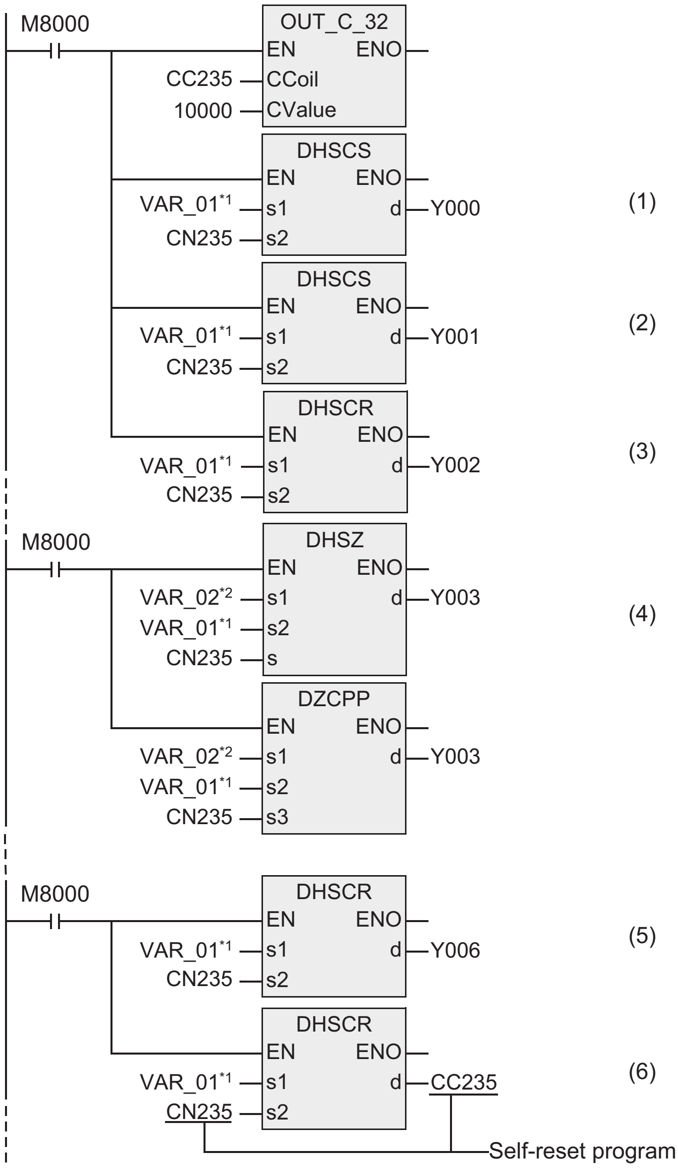
When the external reset input X001 turns ON while the current value of C241 is "100", for example, the current value of C241 is reset to "0". And Y000 is reset at this time even if a counting input is not given.

6. Priority order in operations among DHSCS, DHSCR and DHSZ instructions for the same high speed counter when the same comparison value is used.

1) FX3U and FX3UC PLCs
When the same comparison value is used for the same high speed counter in DHSCS, DHSCR and DHSZ instructions, high speed counter reset (self-reset) by DHSCR instruction is executed with the highest priority (as shown in the table below).
In this case, the comparison results do not change in DHSCS, DHSCR and DHSZ instructions whose comparison value is programmed to be the same as the comparison value for self-reset by DHSCR instruction. To change the comparison results, set the comparison value to "K0".

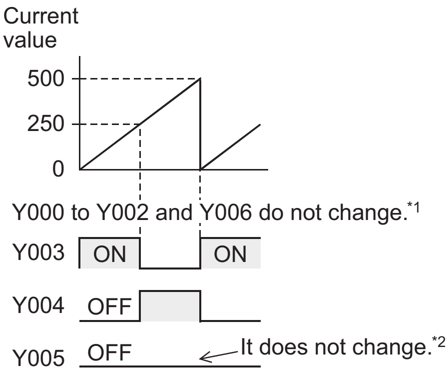
2) FX1S, FX1N, FX1NC, FX2N, FX2NC and FX3G PLCs
Comparison is executed in the programmed sequence without regard to the instructions.

Program sequence	Processing sequence		
	FX3U•FX3UC	FX3G	FX1N•FX1S•FX1NC•FX2N•FX2NC
DHSCS (1)	DHSCR (6) (Self-reset)	DHSCS (1)	DHSCS (1)
DHSCS (2)	DHSZ (4)	DHSCS (2)	DHSCS (2)
DHSCR (3)	DHSCS (1)	DHSCR (3)	DHSCR (3)
DHSZ (4)	DHSCS (2)	DHSZ (4)	(Not supported)
DHSCR (5)	DHSCR (3)	DHSCR (5)	DHSCR (5)
DHSCR (6) (Self-reset)	DHSCR (5)	DHSCR (6) (Self-reset)	DHSCR (6) (Self-reset)



*1. VER_01 is a global label and is defined as K500.
*2. VER_02 is a global label and is defined as K250.

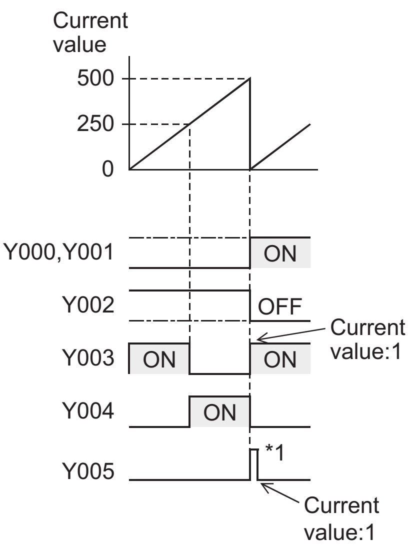
Operation of FX3U and FX3UC PLCs



*1. To change the comparison results by the instructions (1) to (3) and (5) in the previous page, change the comparison value "K500" in the instructions (1) to (3) and (5) in the previous page to "K0".

*2. To set Y005 to ON in the DHSZ instruction (4) in the previous page, set a value smaller than the comparison value "K500". However, due to the response delay at the output, the output may not operate within the short time before the counter's current value is reset to "0" (to K500 (K0)).

Operation of FX3G, FX2N, FX2NC, FX1N, FX1NC and FX1S PLCs.



*1. Due to the response delay at the output, the output may not operate within the short time before the counter's current value is reset from "0" to "1".

7.6.5 DHSCR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction compares the value counted by a high speed counter with a specified value at each count, and immediately resets an external output (Y) when both values become equivalent to each other.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DHSCR	32 bits	Continuous	DHSCR —EN ENO— —s1 d— —s2	DHSCR(EN,s1,s2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Data to be compared with the current value of a high-speed counter or word device storing the data to be compared*1	ANY32
	s2	Device of a high speed counter	ANY32
Output variable	EN0	Execution state	Bit
	d	Bit device to be set to ON when the compared two values are equivalent to each other	Bit

3. Applicable devices

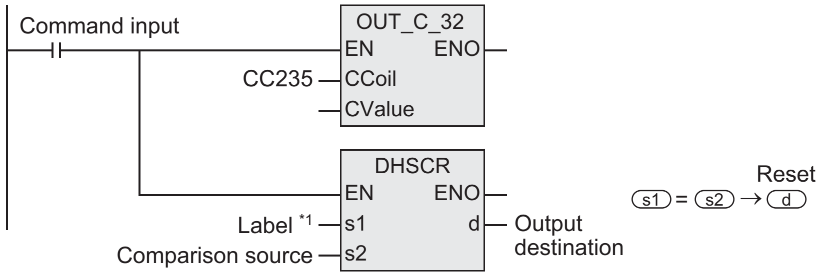
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲2	▲3		●	●	●	●			
s2													●						●					
d		●	●			●	▲1						▲4						●					

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DHSCR)

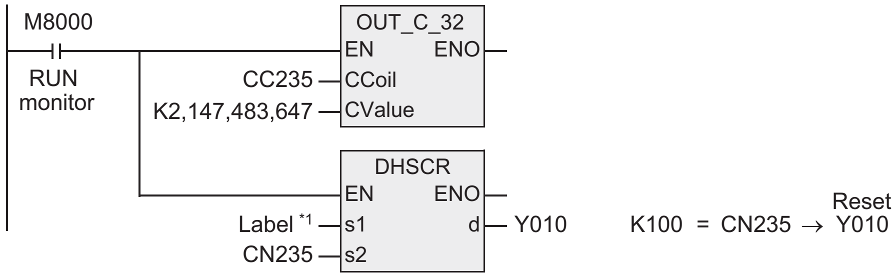
When the current value of the high speed counter of the device specified by (s2) becomes the comparison value of the device specified by (s1) (for example, when the current value changes from "199" to "200" or from "201" to "200" if the comparison value is K200), the bit device specified by (d) is reset (set to OFF) regardless of the operation cycle. In this instruction, the comparison processing is executed after the counting processing in the high speed counter.



*1. This defines the comparison value.

Operation

When the current value of the high speed counter C255 changes (counts) from "99" to "100" or from "101" to "100", Y010 is reset (output refresh).



*1. This defines K100.

Related instruction

The following instructions can be combined with high speed counters.

Instruction	Instruction name
DHSCS	High speed counter set
DHSCR	High speed counter reset
DHSZ	High speed counter zone compare
DHCMOV	High speed counter move
DHSCT	High speed counter compare with data table

Cautions

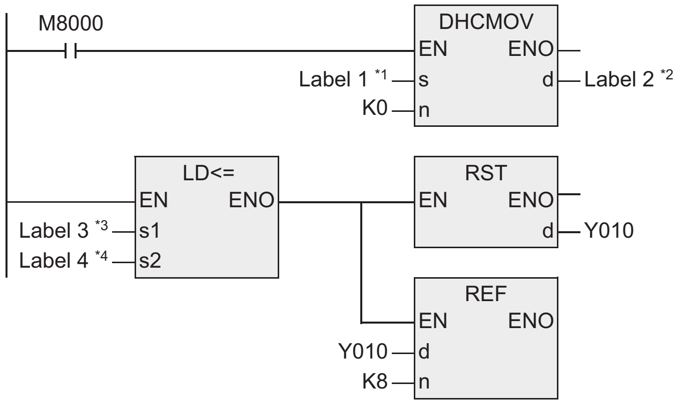
1. Selection of the counter comparison method

When the DHSCS instruction is used, the maximum frequency and total frequency of the high speed counter are affected.
Refer to the counting operation described below, and select according to the contents of control whether to use this instruction or general-purpose comparison instruction.

1) Case to select DHSCR instruction
   - When the output should be given when the counting result becomes equivalent to the comparison value without regard to the scan time of the PLC.

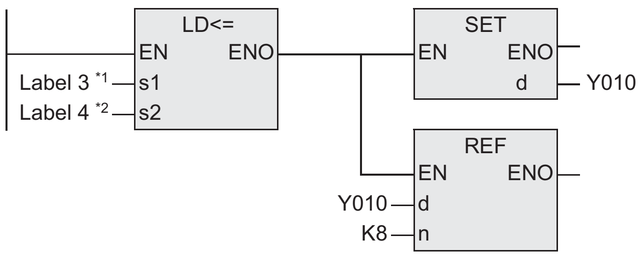
2) Cases to select a general-purpose comparison instruction
   - When the required frequency is beyond the counting performance.
   - When counting is regarded as important, but the effect of the scan time can be ignored in operations according to the counting result.
   - When the number of an instruction exceeds the allowable limit.

For FX3U and FX3UC PLCs



*1. This defines CN251.
*2. This defines D100.
*3. This defines K1000.
*4. This defines D100.

For FX0, FX0S, FX0N, FXU, FX2C, FX1S, FX1N, FX1NC, FX2N, FX2NC and FX3G PLCs



*1. This defines K100.
*2. This defines CN251.

2. Device specification range

Only high speed counters can be specified as (s2).
For details, refer to the following manual.

→ **FX Structured Programming Manual (Device & Common)**

3. Some restrictions to applicable devices

▲1: Applicable to the FX3U and FX3UC PLCs only.
Not indexed.
▲2: Applicable to the FX3U, FX3UC and FX3G PLCs only.
▲3: Applicable to the FX3U and FX3UC PLCs only.
▲4: The same counter of the device specified by (s2) can be used.
(See the program example.)

4. Specifying input and output variables

When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

5. Precedence of DHSCS, DHSCR and DHSZ instructions to one particular high speed counter

→ **Refer to caution 6 in "Common cautions on using instructions for high speed counter" which is described in Section 7.6.4.**

6. Reset operation by an external terminal

→ **Refer to caution 5 in "Common cautions on using instructions for high speed counter" which is described in Section 7.6.4.**

7. Other cautions on use

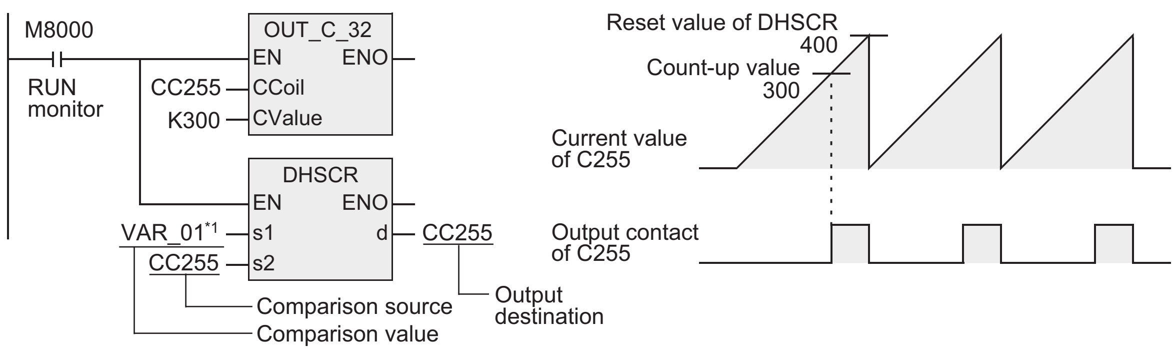
→ **Refer to caution in "Common cautions on using instructions for high speed counter" which is described in Section 7.6.4.**

Program examples

1. Example of self-reset circuit

When the current value of C255 becomes "400", C255 is immediately reset. Its current value becomes "0", and the output contact is set to OFF.

[Structured ladder]



*1. VER_01 is a global label and is defined as K400.

[ST]

```
OUT_C_32(M8000,CC255,K300);
DHSCR(M8000,VAR_01,CC255,CC255);
```

7.6.6 DHSZ

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	○	×	×

Outline

This instruction compares the current value of a high speed counter with two values (one zone), and outputs the comparison result to three bit devices (refresh).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DHSZ	32 bits	Continuous	DHSZ EN ENO s1 d s2 s	DHSZ(EN,s1,s2,s,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Data to be compared with the current value of a high-speed counter or word device storing the data to be compared (Comparison value 1)	ANY32
	s2	Data to be compared with the current value of a high-speed counter or word device storing the data to be compared (Comparison value 2)	ANY32
	s	Device of a high speed counter	ANY32
Output variable	EN0	Execution state	Bit
	d	Head device to which the comparison result is output based on upper and lower comparison values	Bit

3. Applicable devices

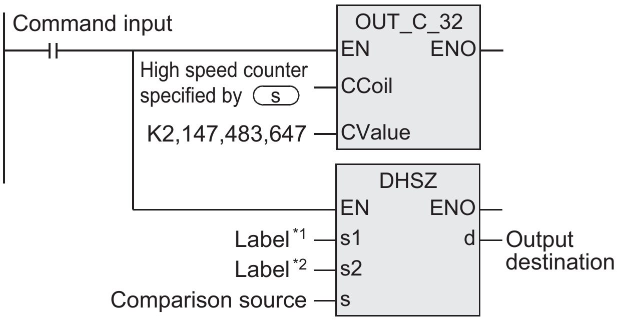
Operand type	Bit Devices							Word Devices												Data type				
	System User							Digit Specification				System User				Special Unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲2	▲3		●	●	●	●			
s2								●	●	●	●	●	●	●	▲2	▲3		●	●	●	●			
s													●						●					
d		●	●			●	▲1												●					

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DHSZ)

The current value of the high speed counter of the device specified by (s) is compared with two comparison points (comparison value 1 and comparison value 2). Based on the comparison result, "smaller than the lower comparison value", "inside the comparison zone" or "larger than the upper comparison value", one among the devices specified by (d) is set to ON regardless of the operation cycle.
In this instruction, the comparison processing is executed after the count processing in the high speed counter.



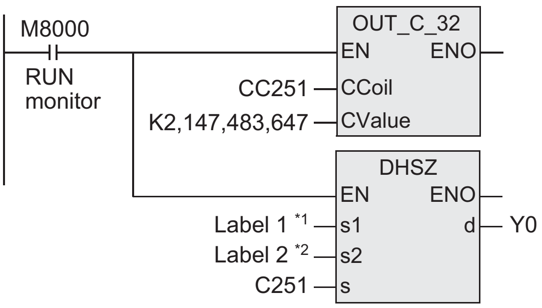
Set to ON

(s1) > (s) → (d)
(s1) ≤ (s) ≤ (s2) → (d)+1
(s) > (s2) → (d)+2

*1. This defines the comparison value 1.
*2. This defines the comparison value 2.

Comparison points
Make sure that the comparison value 1 and the comparison value 2 have the following relationship:
(s1) ≤ (s2)

Operation
When the current value of the high speed counter C251 changes (counts) as shown below, the comparison result is output to one of the outputs Y000, Y001 or Y002.



ON

K1000 > C251 → Y000
K1000 ≤ C251 ≤ K2000 → Y001
C251 > K2000 → Y002

*1. This defines K1000.
*2. This defines K2000.

Comparison pattern	Current value of C251	Change of output contact (Y)		
		Y000	Y001	Y002
(s1) > (s)	1000> (s)	ON	OFF	OFF
	999→1000	ON→OFF	OFF→ON	OFF
	999←1000	OFF→ON	ON→OFF	OFF
(s1) ≤ (s) ≤ (s2)	999→1000	ON→OFF	OFF→ON	OFF
	999←1000	OFF→ON	ON→OFF	OFF
	1000≤ (s) ≤2000	OFF	ON	OFF
	2000→2001	OFF	ON→OFF	OFF→ON
	2000←2001	OFF	OFF→ON	ON→OFF
(s) < (s2)	2000→2001	OFF	ON→OFF	OFF→ON
	2000←2001	OFF	OFF→ON	ON→OFF
	(s) >2000	OFF	OFF	ON

Related instruction

The following instructions can be combined with high speed counters.

Instruction	Instruction name
DHSCS	High speed counter set
DHSCR	High speed counter reset
DHSZ	High speed counter zone compare
DHCMOV	High speed counter move
DHSCT	High speed counter compare with data table

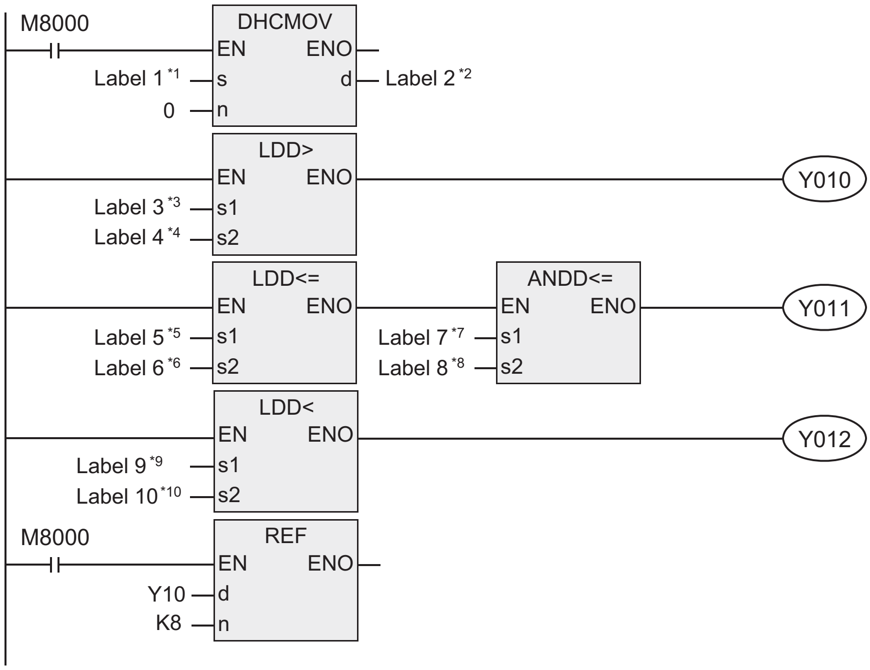
Cautions

1. Selection of the counter comparison method

When the DHSCS instruction is used, the maximum frequency and total frequency of the high speed counter are affected.
Refer to the counting operation described below, and select according to the contents of control whether to use this instruction or general-purpose comparison instruction.

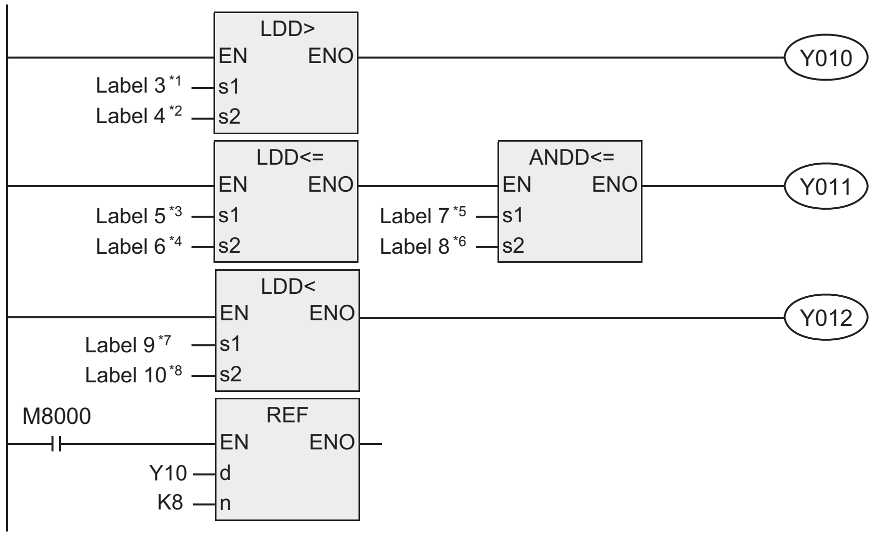
1) Case to select DHSCS instruction
   - When the output should be given when the counting result becomes equivalent to the comparison value without regard to the scan time of the PLC.
2) Cases to select a general-purpose comparison instruction
   - When the required frequency is beyond the counting performance.
   - When counting is regarded as important, but the effect of the scan time can be ignored in operations according to the counting result.
   - When the number of an instruction exceeds the allowable limit.

   For FX3U and FX3UC PLCs



*1. This defines CN251.
*2. This defines D100.
*3. This defines K10000.
*4. This defines D100.
*5. This defines K10000.
*6. This defines D100.
*7. This defines K20000.
*8. This defines D100.
*9. This defines K20000.
*10. This defines D100.

For FX0, FX0S, FX0N, FXU, FX2C, FX1S, FX1N, FX1NC, FX2N, FX2NC and FX3G PLCs



*1. This defines K10000.
*2. This defines CN251.
*3. This defines K10000.
*4. This defines CN251.
*5. This defines K20000.
*6. This defines CN251.
*7. This defines K20000.
*8. This defines CN251.

2. Device specification range

Only high speed counters can be specified as (s2).
For details, refer to the following manual.

→ **FX Structured Programming Manual (Device & Common)**

3. Some restrictions to applicable devices

▲1: Applicable to the FX3U and FX3UC PLCs only.
Not indexed (V, Z).
▲2: Applicable to the FX3U, FX3UC and FX3G PLCs only.
▲3: Applicable to the FX3U and FX3UC PLCs only.

4. Specifying input and output variables

When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

5. Caution on values set in the comparison value 1 (s1) and comparison value 2 (s2)

Make sure that (s1) is smaller than or equivalent to (s2).

6. Relationship between the comparison timing and the result output

1) DHSZ instruction executes comparison and outputs the result only when a counting pulse is input to a high speed counter.
(When (s1) is "1000" and (s2) is "1999", the output (d) is set to ON as soon as the current value of C235 changes from "999" to "1000" or from "1999" to "2000".)
2) Because the comparison result cannot be obtained when restoring the power or when the PLC mode switches from STOP to RUN, the result is not output even if the comparison condition is provided.

→ **For details, refer to "Program in which comparison result is set to ON when power is turned ON [ZCP] instruction" that is described later.**

7. Precedence of DHSCS, DHSCR and DHSZ instructions to one particular high speed counter

→ **Refer to caution 6 in "Common cautions on using instructions for high speed counter" which is described in Section 7.6.4.**

8. Reset operation by an external terminal

→ Refer to caution 5 in "Common cautions on using instructions for high speed counter" which is described in Section 7.6.4.

9. Number of occupied devices

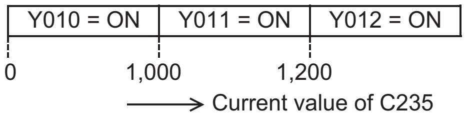
1) The comparison value occupies two devices from s1 or s2 respectively.
2) The output occupies three devices from d .

Program in which comparison result is set to ON when power is turned ON [ZCP] instruction

DHSZ instruction outputs the comparison result only when a counting pulse is input. Even if the current value of C235 is "0", Y010 remains OFF at the time of startup.
For initializing Y010, compare the current value of C235 with K1000 and K1200 and drive Y010 by DZCPP instruction (for general zone comparison) as pulse operation only at the time of startup.
Refer to the program example shown below.

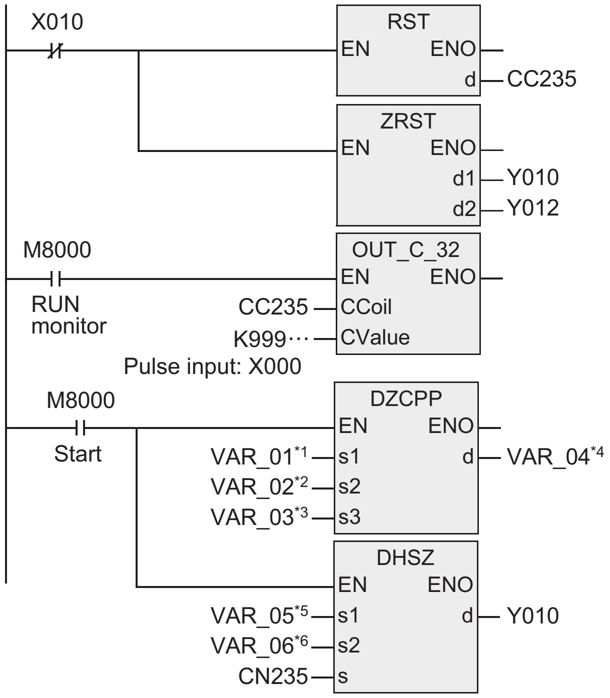
Explanation of operation

The outputs Y010 to Y012 are as shown below.



Program examples

[Structured ladder]



Y010 to Y012 are reset.

Immediately after start, comparison is executed only once.
K1000 > CN235 :Y010 ON
K1000 ≤ CN235 ≤ K1200 :Y011 ON
K1200 < CN235 :Y012 ON

Immediately after start, comparison is executed by interrupt when each pulse is input from X000.
K1000 > CN235 :Y010 ON
K1000 ≤ CN235 ≤ K1200 :Y011 ON
K1200 < CN235 :Y012 ON

*1. VER_01 is a global label and is defined as K1000.
*2. VER_02 is a global label and is defined as K1200.
*3. VER_03 is a global label and is defined as CN235.
*4. VER_04 is a global label and is defined as Y010.
*5. VER_05 is a global label and is defined as K1000.
*6. VER_06 is a global label and is defined as K1200.

[ST]

```
RST(NOT X010,);
ZRST(NOT X010,Y010,Y012);
OUT_C_32(M8000,CC235,K999···);
DZCPP(X010,VAR_01,VAR_02,VAR_03,VAR_04);
DHSZ(X010,VAR_05,VAR_06,CC235,Y010);
```

Timing chart

In the part 1) in the timing chart, Y010 remains OFF if the current value of a high speed counter (C235 in the example below) is "0" when restoring the power.

1) For initializing Y010, the current value of C235 is compared with K1000 and K1200, and Y010 is driven using the DZCPP instruction (for general zone comparison) as pulse operation only in RUN.
2) The comparison result in Y010 is latched until an input pulse is input and the comparison output is driven by the DHSZ instruction.
3) According to the current value of the counter, the DHSZ instruction drives the output Ⓐ, Ⓑ or Ⓒ.

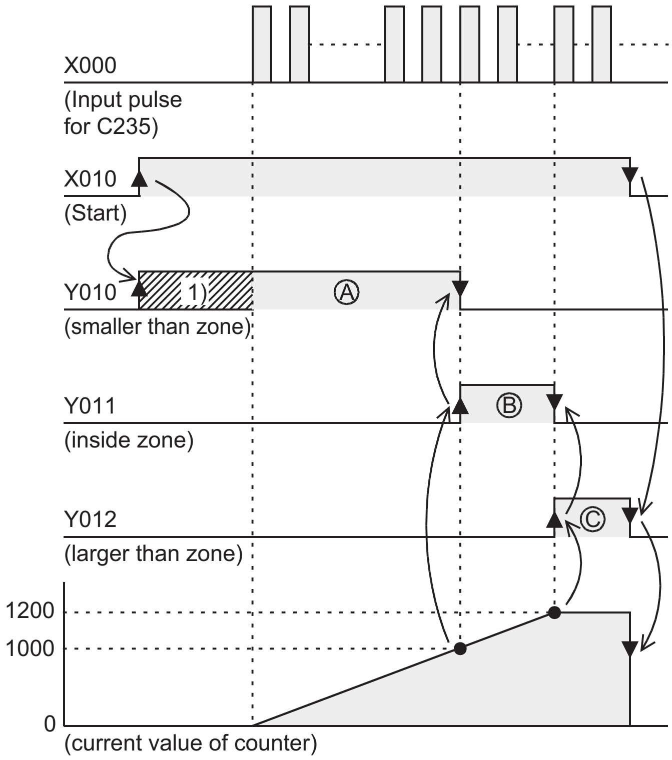


Table high speed comparison mode (M8130)

This section explains the table high speed comparison mode (high speed pattern output) of the DHSZ instruction.
When two or more outputs should be activated at one time, use the DHSCT instruction which can change up to 16 outputs.
(Valid for the FXU PLC, V3.07 or later)

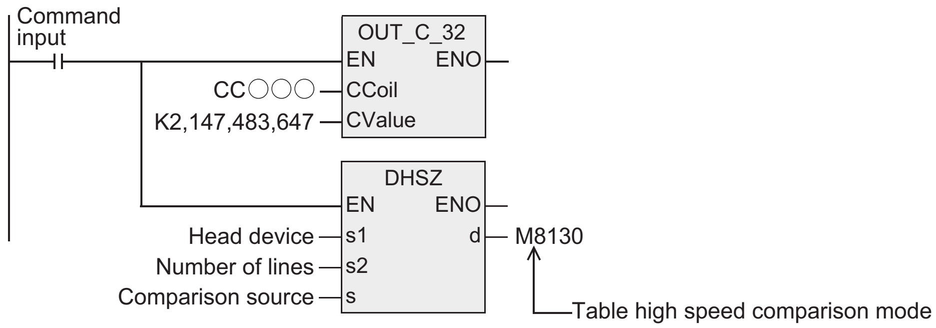
1. Set data

Operand type	Description	Data type
(s1)	Head word device number storing the data table (only data register D)	ANY32
(s2)	Number of lines in the table (only K or H) ... K1 to K128 or H1 to H80	ANY32
(s)	Device number of a high speed counter	ANY32
(d)	Special auxiliary relay for declaring the table high speed comparison mode	Bit

Function and operation explanation

1. 32-bit operation (DHSZ)

When the special auxiliary relay M8130 for declaring the table high speed comparison mode is specified as (d) in the DHSZ instruction, the special function shown below is provided.



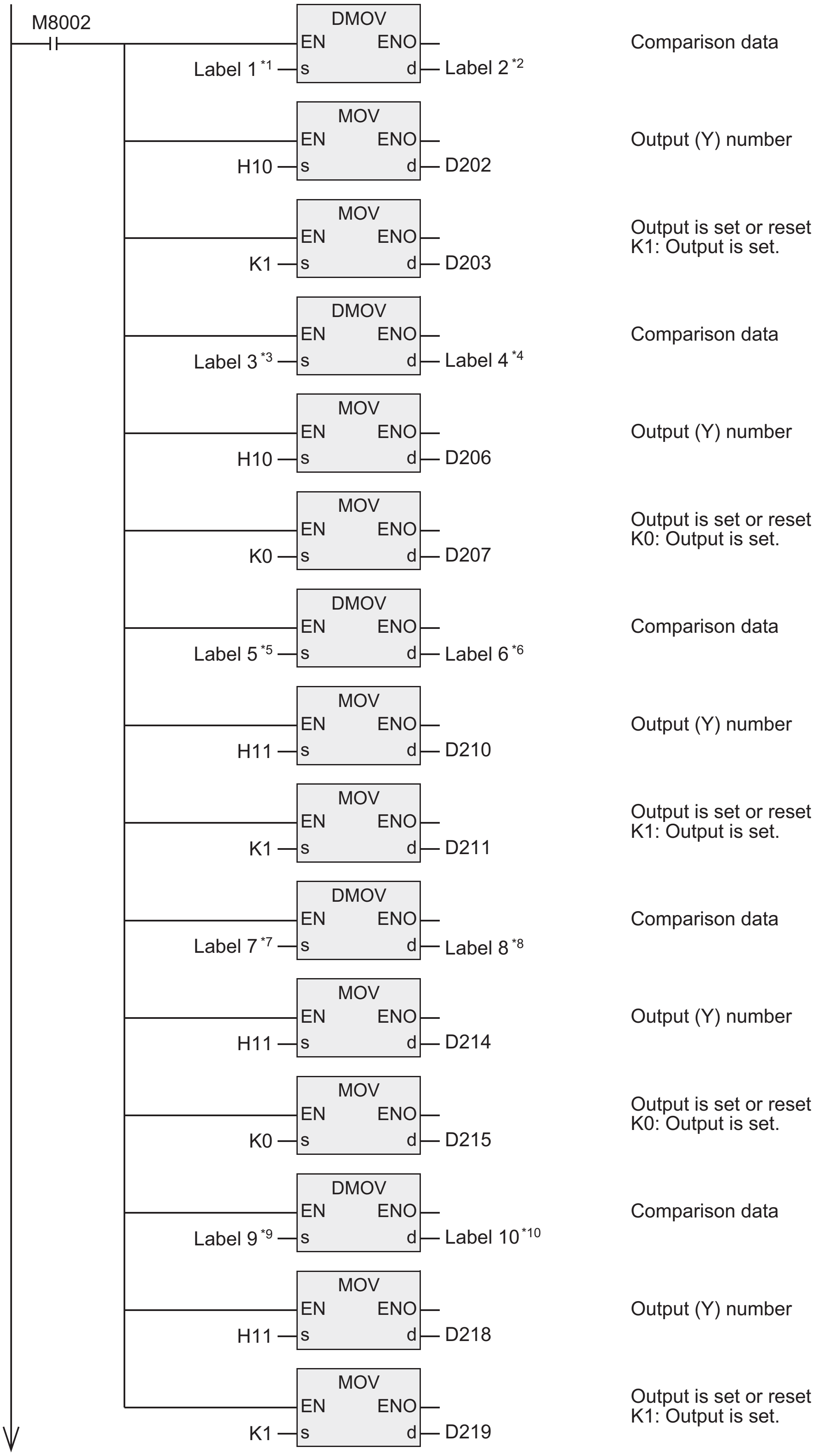
Comparison table

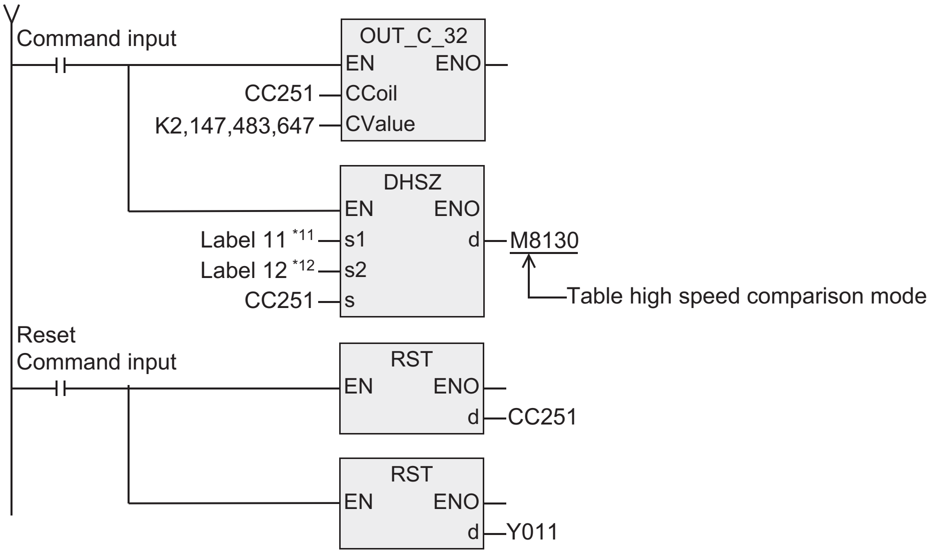
Comparison data (32 bits)	Output (Y) number	SET/RST	Table counter (D8130)
s1 +1, s1	s1 +2	s1 +3	0 ↓
s1 +5, s1 +4	s1 +6	s1 +7	1 ↓
s1 +9, s1 +8	s1 +10	s1 +11	2 ↓
to	to	to	to
s1 +5, s1 +4	s1 +6	s1 +7	s2 -1 ↓ Repeated from "0"

1) Specify the head number for the comparison table as s1
Because one line in the comparison table uses four devices, s2 × 4 devices are occupied from s1 .
2) Specify the number of lines in the comparison table as s2 .
The created table starts from the head register s1 , and has the number of lines specified in s2 .
3) Comparison data
Make sure that the comparison data is 32 bits.
4) Output (Y) number
Specify each digit of the (Y) number in hexadecimal form.
Example: When specifying Y010, specify "H10".
When Specifying Y020, specify "H20".
5) Specification of set and reset
These set and reset are directly controlled as interrupt.

	Contents of setting
Set (ON)	K1/H1
Reset (OFF)	K0/H0

2. Operation

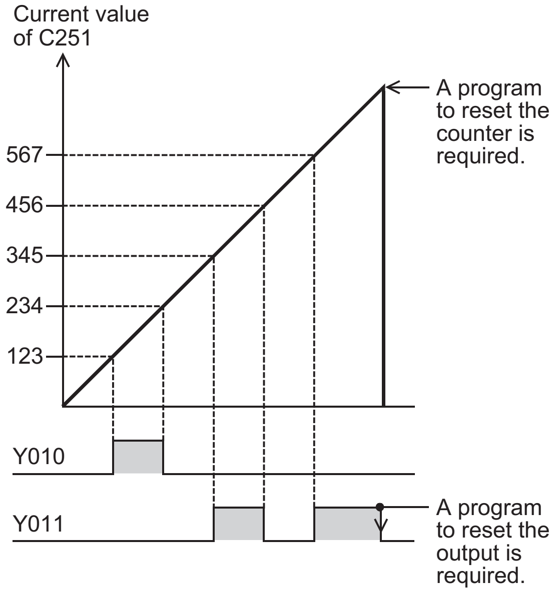

M8002
DMOV EN ENO s d
Label 1 *1
Label 2 *2
Comparison data
MOV EN ENO s d
H10
D202
Output (Y) number
MOV EN ENO s d
K1
D203
Output is set or reset
K1: Output is set.
DMOV EN ENO s d
Label 3 *3
Label 4 *4
Comparison data
MOV EN ENO s d
H10
D206
Output (Y) number
MOV EN ENO s d
K0
D207
Output is set or reset
K0: Output is set.
DMOV EN ENO s d
Label 5 *5
Label 6 *6
Comparison data
MOV EN ENO s d
H11
D210
Output (Y) number
MOV EN ENO s d
K1
D211
Output is set or reset
K1: Output is set.
DMOV EN ENO s d
Label 7 *7
Label 8 *8
Comparison data
MOV EN ENO s d
H11
D214
Output (Y) number
MOV EN ENO s d
K0
D215
Output is set or reset
K0: Output is set.
DMOV EN ENO s d
Label 9 *9
Label 10 *10
Comparison data
MOV EN ENO s d
H11
D218
Output (Y) number
MOV EN ENO s d
K1
D219
Output is set or reset
K1: Output is set.



*1. This defines K123.
*2. This defines D200.
*3. This defines K234.
*4. This defines D204.
*5. This defines K345.
*6. This defines D208.
*7. This defines D456.
*8. This defines D212.
*9. This defines D567.
*10. This defines D216.
*11. This defines D200.
*12. This defines K5.

Comparison table

Comparison data	Output (Y) number	SET/RST	Table counter
D201, D200 K123	D 202 H10	D 203 K1	0 ↓
D205, D204 K234	D 206 H10	D 207 K0	1 ↓
D209, D208 K345	D 210 H11	D 211 K1	2 ↓
D213, D212 K456	D 214 H11	D 215 K0	3 ↓
D217, D216 K567	D 218 H11	D 219 K1	4 ↓ Repeated from "0"



1) When this instruction is executed, the top table in the data table is set as the comparison target data.
2) When the current value of the high speed counter C251 is equivalent to the comparison target data table, the output (Y) number specified in the comparison data table is set or reset.
   This output processing is directly executed without regard to completion of output refresh by END instruction.
3) "1" is added to the current value of the table counter D8130.
4) The comparison target data table is transferred to the next table.
5) The step 2) and 3) are repeated until the current value of the table counter D8130 becomes "4".
   When the current value becomes "4", the program execution returns to the step 1), and the table counter D8130 is reset to "0".
   At this time, the complete flag M8131 turns ON.
6) When the command contact is set to OFF, execution of the instruction is stopped and the table counter D8130 is reset to "0".

Cautions

1. Limitation in the number of DHSZ instruction

This instruction can be programmed only once in a program.
With regard to the DHSCS, DHSCR, DHSZ and DHSCT instructions used for other purposes, a limited number of instructions including the DHSZ instruction can be driven at one time.

2. When the command input is set to OFF in the middle of execution

Execution of the instruction is aborted, and the table counter D8130 is reset to K0.
However, outputs which have been set or reset remain in the current status.

3. Output start timing

After the DHSZ instruction is first executed, creation of the table is completed by END instruction. After that, the DHSZ instruction becomes valid.
Accordingly, the output is activated from the second scan.

4. Current value of a high speed counter

Be sure to execute the DHSZ instruction from a point where the current value of the high speed counter (regarded as the operation target) is smaller than the value in the first line in the comparison table.

Frequency control mode (DHSZ, DPLSY) (M8132)

When the special auxiliary relay M8132 for declaring the frequency control mode is specified as (d) in the DHSZ instruction, the special function shown below is provided if DPLSY instruction is combined.
At this time, only a data register D can be specified as (s1), and a constant K or H can be specified as (s2).
The available range is limited to "1 ≤ K, H ≤ 128".
A high speed counter can be specified as (s).
This function is different from the zone comparison described above.
PLSY instruction is as shown on the next page, and only the pulse output can be changed by users.
(Valid for the FXU PLC, V3.07 or later)

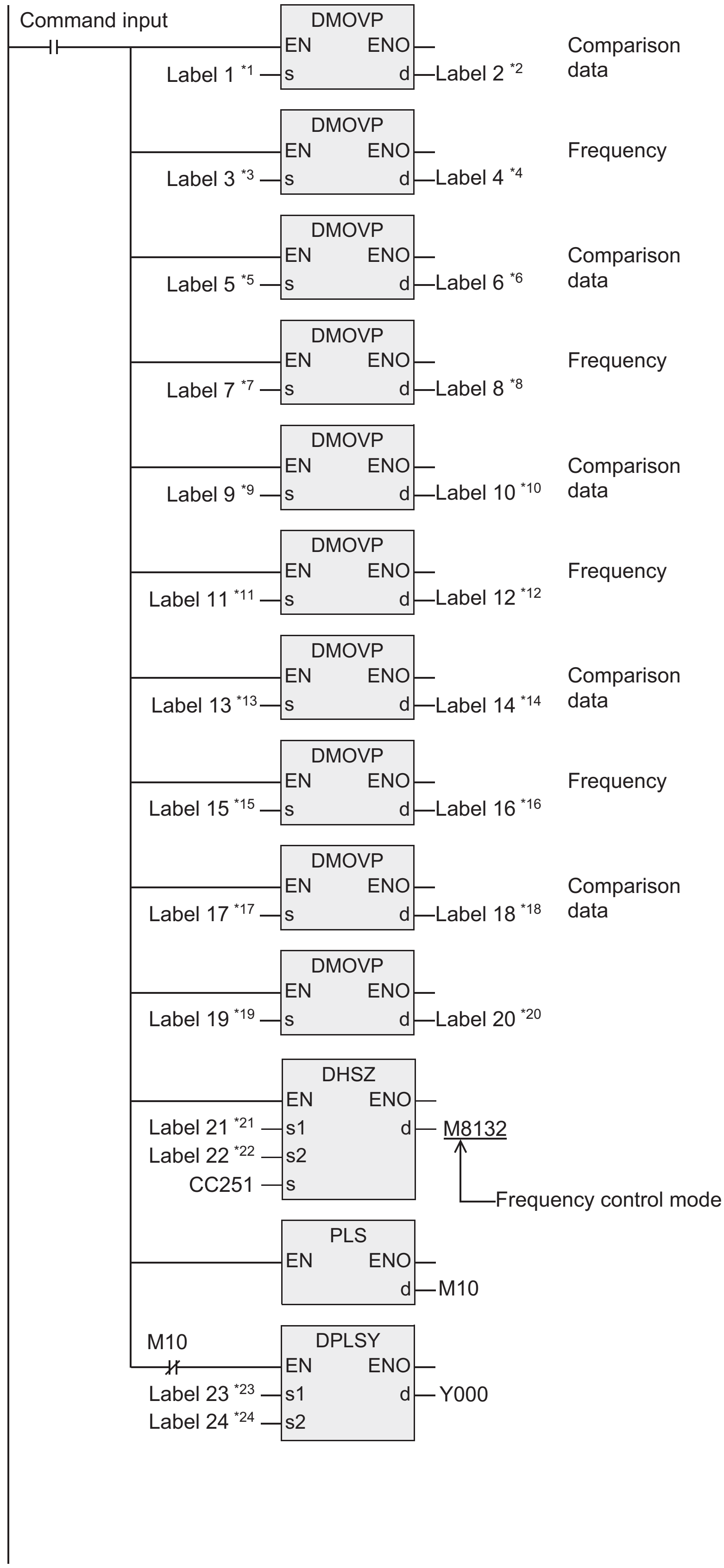
1. Control example

Example of table configuration and data setting

Comparison data	Frequency	Table counter (D8131)
D 301, D 300 K 20	D 302, D 303 K300	0 ↓
D 305, D 304 K600	D 306, D 307 K500	1 ↓
D 309, D 308 K700	D 310, D 311 K200	2 ↓
D 313, D 312 K800	D 314, D 315 K100	3 ↓
D 317, D 316 K 0	D 318, D 319 K 0	4 ↓

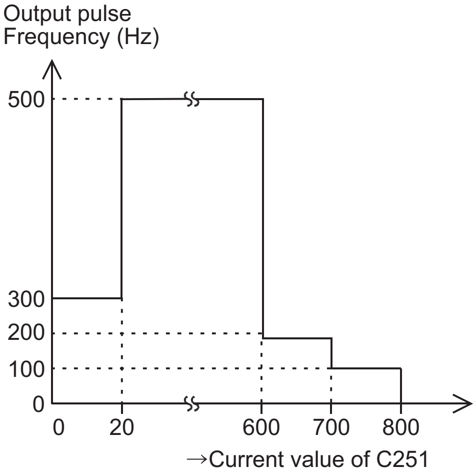
Head device (32 bits) specified as (s1)

Number of lines specified as (s2)



*1. This defines K20.
*2. This defines D300.
*3. This defines K300.
*4. This defines D302.
*5. This defines K600.
*6. This defines D304.
*7. This defines K500.
*8. This defines D306.
*9. This defines D700.
*10.This defines D308.
*11.This defines D200.
*12.This defines D310.
*13.This defines K800.
*14.This defines D312.
*15.This defines K100.
*16.This defines D314.
*17.This defines K0.
*18.This defines D316.
*19.This defines K0.
*20.This defines D318.
*21.This defines the head device.
*22.This defines the number of lines.
*23.This defines D8132.
*24.This defines K0.

Output pulse characteristics



1) Write prescribed data in advance to data registers constructing the table as shown in this program example.
2) The output frequency of the DPLSY instruction remains in the value (D303, D302) until the current value of a high speed counter specified in (s) becomes equivalent to (D301, D300). (D302 specifies low-order 16 bits. D303 specifies high-order 16 bits, but is always "0".)
3) The operation in the second line is started after that, and then the operation in each line is executed in turn.
4) When the operation in the last line is completed, the complete flag M8133 turns ON. The program execution returns to the first line, and the operation is repeated.
5) For stopping the operation in the last line, set the frequency in the last table to K0.
6) When the command input is set to OFF, the pulse output turns OFF and the table counter D8131 is reset.
7) After DHSZ instruction is first executed, creation of the table is completed at the END instruction. The DHSZ instruction becomes valid after that.
8) Accordingly, the contact of PLS M10 is used so that the DPLSY instruction is executed from the second scan after the command input has been set to ON.

Data can be written to the table in a program as shown in this example or directly using keys in peripheral equipment.

1) M8132: This is the special auxiliary relay for declaring the frequency control mode.
2) D8132: In the frequency control mode, the frequency set in the table is received by D8132 sequentially according to the table counter D8131 count.
3) D8134 (low-order), D8135 (high-order):
In the frequency control mode, the comparison data in the table is received sequentially according to the table counter count.

Cautions

1) DHSZ instruction can be used only once.
2) With regard to the DHSCS, DHSCR, DHSZ and DHSCT instructions for other purposes, a limited number of instructions including the DHSZ instruction can be driven at one time.
3) Because the table is created when the END instruction is executed, it is necessary to delay execution of the DPLSY instruction until creation of the table is completed.
4) Do not change the data table while the DHSZ instruction is driven.
5) In the frequency control mode, simultaneous output to Y000 to Y001 is not permitted.

7.6.7 SPD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This instruction counts the input pulse for a specified period of time as interrupt input.
The function of this instruction varies depending on the version.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SPD	16 bits	Continuous	SPD EN ENO s1 d s2	SPD(EN,s1,s2,d);
DSPD	32 bits	Continuous	DSPD EN ENO s1 d s2	DSPD(EN,s1,s2,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Device of pulse input (X)	Bit	
	s2	Time data (ms) or word device storing the data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head word device storing the pulse density data	ARRAY [0..2] OF ANY16	ARRAY [0..2] OF ANY32

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1	▲1																		●					
s2								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
d												●	●	●	▲2		●	●	●					

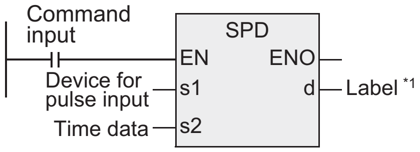
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (SPD)

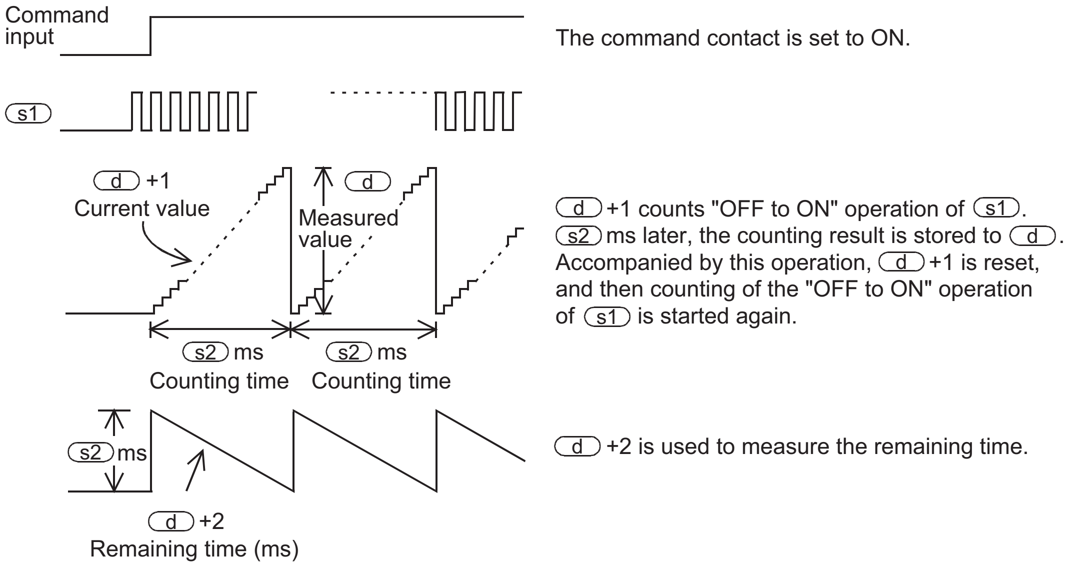
The input pulse specified by (s1) is counted only for the period of "time specified by (s2) multiplied by 1 ms." The measured value is stored in (d), the current value is stored in (d)+1, and the remaining time is stored in (d)+2 (ms).

By repeating this operation, the measured value (d) will store the pulse density (which is proportional to the rotation speed).

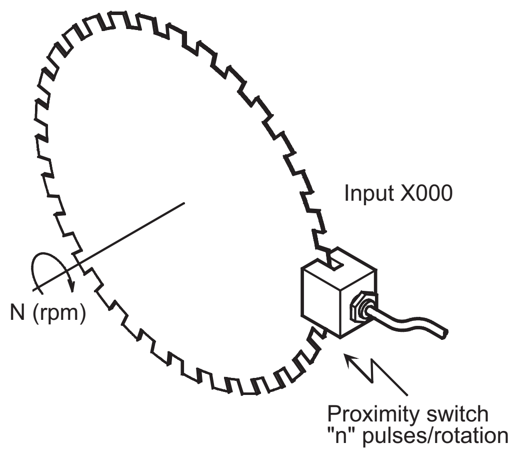


*1. This defines the device that stores the pulse density data.

1) Timing chart



2) The measured value (d) is in proportion to the number of rotations as shown below.

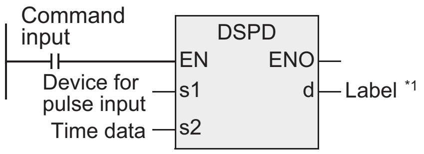


$$N = \frac{60(d)}{nt} \times 10^3 \text{ (r/min)}$$

t: Measurement time duration specified as (s2) (ms).

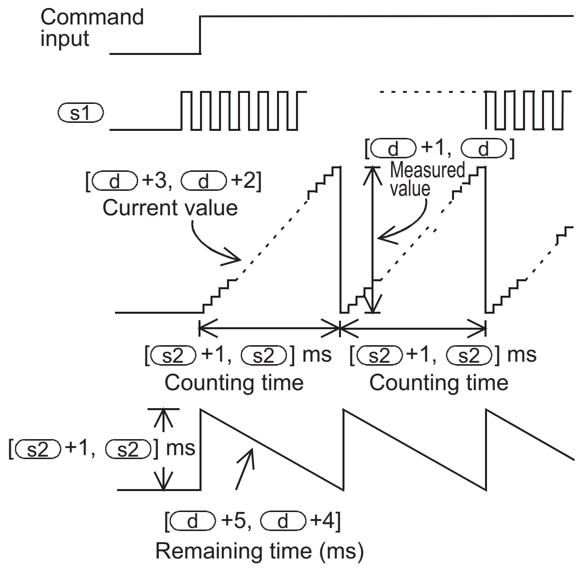
2. 32-bit operation (DSPD) [FX3U, FX3UC Ver.2.20 or later]

The input pulse specified by (s1) is counted only for the period of "time specified by (s2) multiplied by 1 ms." The measured value is stored in [(d)+1, (d)], the current value is stored in [(d)+3, (d)+2], and the remaining time is stored in [(d)+5, (d)+4] (ms).
By repeating this operation, the measured value [(d)+1, (d)] will store the pulse density (which is proportional to the rotation speed).



*1. This defines the device that stores the pulse density data.

1) Timing chart

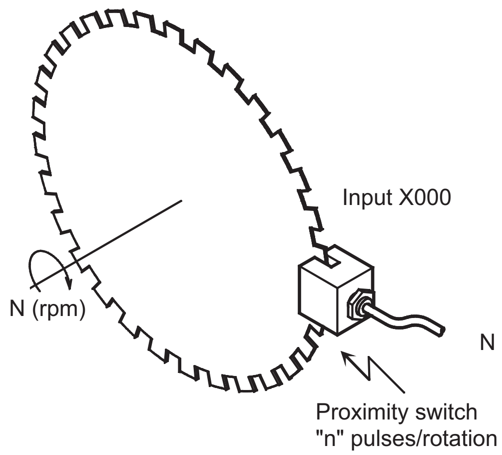


The command contact is set to ON.

[(d)+3, (d)+2] counts "OFF to ON" operation of (s1).
[(s2)+1, (s2)] ms later, the counting result is stored to [(d)+1, (d)].
Accompanied by this operation, [(d)+3, (d)+2] is reset, and then counting of the "OFF to ON" operation of (s1) is started again.

[(d)+5, (d)+4] is used to measure the remaining time.

2) The value [(d)+1, (d)] is in proportion to the number of rotations as shown below.



$$N = \frac{60\,[(d)+1,\ (d)]}{nt} \times 10^{3}\ \text{(r/min)}$$

t: Measurement time duration specified as [(s2)+1, (s2)] (ms).

Cautions

1. Input specifications of the input specified by (s1)

1) (s1) can specify the following ranges.
FX3U, FX3UC, and FX3G PLCs: X000 to X007
FX1S, FX1N, FX2N, FX1NC and FX2NC PLCs: X000 to X005
FXU and FX2C PLCs: X000 to X005

2) An input device X000 to X007 (X000 to X005) specified by (s1) cannot overlap the following functions or instructions:
- High speed counter
- Input interrupt
- Pulse catch
- Pulse width measurement
- DSZR
- DVIT
- ZRN

3) For one input, this instruction can be used only once.

4) The maximum input frequency is shown below:

Used input number	FX3UC PLC	FX3U PLC	
		Main unit	FX3U-4HSX-ADP
X000 to X005	100kHz*1	100kHz*1	200kHz
X006 to X007	10kHz	10kHz	

*1. When receiving pulses within the response frequency range of 50k to 100 kHz, perform the following actions:
- Make sure that the wiring length is 5 m or less.
- Connect a bleeder resistor of 1.5 kΩ (1 W or more) to the input terminal, and make sure that the load current in the open collector transistor output of the external equipment is 20 mA or more.

Used input number	FX3G PLC
X000, X001, X003, X004	60kHz
X002, X005, X006, X007	10kHz

Used input number	FX1S, FX1N, FX2N, FX1NC and FX2NC PLCs
X000, X001	60kHz
X002, X003, X004, X005	10kHz

Used input number	FXU and FX2C PLCs
X000, X002, X003	10kHz
X001, X004, X005	7kHz

2. Specifying input and output variables

When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

3. Occupied devices

1) When using the 16-bit operation
Three devices are occupied from a device specified in (d).

2) When using the 32-bit operation
Six devices are occupied from a device specified in (d).

4. Restrictions to devices

▲1: X000 to X007 (X005) can be specified.
Refer to "Cautions".
▲2: Applicable to the FX3U, FX3UC and FX3G PLCs only.
▲3: Applicable to the FX3U and FX3UC PLCs only.

7.6.8 PLSY

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction generates a pulse signal.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
PLSY	16 bits	Continuous	PLSY EN ENO s1 d s2	PLSY(EN, s1, s2, d);
DPLSY	32 bits	Continuous	DPLSY EN ENO s1 d s2	DPLSY(EN, s1, s2, d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Frequency data (Hz) or the word device storing the data	ANY16	ANY32
	s2	Pulse quantity data or the word device storing the data	ANY16	ANY32
Output variable	EN0	Execution state	Bit	
	d	Bit device (Y) from which pulses are output	Bit	

3. Applicable devices

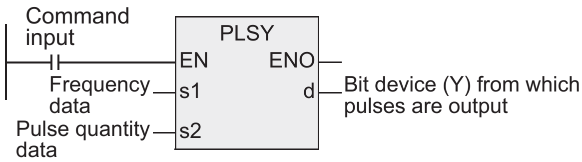
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
s2								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
d		▲1																						

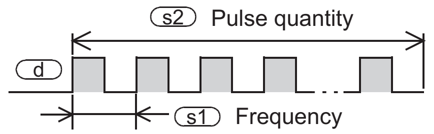
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (PLSY)

A pulse train of frequency specified by (s1) is output in the quantity specified by (s2) from the output (Y) specified by (d).

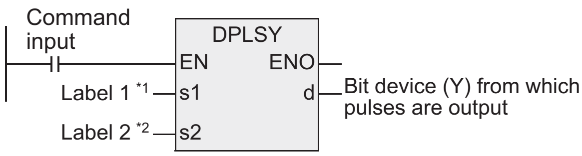


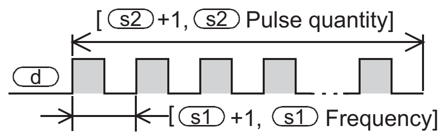


For setting (s1), (s2) and (d), refer to the "cautions".

2. 32-bit operation (DPLSY)

A pulse train of frequency specified by (s1) is output in the quantity specified by (s2) from the output (Y) specified by (d).





*1. This defines the frequency data.
*2. This defines the pulse quantity data.
For setting (s1), (s2) and (d), refer to the "cautions".

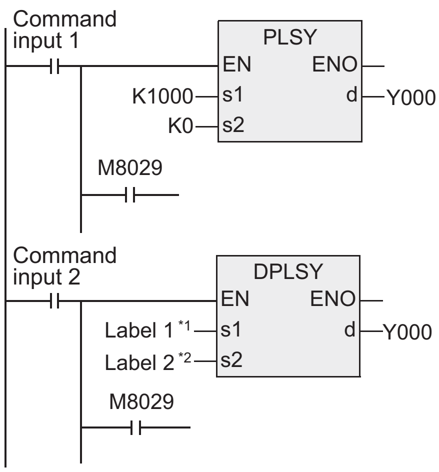
Related devices

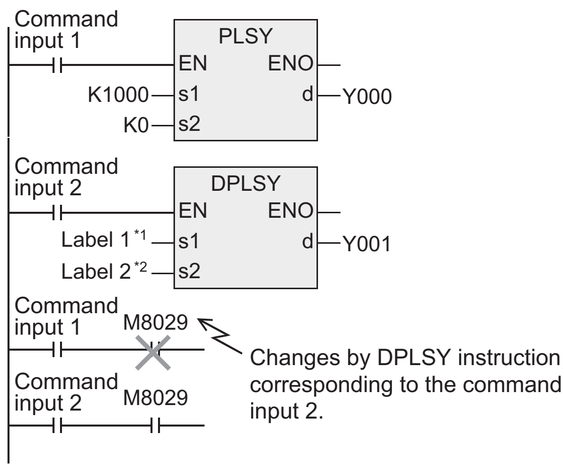
1. Instruction execution complete flag

The instruction execution complete flag M8029 used for PLSY instruction can be used also for other instructions. When using other instructions, setting the M8029 flag to ON or OFF, or using two or more PLSY instructions, be sure to use each M8029 flag just after an instruction to be monitored.
For the method of using the instruction execution complete flag, refer to the following manual.

→ **FX Structured Programming Manual (Device & Common)**

Device	Name	Description
M8029	Instruction execution complete	ON : Generation of specified number of pulses is completed. OFF : Generation of pulses is paused before the specified number of pulses is reached or the continuous pulse generation operation is stopped.





*1. This defines K1000.
*2. This defines K0.

2. Monitoring the current number of generated pulses

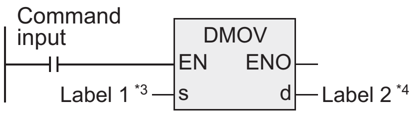
The number of pulses output from Y000 or Y001 is stored in the following special data resistors.

Device		Description	Contents of data
High order	Low order		
D8141*1	D8140*1	Accumulated number of pulses output from Y000	Accumulated number of pulses output from Y000 by PLSY and PLSR instructions
D8143*1	D8142*1	Accumulated number of pulses output from Y001	Accumulated number of pulses output from Y001 by PLSY and PLSR instructions
D8137*2	D8136*2	Total accumulated number of pulses output from Y000 and Y001	Total accumulated number of pulses output from Y000 and Y001 by PLSY and PLSR instructions.

*1. The FX0, FX0S, FX0N, FXU or FX2C PLC is not compatible with this function.

*2. The FX0, FX0S or FX0N PLC is not compatible with this function.
The FXU PLC of V3.07 or later is compatible.

The contents of each data register can be cleared using the following program.



*3. This defines K0.
*4. This defines the low order devices in the table above.

3. How to stop the pulse output

- When the command input is set to OFF, the pulse generation is immediately stopped. When the command input is set to ON again, pulse generation operation restarts from the beginning.
- When the special auxiliary relays (M) shown below are set to ON, the pulse output is stopped.

Device			Description
FX3U, FX3UC	FX3G	FX1S, FX1N, FX1NC	
M8349	M8145, M8349	M8145	Immediately stops pulse output from Y000.
M8359	M8146, M8359	M8146	Immediately stops pulse output from Y001.

To restart pulse output, set the device corresponding to the output signal to OFF, and then drive the pulse output instruction again.

Cautions

1. When a word device is specified as (s1) or (s2)

When the value of the word device is changed while the instruction is executed, the following operation results.

- When the data in (s1) is changed, the output frequency changes accordingly.
- When the data in (s2) is changed, the change (new value) becomes valid the next time the instruction is driven.

2. Frequency (s1)

When using transistor outputs in the main unit, set the output frequency specified by (s1) as follows.

- FX3U and FX3UC PLCs : 16-bit instruction→1 to 32,767Hz
  32-bit instruction→1 to 200,000Hz (When using special high speed output adapter)
  →1 to 100,000Hz (When using main unit)
- FX3G PLC : 16-bit instruction→1 to 32,767Hz
  32-bit instruction→1 to 100,000Hz
- FX2N and FX2NC PLCs : 2 to 20,000Hz
- FX1NC PLC : 1 to 10,000Hz
- FX1S and FX1N PLCs : 16-bit instruction→1 to 32,767Hz
  32-bit instruction→1 to 100,000Hz
- FX0, FX0S and FX0N PLCs : 10 to 2,000Hz
- FXU and FX2C PLCs : 1 to 1,000Hz

3. Specifying input and output variables

When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

4. Pulse quantity (s2)

The pulse quantity can be set in the range from 1 to 32767 (PLS) for 16-bit instructions and from 1 to 2,147,483,647 (PLS) for 32-bit instructions. If set to zero, pulse generates infinitely.

5. Pulse output

1) Only a transistor output on the main unit or the following special high speed output adapters*1 can be specified in (d).

- FX3U, FX3UC and FX3G : Y000, Y001
- FX1S, FX1N, FX2N, FX1NC and FX2NC PLCs : Y000, Y001
- FX0, FX0S and FX0N PLCs : Y000
- FXU and FX2C PLCs : Valid for all Y

When using the PLSY instruction with a relay output type FX3U PLC, a special high speed output adapter is required.

*1. Special high speed output adapters can be connected only to the FX3U PLC.

2) The duration of the ON/OFF pulses is 50% (ON = 50%, OFF = 50%)

3) The pulse output is controlled by the dedicated hardware not affected by the sequence program (operation cycle).

4) If the command input is set to OFF during continuous pulse output, the output from the device specified by (d) turns OFF.

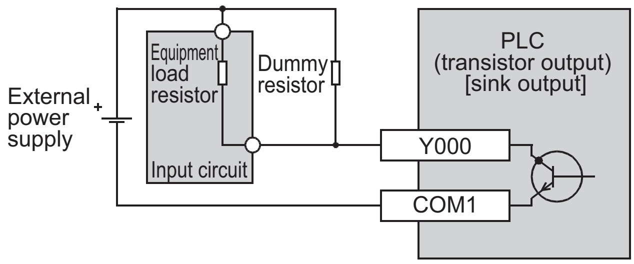
6. Restrictions to target devices

▲1: Refer to item 1 of "Cautions".
▲2: Applicable only to the FX3U, FX3UC and FX3G PLCs.
▲3: Applicable only to the FX3U and FX3UC PLCs.
(Special high speed output adapters can be connected only to the FX3U PLC.)

7. Handling of pulse output terminals in the main units

The outputs Y000 and Y001 are the high speed response type.
When using a pulse output instruction or positioning instruction, adjust the load current of the open collector transistor output.

When the load is smaller, connect a dummy resistor in parallel to the outside of a used output terminal (Y000 or Y001) as shown in the circuit diagram below so that the specified current shown below flows in the output transistor.



1) For FX3U, FX3UC and FX3G PLCs
   Operating voltage range : DC5 to 24V
   Operating current range : 10 to 160 mA
   Output frequency : 100 kHz or less

2) For FX2N and FX2NC PLCs
   Operating voltage range : DC5V
   Operating current range : 0.1A
   Output frequency : 20 kHz or less

   Operating voltage range : DC12 to 24V
   Operating current range : 0.1A
   Output frequency : 10 kHz or less

3) For FX1NC PLC
   Operating voltage range : DC5V
   Operating current range : 10 to 100 mA
   Output frequency : 10 kHz or less

   Operating voltage range : DC12 to 24V
   Operating current range : 50 to 100 mA
   Output frequency : 10 kHz or less

4) For FX1S and FX1N PLCs
   Even without connecting a dummy resistor, a pulse output of 100 kHz or less can be generated under 5 to 24 VDC (10 to 100 mA).

5) For FX0, FX0S, FX0N, FXU and FX2C PLCs
   Have a current of about 100 mA flow for the FX2C PLC and a current of about 200 mA flow for the FX0, FX0S, FX0N and FXU PLCs and extension.

8. Cautions on using special high speed output adapters (FX3U PLC)

1) Outputs of special high speed output adapters work as differential line drivers.
2) Set the pulse output type setting switch in a special high speed output adapter to the "pulse train + direction" (PLS - DIR) side.
If the switch is set to the "forward rotation pulse train - reverse rotation pulse train" (FP - RP) side, normal operation is disabled. The pulse output destination changes depending on the PLC output status as shown in the table below.

Pulse output destination	Output affecting operation	Operation
(d) = Y000	Y004	While Y004 is ON, pulses are output from Y000 in the high speed output adapter. While Y004 is OFF, pulses are output from Y004 in the high speed output adapter.
(d) = Y001	Y005	While Y005 is ON, pulses are output from Y001 in the high speed output adapter. While Y005 is OFF, pulses are output from Y005 in the high speed output adapter.

3) Set the pulse output type setting switch while the PLC is in STOP or while the power is OFF.
Do not manipulate the pulse output type setting switch while pulses are being output.
4) When special high speed output adapters are connected, the same output numbers in the main unit are assigned as shown in the table below.
Only wire the appropriate terminals. (Use either one only. Do not connect to the other terminal.)
Outputs of special high speed output adapters and main units operate as follows.

Assignment of output numbers in special high speed output adapters

Status of output type setting switch	Signal name	Setting name in each positioning instruction	Output number			
			1st unit		2nd unit	
			1st axis	2nd axis	3rd axis	4th axis
"FP - RP" side	Forward rotation pulse train (FP)	Pulse output destination	Y000	Y001	Y002	Y003
	Reverse rotation pulse train (RP)	Rotation direction signal	Y004	Y005	Y006	Y007
"PLS - DIR" side	Pulse train	Pulse output destination	Y000	Y001	Y002	Y003
	Direction	Rotation direction signal	Y004	Y005	Y006	Y007

Output operation

	Output operation
Relay output type main unit	While instruction is activated, relevant output is ON. (LED is also ON.) Use special high speed output adapter.
Special high speed output adapter	Operated (ON/OFF) Set the output frequency to "200 kHz" or less.
Transistor output type main unit	Operated (ON/OFF) Set the output frequency to "100 kHz" or less.

9. Others

1) When using the same output relay (Y000 or Y001) in several instructions.
While a pulse output monitor (BUSY/READY) flag is ON a pulse output instruction and positioning instruction for the same output relay cannot be executed.
While a pulse output monitor flag is ON even after the instruction drive contact is set to OFF, a pulse output instruction or positioning instruction for the same output relay cannot be executed.
Before executing such an instruction, wait until the pulse output monitor flag turns OFF and one or more operation cycles pass.
(Only the FX3U, FX3UC, FX3G, FX1N, FX1NC and FX1S PLCs are compatible with the pulse output monitor flags.)

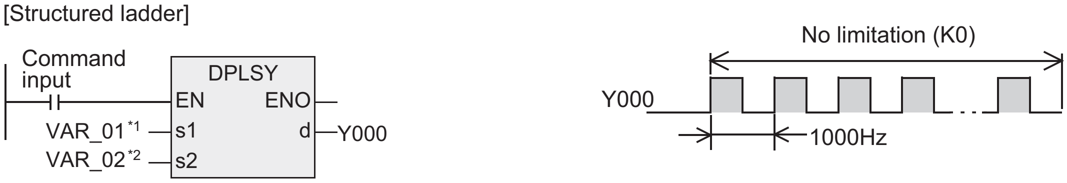
Pulse output destination device	Pulse output monitor flag		
	FX3U, FX3UC	FX3G	FX1N, FX1NC, FX1S
Y000	M8340	M8340, M8147	M8147
Y001	M8350	M8350, M8148	M8148

2) "Frequency control mode" in which DHSZ and DPLSY instructions are combined can be used only once in a program.

Program examples (When outputting pulses without any limitation)

When the device specified by (s2) is set to K0, pulses are output without any limitation.

[Structured ladder]



*1. VAR_01 is a global label and is defined as K1000.
*2. VAR_02 is a global label and is defined as K0.

[ST]

```
DPLSY(X000,VAR_01,VAR_02,Y000);
```

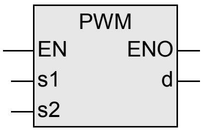
7.6.9 PWM

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This instruction outputs pulses with a specified period and ON duration.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
PWM	16 bits	Continuous		PWM(EN, s1, s2, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Pulse width (ms) data or word device storing the data	ANY16
	s2	Period data (ms) or word device storing the data	ANY16
Output variable	ENO	Execution state	Bit
	d	Device (Y) from which pulses are to be output	Bit

3. Applicable devices

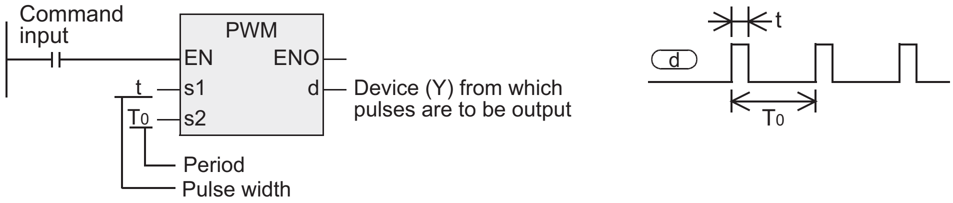
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
s2								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
d		▲1																	●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (PWM)

Pulses whose ON pulse width (ms) is specified by (s1) are output in periods (ms) specified by (s2) to the device specified by (s1).



- Specify the pulse width "t" in (s1).
  Allowable setting range: 0 to 32,767ms
- Specify the period "T0" in (s2).
  Allowable setting range: 1 to 32,767ms
- Specify the output (Y) number from which pulses are to be output in (d).
  Allowable setting range: Refer to "Cautions".

Cautions

1. Setting the pulse width and period

Make sure that the pulse width (s1) and period (s2) satisfy the relationship "(s1) ≤ (s2)".

2. Pulse output

1) Only the following outputs can be specified in (d) according to the system configuration.
   - For FX3U, FX3UC and FX3G PLCs
     - When using special high speed output adapters[*1]: Y000, Y001, Y002[*2], Y003[*2]
     - When using transistor outputs on the main unit (that is, when not using special high speed output adapters): Y000, Y001, Y002[*3]

     *1. Special high speed output adapters can be connected only to the FX3U PLC.
     When using the PWM instruction with a relay output type FX3U PLC, a special high speed output adapter is required.

     *2. When specifying Y002 or Y003 on a special high speed output adapter, a second special high speed output adapter is required.

     *3. "Y002" cannot be used for the 14- or 24-type FX3G PLC.
   - For FX1S, FX1N, FX2N, FX1NC and FX2NC PLCs
     Only Y000 or Y001 is valid (transistor output).
   - For FX0, FX0S and FX0N PLCs
     Only Y001 is valid (transistor output).
   - For FXU and FX2C PLCs
     All Ys are valid (transistor output).
2) The pulse output is controlled by interrupt processing not affected by the sequence program (operation cycle).
3) If the command input is set to OFF, the output from the device specified by (d) turns OFF.
4) While a pulse output monitor (BUSY/READY) flag is ON, a pulse output or positioning instruction for the same output relay cannot be executed.
   While a pulse output monitor flag is ON even after the instruction derive contact is set to OFF, a pulse output or positioning instruction for the same output relay cannot be executed.
   Before executing a pulse output or positioning instruction, wait until the pulse output monitor flag turns OFF and one or more operation cycles pass.
   (Only the FX3U, FX3UC and FX3G PLCs are compatible with the pulse output monitor flags.)

Pulse output destination device	Pulse output monitor flag
Y000	M8340
Y001	M8350
Y002	M8360
Y003	M8370

3. Restrictions to target devices

▲1: Refer to item 1 of "Cautions".
▲2: Applicable only to the FX3U, FX3UC and FX3G PLCs.
▲3: Applicable only to the FX3U and FX3UC PLCs.
("Y002" cannot be used for the 14- or 24-type FX3G PLC.)
(Special high speed output adapters can be connected only to the FX3U PLC.)

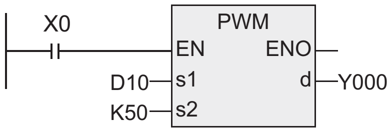
4. Cautions on using special high speed output adapters

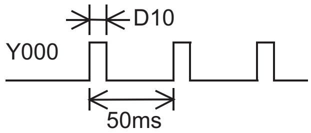
→ **Refer to item 7 in 7.6.9 "Cautions".**

Program examples

When the contents of D10 are changed in the range from "0" to "50" in the program example shown below, the average output from Y000 will be in the range from 0 to 100%.
When the contents of D10 exceed "50", it becomes an error.
In this program example, the FX3U series main unit (sink output) is used. For wiring details, refer to the manual of the PLC used.

[Structured ladder]

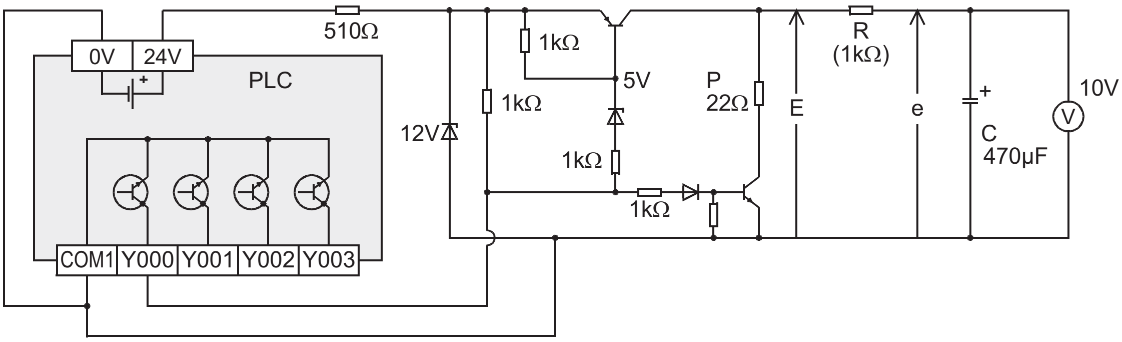




[ST]

```
PWM(X0,D10,K50,Y000);
```

Example of smoothing circuit



R>>P
τ=R(kΩ)×C(μF)=470ms>>To
The time constant τ of the filter should be considerably larger than the pulse cycle T0.

The ripple value "Δe" in the mean output current "e" is approximately "$\frac{\Delta e}{e} \leq \frac{To}{\tau}$".

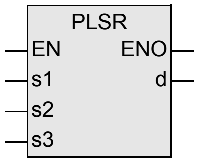
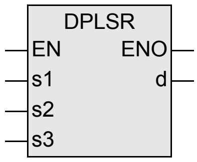
7.6.10 PLSR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

This pulse output instruction has the acceleration/deceleration function.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
PLSR	16 bits	Continuous		PLSR(EN, s1, s2, s3, d);
DPLSR	32 bits	Continuous		DPLSR(EN, s1, s2, s3, d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s1)	Maximum frequency data (Hz) or the word device storing the data	ANY16	ANY32
	(s2)	Total number of output pulses (PLS) or word device storing the data	ANY16	ANY32
	(s3)	Acceleration/deceleration time (ms) data or word device storing the data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Device (Y) from which pulses are to be output	Bit	

3. Applicable devices

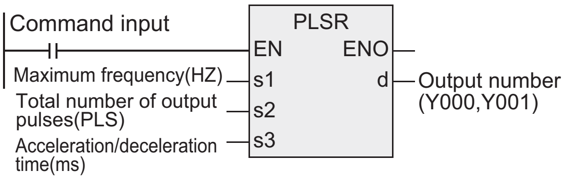
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit Specification				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
(s2)								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
(s3)								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
(d)		▲1																	●					

▲: Refer to "Cautions".

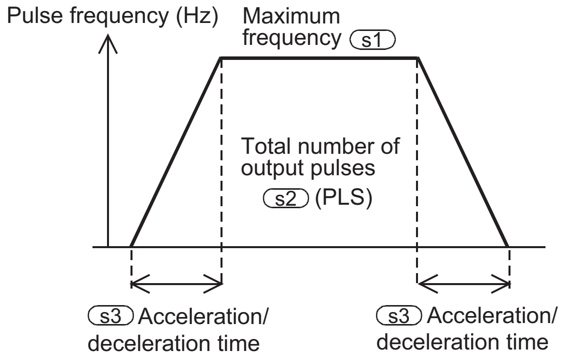
Function and operation explanation

1. 16-bit operation (PLSR)

Pulses are output from output (Y) specified by d by the number of output pulses specified by s2 with acceleration/deceleration to the maximum frequency specified by s1 over the time (ms) specified by s3.

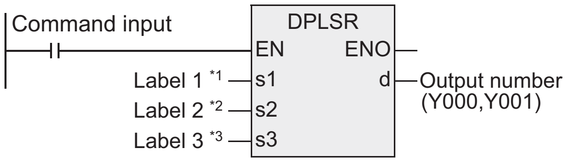


Refer to "Cautions" for setting s1, s2, s3 and d.



2. 32-bit operation (DPLSR)

Pulses are output from output (Y) specified by d by the number of output pulses specified by s2 with acceleration/deceleration to the maximum frequency specified by s1 over the time (ms) specified by s3.

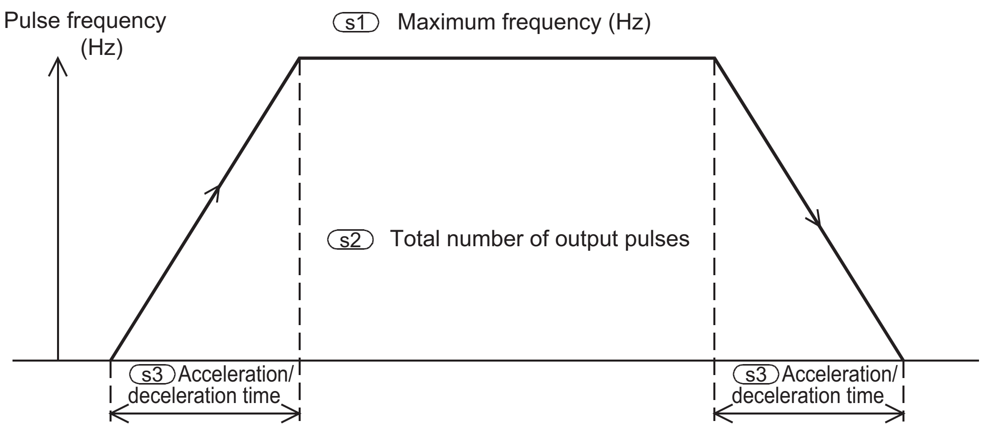


*1. This defines the maximum frequency (Hz).
*2. This defines the total number of output pulses (PLS).
*3. This defines the acceleration/deceleration time (ms).

Refer to "Cautions" for setting s1, s2, s3 and d.

3. Pulse output specifications

- Simple positioning (with the acceleration/deceleration function)
  The operation pattern is as shown below:



- Output processing
  The pulse output is controlled by the dedicated hardware regardless of the operation cycle.
- Data change while the instruction is executed
  Even if operands are overwritten while the instruction is executed, such changes are not reflected immediately. The changes become valid the next time the instruction is driven.

Related devices

1. Instruction execution complete flag

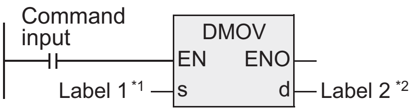
Device	Name	Description
M8029	Instruction execution complete	OFF : The command input is OFF, or pulses are being output. (This flag does not turn ON if the pulse output is interrupted in the middle of output.) ON : Output of the number of pulses set in (s2) is completed.

2. Monitoring the number of generated pulses

The number of pulses output from Y000 or Y001 is stored in the following special data registers:

Device		Description	Contents of data
High order	Low order		
D8141	D8140	Accumulated number of pulses output from Y000	Accumulated number of pulses output from Y000 by PLSY and PLSR instructions
D8143	D8142	Accumulated number of pulses output from Y001	Accumulated number of pulses output from Y001 by PLSY and PLSR instructions
D8137	D8136	Total accumulated number of pulses output from Y000 and Y001	Total accumulated number of pulses output from Y000 and Y001 by PLSY and PLSR instructions

The contents of each data register can be cleared using the following program.



*1. This defines K0.
*2. This defines the low order device in the table above.

3. How to stop the pulse output

- When the command input is set to OFF, the pulse generation is immediately stopped. When the command input is set to ON again, pulse generation operation restarts from the beginning.
- When the special auxiliary relays (M) shown below are set to ON, the pulse output is stopped.

Device			Description
FX3U, FX3UC	FX3G	FX1S, FX1N, FX1NC	
M8349	M8145, M8349	M8145	Immediately stops pulse output from Y000.
M8359	M8146, M8359	M8146	Immediately stops pulse output from Y001.

To restart pulse output again, set the device corresponding to the output number to OFF, and then drive the pulse output instruction again.

Cautions

1. Maximum frequency (s1)

When using transistor outputs on the main unit, set the maximum frequency specified by (s1) as follows.

- For FX3U and FX3UC PLCs: 10 to 100,000 Hz or less (200,000 Hz or less when using special high speed adapter.)
- For FX3G PLC : 10 to 100,000 Hz or less
- For FX2N and FX2NC PLCs : 10 to 20,000 Hz
- For FX1S and FX1N PLC : 10 to 100,000 Hz
- For FX1NC PLC : 10 to 10,000 Hz

2. Total number of output pulses (s2)

Set the total number of output pulses specified by (s2) as follows.

- For FX3U and FX3UC PLCs : 16-bit instruction→1 to 32,767PLS
  32-bit instruction→1 to 2,147,483,647PLS
- For FX3G PLC : 16-bit instruction→110 to 32,767PLS
  32-bit instruction→110 to 2,147,483,647PLS
- For FX2N and FX2NC PLCs : 16-bit instruction→110 to 32,767PLS
  32-bit instruction→110 to 2,147,483,647PLS
  When setting a value less than "110", the operation is as follows.
  PLC version V3.00 or before : The pulses are not generated normally.
  PLC version V3.00 or later : One tenth of the maximum frequency is generated.
  If the maximum frequency is 100 Hz or less, "10 Hz" is generated.
- For FX1S, FX1N and FX1NC PLCs : 16-bit instruction→110 to 32,767(PLS)
  32-bit instruction→110 to 999,999(PLS)
  When setting a value less than "110", the pulses are not generated normally.

3. Acceleration/deceleration time (s3)

Set the acceleration/deceleration time specified by (s3) as follows.
For FX3U, FX3UC and FX3G PLCs: 50 to 5000 (ms)
For FX2N and FX2NC PLCs: 5000 (ms) or less
Follow, however, the conditions a) through d).

a) Make the acceleration/deceleration time be 10 times or more the PLC's maximum scan time (value of D8012 or greater). If set to less than 10 times, the acceleration/deceleration timing becomes instable.

b) The following equation defines the allowable minimum that can be set as the acceleration/ deceleration time.

$$(s3) \geq \frac{90000}{(s1)} \times 5$$

If setting a value less than the equation indicates, the error in the acceleration time becomes large. If set to less than "90000/(s1), operation takes place while rounding it up to "90000/(s1)".

c) The following equation defines the allowable maximum that can be set as the acceleration/ deceleration time.

$$(s3) \leq \frac{(s2)}{(s1)} \times 818$$

d) The number of speed changes (steps) for the acceleration/deceleration is fixed to "10 times" as shown in the previous page.

If unable to set the PLC to these conditions, lower the maximum frequency specified by (s1).

- For FX1S, FX1N and FX1NC PLCs: 50 to 5000(ms)

4. Pulse output

- Only a transistor output on the main unit or Y000 or Y001 on a special high speed output adapter*1 can be specified in (d).

  *1. Special high speed output adapters can be connected only to the FX3U PLC.
  When using the PLSR instruction with a relay output type FX3U PLC, a special high speed output adapter is required.
- The duration of the ON/OFF pulses is 50% (ON = 50%, OFF = 50%)
- The pulse output is controlled by the dedicated hardware not affected by the sequence program (operation cycle).
- If the command input is set to OFF during continuous pulse output, the output from the device specified by (d) turns OFF.

5. Specifying input and output variables

When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

6. Restrictions to target devices

▲1: Refer to item 4 in "Cautions".
▲2: Applicable only to the FX3U, FX3UC and FX3G PLCs.
▲3: Applicable only to the FX3U and FX3UC PLCs.
(Special high speed output adapters can be connected only to the FX3U PLC.)

7. Handling of pulse output terminals in the main units of the FX3U, FX3UC and FX3G PLCs.

→ Refer to item 6 in 7.6.8 "Cautions".

8. Cautions on using special high speed output adapters

→ Refer to item 7 in 7.6.8 "Cautions".

9. Others

1) When using the same output relay (Y000 or Y001) in several instructions.
While a pulse output monitor (BUSY/READY) flag is ON a pulse output instruction and positioning instruction for the same output relay cannot be executed.
While a pulse output monitor flag is ON even after the instruction drive contact is set to OFF, a pulse output instruction or positioning instruction for the same output relay cannot be executed.
Before executing such an instruction, wait until the pulse output monitor flag turns OFF and one or more operation cycles pass.
(Only the FX3U, FX3UC, FX1N, FX1NC and FX1S PLCs are compatible with the pulse output monitor flags.)

Pulse output destination device	Pulse output monitor flag		
	FX3U, FX3UC	FX3G	FX1N, FX1N, FX1S
Y000	M8340	M8340, M8147	M8147
Y001	M8350	M8350, M8148	M8148

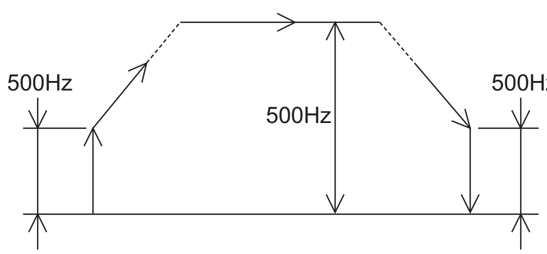
2) Minimum frequency for FX1S and FX1N PLCs
The output frequency of the PLSR instruction ranges from 10 to 100,000 Hz.
If the maximum speed or the speed during acceleration or deceleration goes out of this range, the value is automatically rounded up or down to a value within the range.
Note, however, that the following equation defines the minimum output frequency actually generated.

$$\sqrt{\text{maximum frequency}(\text{S1}\cdot)\ \text{Hz} \div (2 \times (\text{acceleration or deceleration time}(\text{S3}\cdot)\ \text{ms} \div 1000))} = \text{minimum frequency of output pulse frequency}$$

- Note that the frequency after the first step acceleration or the final frequency after deceleration does not go below the value calculated by the above equation.
[Example] Maximum speed: 50000Hz Acceleration/deceleration time: 100ms

$$\sqrt{50000 \div (2 \times (100 \div 1000))} = 500\text{Hz}$$

When setting 50000 Hz into the maximum frequency (S1·)
→ The actual output frequency is "500 Hz" at the first step of acceleration and the last step of deceleration.



7.7 Handy Instruction

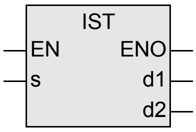
7.7.1 IST

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This is a command for controlling the initial state and special auxiliary relay automatically in a program by stepladder.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
IST	16 bits	Continuous		IST(E, s, d1, d2);

2. Set data

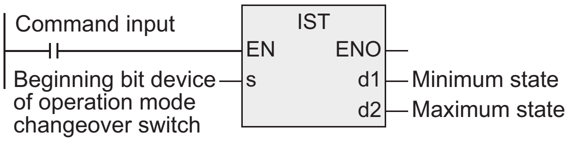
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Beginning bit device of operation mode changeover switch	Bit
Output variable	ENO	Execution state	Bit
	d1	Minimum state of practical state in automatic mode [d1<d2]	Bit
	d2	Maximum state of practical state in automatic mode [d1<d2]	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices													Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●	●	●				▲1												●					
d1						▲2													●					
d2						▲2													●					

▲: Refer to "Cautions".

Function and operation explanation



- In (s), designate beginning input of operation mode.
  Selection switch for operation mode occupies 8 points from the beginning device designated in (s), and the following switch functions are assigned individually.
  As shown in the table below, when X020 is assigned, X020 to X024 must be set in rotary switch so as not to be turned ON simultaneously.
  Wiring is not needed for switches not in use, but such switches cannot be used in other applications because they are occupied by IST command.

Source	Device No. (example)	Switch function
(s)	X020	Individual operation
(s)+1	X021	Return to home position
(s)+2	X022	Stepping
(s)+3	X023	One-cycle operation

Source	Device No. (example)	Switch function
(s)+4	X024	Continuous operation
(s)+5	X025	Return home start
(s)+6	X026	Automatic start
(s)+7	X027	Stop

- In (d1), designate minimum number of practical state (for automatic mode).
- In (d2), designate maximum number of practical state (for automatic mode).

1. Control of device by switch operation (occupied device)

When command input is turned ON, next device is automatically changed over and controlled. No change if command input is turned OFF.

Device No.	Operation function
M8040	Transfer ban
M8041*1	Transfer start
M8042	Start pulse
M8043*1	Return home end
M8045	All output reset ban
M8047*2	STL monitor valid

Device No.	Operation function
S0	Initial state of individual operation
S1	Initial state of return to home position
S2	Initial state of automatic operation

*1. Cleared when changed from RUN to STOP

*2. Process during END command execution

The following states should not be programmed as general states.

Device No.	Operation function
S0 to S9	Occupied for initial state. • S0 to S2 can be used for individual operation, return to home position, or automatic operation as specified above. • S3 to S9 can be used freely.
S10 to S19	Occupied for return to home position.

If return home end (M8043) is not ON, all outputs are OFF if changed over to individual (X020), return to home position (X021), or automatic (X022, X023, X024).
Automatic operation can be resumed after returning to home position completely.

→ **Before introduction, refer to "IST command introduction examples (examples of work transfer mechanism)" given below.**

Cautions

1. Devices designated in (S) and switches to be used

Not all mode selection switches are used.
Vacant numbers (not usable in other applications) are designated in switches not in use.

2. Program sequence of IST command and STL command

- IST command must be programmed prior to a series of STL circuits such as states S0 to S2.

3. State to be used in operation for return to home position

Use S10 to S19 for state for return to home position.
In final state for return to home position, after setting M8043, terminate by self-resetting.

4. Limit times of use of command

IST command can be programmed only once in the program.

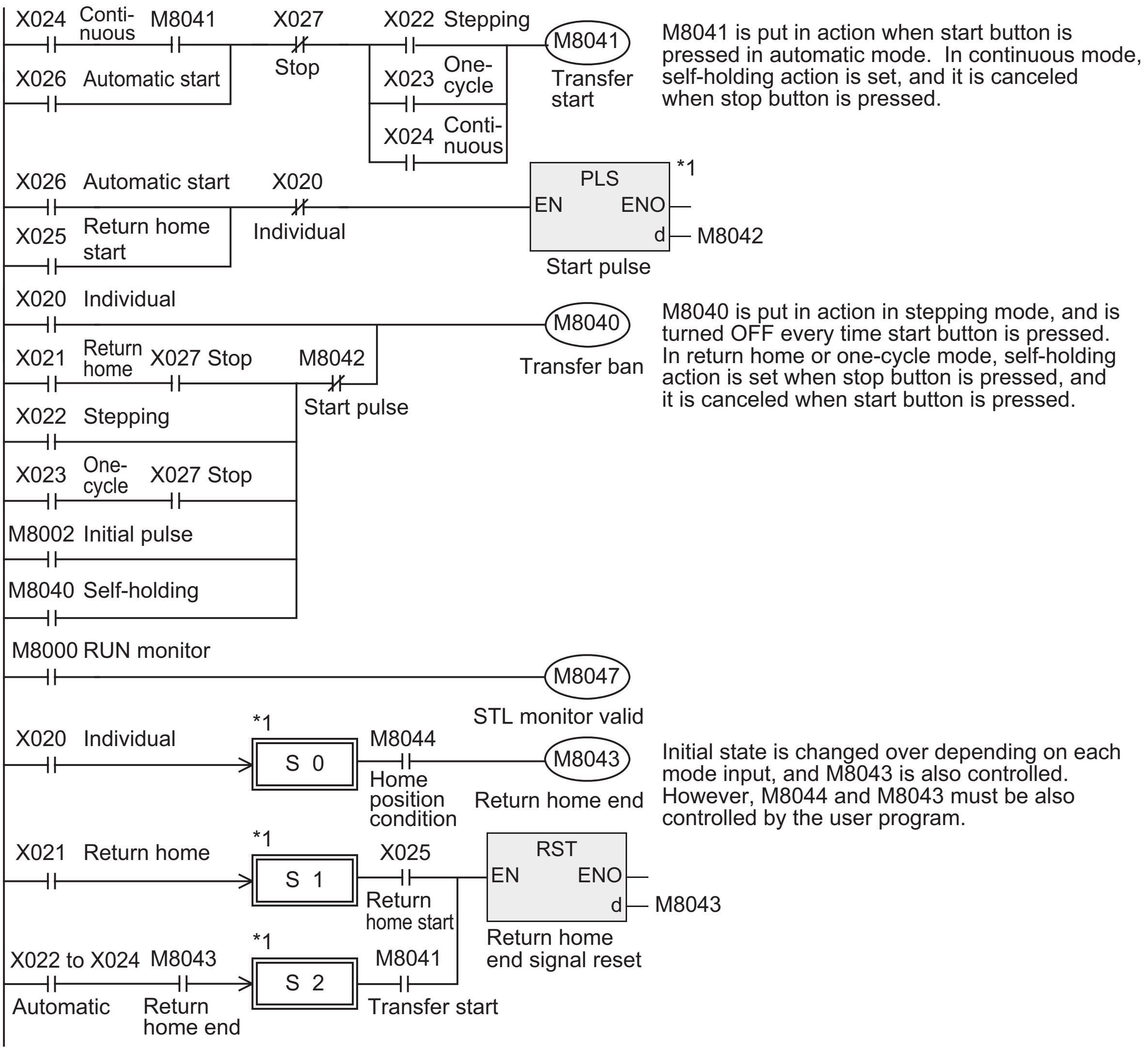
5. Object devices are restricted.

▲1: FX3U, FX3UC PLCs only are applicable.
However, index modifier (V, Z) is not applicable.
▲2: The device range is limited as follows by the PLC.
S20 to S899, 1000 to S4095 (FX3U, FX3UC, FX3G PLCs)
S20 to S899 (FX2N, FX2NC PLCs) (FXU, FX2C PLCs)
S20 to S999 (FX1N, FX1NC PLCs)
S20 to S127 (FX1S PLC) (FX0N PLC)
S20 to S63 (FX0, FX0S PLCs)

IST command equivalent circuit

Detail of special auxiliary relay (M) or initial state (S0 to S9) controlled automatically by IST command is as shown in the following equivalent circuit. (Read as reference knowledge.)
This equivalent circuit cannot create program.

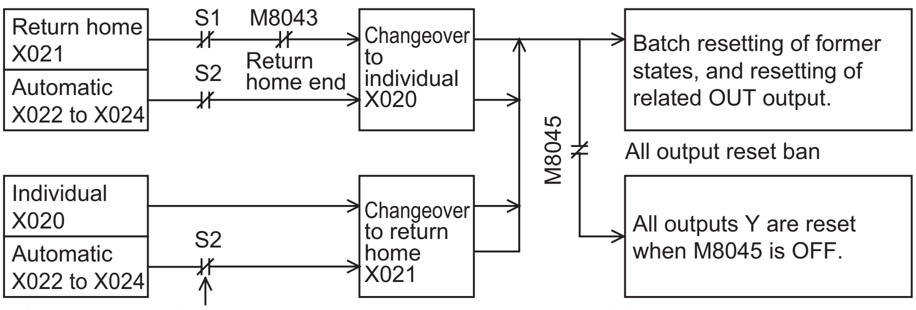
1. Equivalent circuit



*1. Equivalent circuit is presented for explanation, and it cannot be actually programmed as shown herein.

2. Changeover of operation mode

When the modes are changed over in individual, return home, and automatic modes, unless the machine is at home position, all outputs and former states are reset in batch. (When M8045 is driven, all outputs[*1] are not reset.)

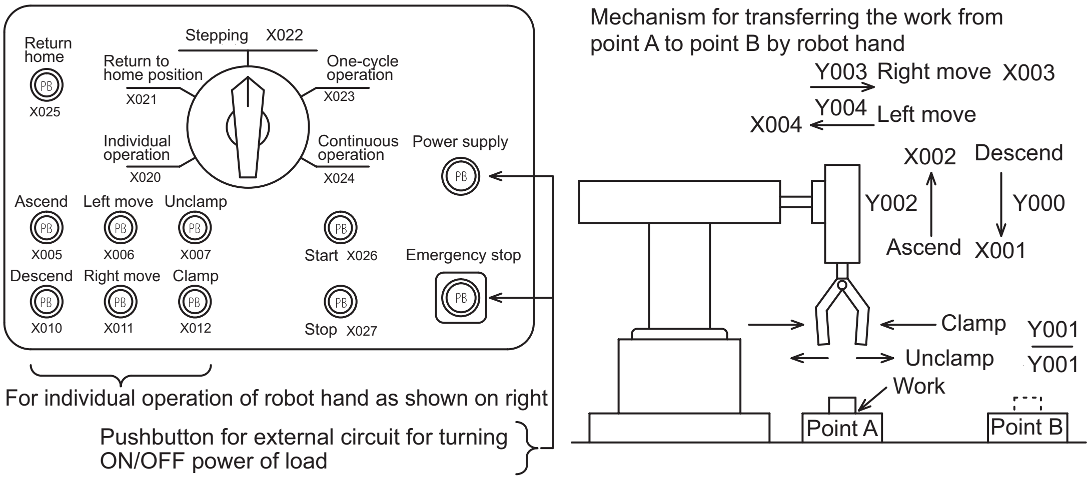


If changed over from automatic to return home during action of S2, the states other than initial state and outputs are not reset.

*1. All outputs: outputs not driven from the state of the device designated in (S) (Y), and outputs driven by OUT, SET commands from the state of the device designated in (S) (Y).

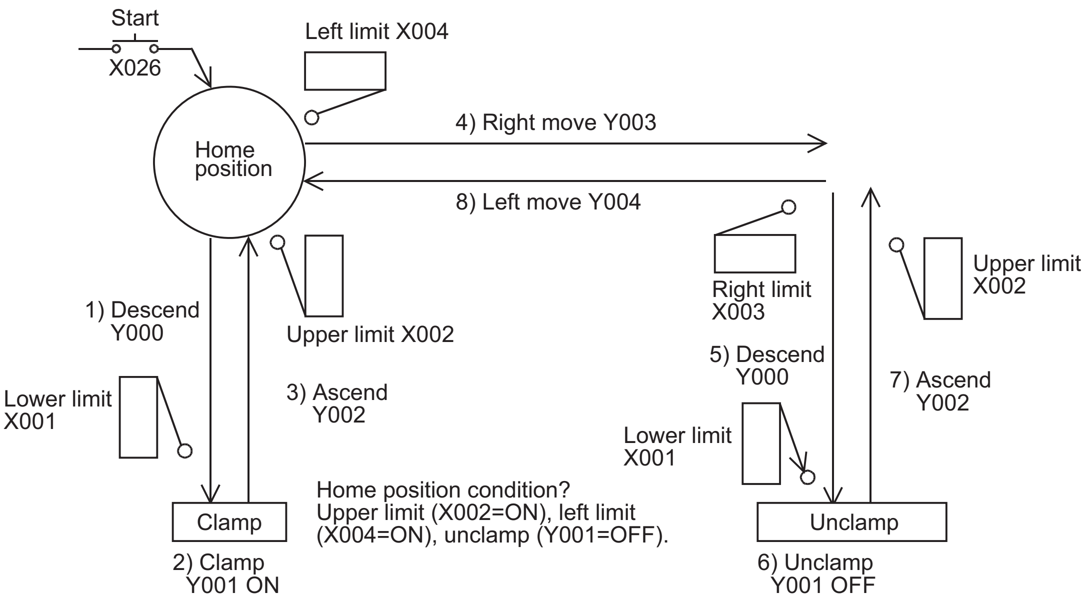
IST command introduction examples (examples of work transfer mechanism)

1. Operation mode



Operation mode		Operation content
Manual	Individual operation:	Mode for turning ON/OFF each load by individual pushbutton.
	Return to home position:	Mode for returning machine to home position automatically when return home pushbutton is pressed.
Automatic	Stepping:	To advance process by process every time start button is pressed.
	One-cycle operation:	When start button is pressed at home position, the machine is operated by one cycle automatically and stops at home position. If stop button is pressed on the way, the process stops immediately, and start button is pressed, the operation is resumed from the stopped position, and stops automatically at home position.
	Continuous operation:	When start button is pressed at home position, continuous repeating operation is started. When stop button is pressed, the machine operates to home position, and then stops.

2. Transfer mechanism



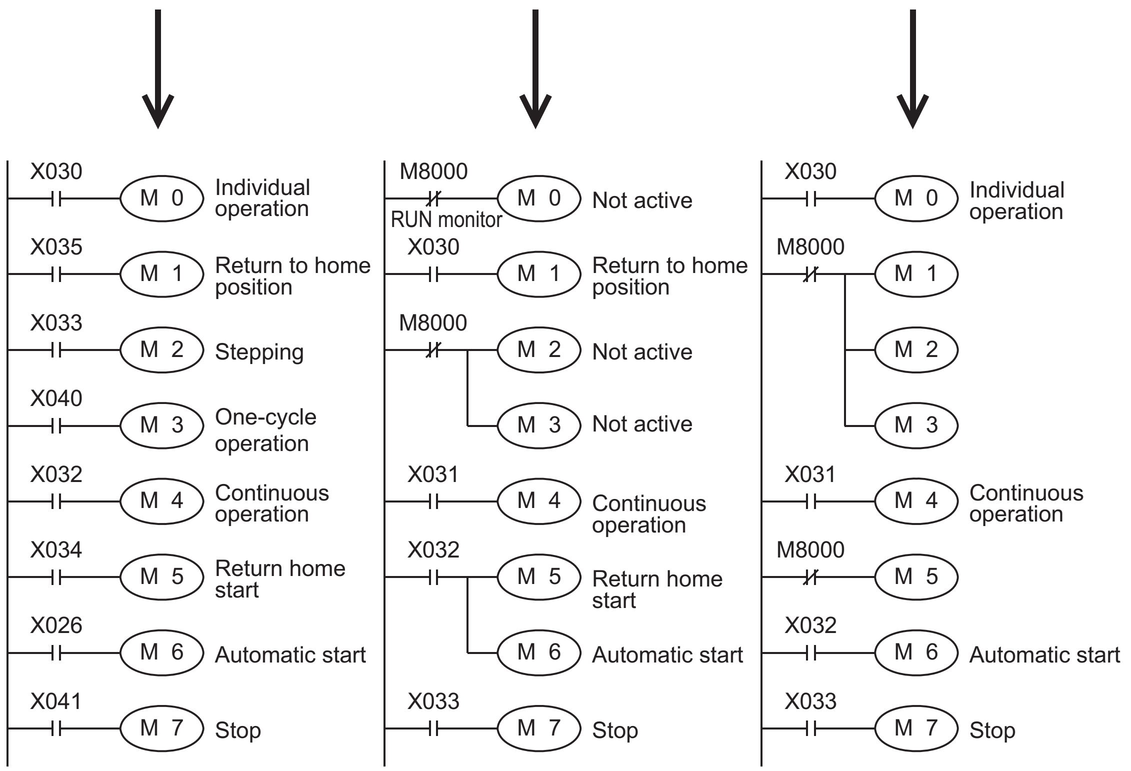
The home position is the upper left corner, and the work is sequentially transferred from left to right in the sequence of descend, clamp, ascend, right move, descend, unclamp, ascend, and left move.
Solenoid valve of double solenoid (two inputs: drive/non-drive) is used for descend/ascend, left move/right move, and solenoid of single solenoid (active only while energized) is used for clamp.

3. Assignment of mode select inputs

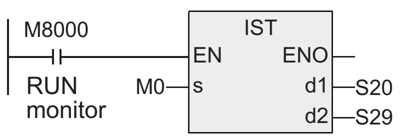
In order to use IST command, inputs of the following serial numbers must be assigned in the mode inputs. If the numbers are not serial, or modes are partly omitted, the array is modified by using auxiliary relays as shown below, which is used as the mode designation beginning input.

- X020: Individual operation
- X021: Return to home position
- X022: Stepping
- X023: One-cycle operation
- X024: Continuous operation
- X025: Return home start
- X026: Automatic start
- X027: Stop

When inputs are not serial numbers	In the case of continuous/return home mode only	In the case of continuous/ individual mode only
(Example) X030:Individual operation X035:Return to home position X033:Stepping X040:One-cycle operation X032:Continuous operation X034:Return home start X026:Automatic start X041:Stop	(Example) X030:Return to home position X031:Continuous operation X032:Common for automatic start and return home start X033:Stop	(Example) X030:Individual operation X031:Continuous operation X032:Automatic start X033:Stop



In this example, M0 is used as the mode designation beginning input.



4. Special auxiliary relays for IST command(M)

Auxiliary relays (M) used in IST command are divided into those controlled automatically by the command depending on the circumstances, and others that must be controlled by the program depending on the operation preparation or purpose of control.

1) Those controlled automatically by IST command

a) M8040: Transfer ban
When this auxiliary relay is put in action, transfer of all states is banned.

individual	: Always M8040 is active.
Return home, one-cycle	: When stop button is pressed, the operation is held until start button is pressed.
Stepping	: Always M8040 is active. However, only when start button is pressed, the operation is stopped, and is transferred.
Other	: Operation is held when PLC is changed from STOP to RUN, and is canceled when start button is pressed. In transfer ban state, the output in the state continues to operate.

b) M8041: Transfer start
This is the auxiliary relay required as transfer condition from initial state S2 to next state.

Individual, return home	: Not active.
Stepping, one-cycle	: Active only while start button is pressed.
Continuous	: Operation is held when start button is pressed, and canceled when stop button is pressed.

c) M8042: Start pulse
Active momentarily only when start button is pressed.

d) M8047: STL monitor valid
M8047 is turned ON when IST command is used.
STL monitor is valid when M8047 is ON, and active state numbers (S0 to S899) are stored in the special auxiliary relays D8040 to D8047 in numerical order.
As a result, up to eight operation state numbers can be monitored.
If any one of these states is active, the special auxiliary relay M8046 operates.

2) Those driven by sequence program

→ **For detail of these controls, see next page.**

a) M8043: Return home end
In the return home mode, when the machine reaches the home position, operate this special auxiliary relay (M) from the user side program.

b) M8044: Home position condition
You can drive this special auxiliary relay by detecting the home position condition of the machine. The signal is valid in all modes.

c) M8045: All output reset ban
When the modes are changed over in individual, return home, and automatic modes, unless the machine is at home position, all outputs and operation states are reset. However, when the M8045 is driven, only the operation states are reset.

5. Program examples

1) Circuit diagram

In the sequence circuit shown below, other portions than the shaded area are routine circuits. You can program the shaded area circuit according to the content of the control.

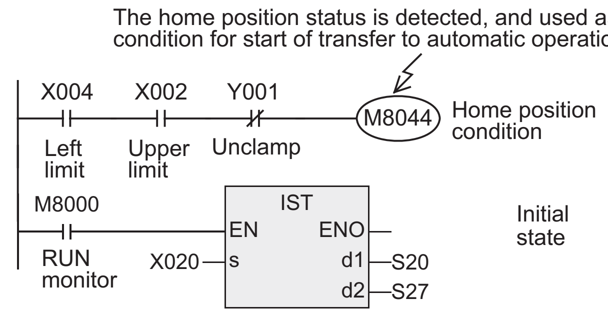
a) Initial circuit

During operation of the machine, the operation can be changed over freely within the "automatic operation' mode (stepping/one-cycle/continuous).

During operation of the machine, if changed over among "individual operation"/ "return home"/ "automatic operation", for safety precaution, all outputs are once reset, and the mode after changeover is valid.

(You cannot reset when all outputs reset ban M8045 is turned ON.)

[Structured ladder]



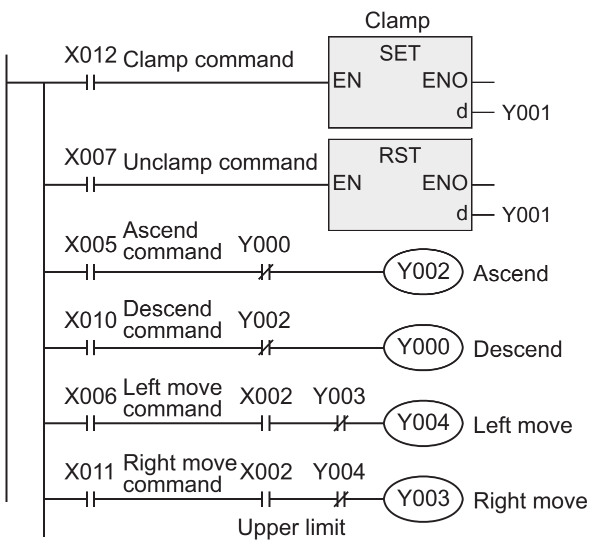
[ST]

```
M8044:= X004 AND X002 AND NOT Y001;
IST(M8000, X020, S20, S27);
```

b) Individual operation

Program is not required when individual mode is not available.

[Structured ladder]

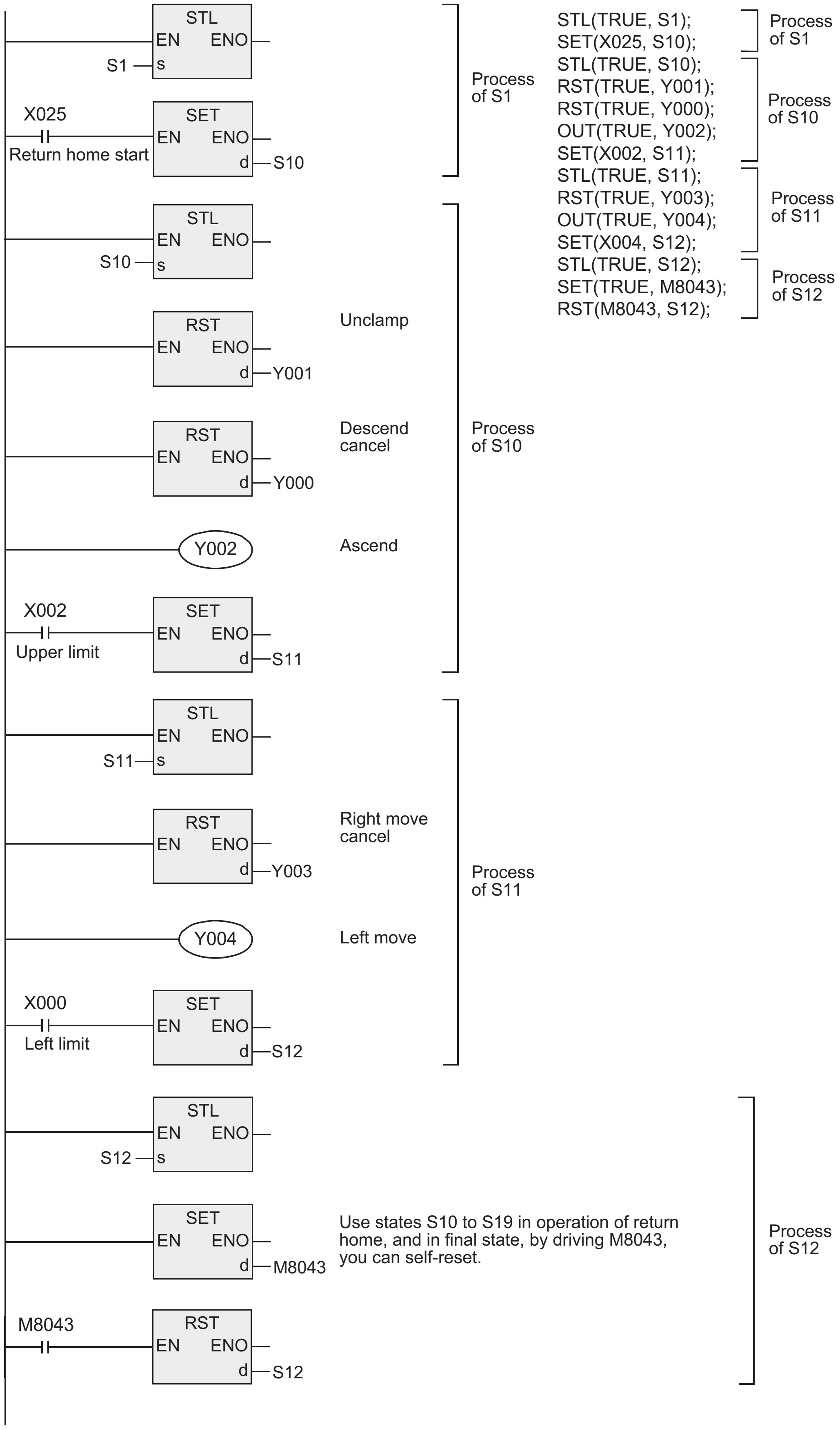


[ST]

```
SET(X012, Y001);
RST(X007, Y001);
Y002:= X005 AND NOT Y000;
Y000:= X010 AND NOT Y002;
Y004:= X006 AND X002 AND NOT Y003;
Y003:= X011 AND X002 AND NOT Y004;
```

c) Return to home position
Program is not required when return home mode is not available. However, before automatic operation, return home end M8043 must be once set.

[Structured ladder]

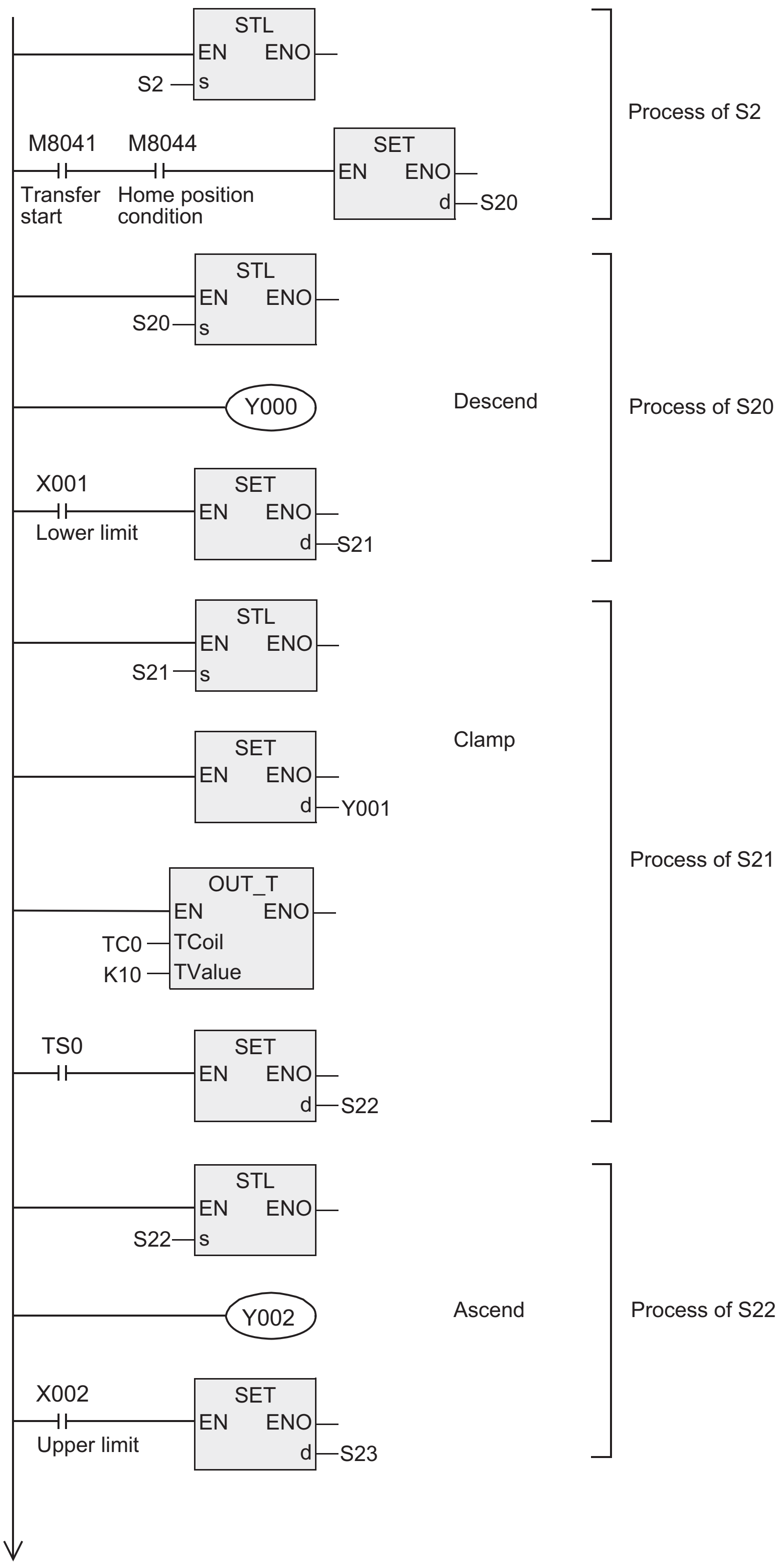



```
STL(TRUE, S1);
SET(X025, S10);
STL(TRUE, S10);
RST(TRUE, Y001);
RST(TRUE, Y000);
OUT(TRUE, Y002);
SET(X002, S11);
STL(TRUE, S11);
RST(TRUE, Y003);
OUT(TRUE, Y004);
SET(X004, S12);
STL(TRUE, S12);
SET(TRUE, M8043);
RST(M8043, S12);
```

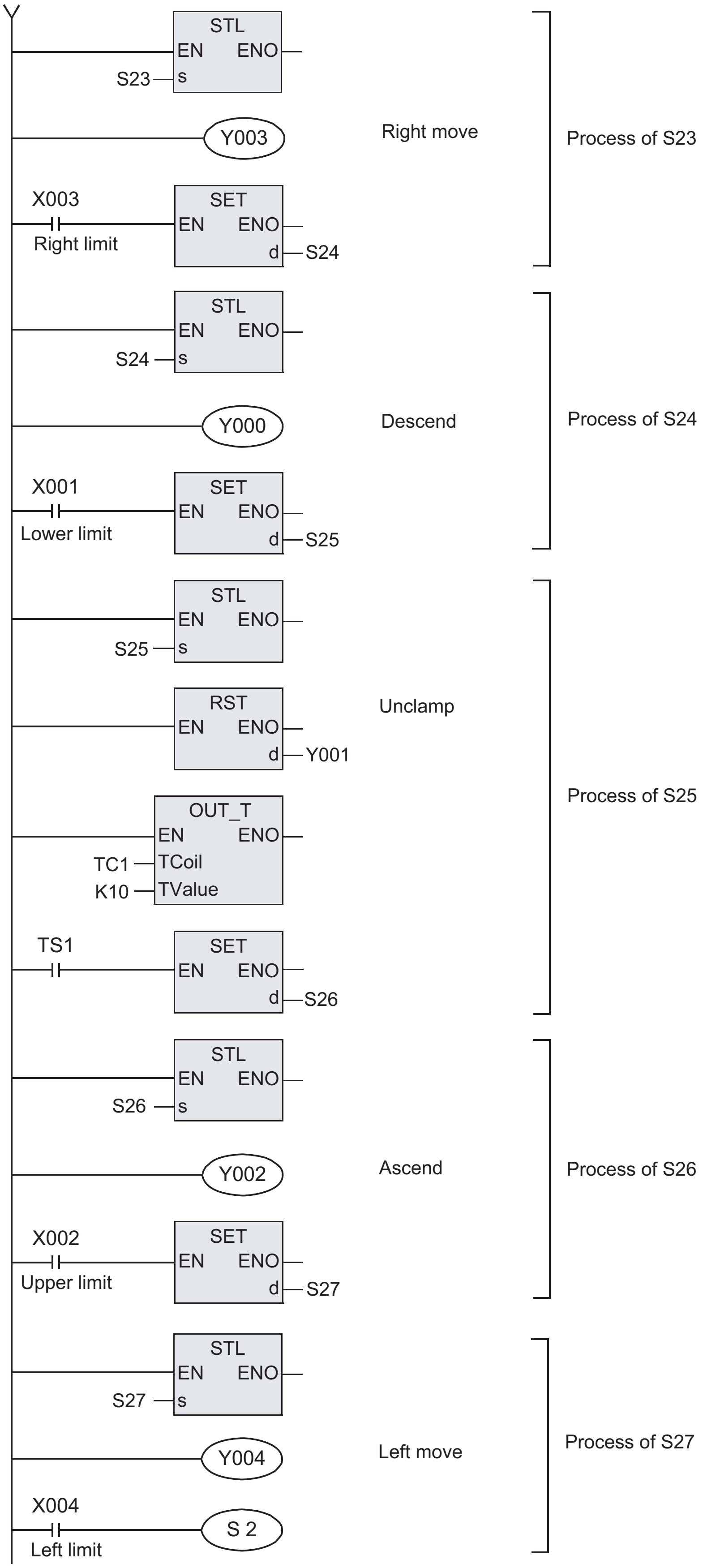
Code lines	Process
STL(TRUE, S1); SET(X025, S10);	Process of S1
STL(TRUE, S10); … SET(X002, S11);	Process of S10
STL(TRUE, S11); … SET(X004, S12);	Process of S11
STL(TRUE, S12); … RST(M8043, S12);	Process of S12

d) Automatic operation (stepping/one-cycle/continuous)

[Structured ladder]



[Structured ladder]



[ST]

```
STL(TRUE, S2);                                        ] Process of S2
IF(M8041 AND M8044) THEN SET(TRUE, S20);              ]
STL(TRUE, S20);                                       ]
OUT(TRUE, Y000);                                      ] Process of S20
SET(X001, S21);                                       ]
STL(TRUE, S21);                                       ]
SET(TRUE, Y001);                                      ] Process of S21
OUT_T(TRUE, TC0, K10);                                ]
SET(TS0, S22);                                        ]
STL(TRUE, S22);                                       ]
OUT(TRUE, Y002);                                      ] Process of S22
SET(X002, S23);                                       ]
STL(TRUE, S23);                                       ]
OUT(TRUE, Y003);                                      ] Process of S23
SET(X003, S24);                                       ]
STL(TRUE, S24);                                       ]
OUT(TRUE, Y000);                                      ] Process of S24
SET(X001, S25);                                       ]
STL(TRUE, S25);                                       ]
RST(TRUE, Y001);                                      ] Process of S25
OUT_T(TRUE, TC1, K10);                                ]
SET(TS1, S26);                                        ]
STL(TRUE, S26);                                       ]
OUT(TRUE, Y002);                                      ] Process of S26
SET(X002, S27);                                       ]
STL(TRUE, S27);                                       ]
OUT(TRUE, Y004);                                      ] Process of S27
OUT(X004, S2);                                        ]
```

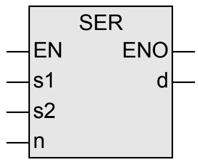
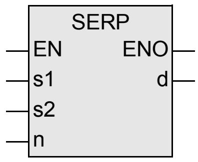
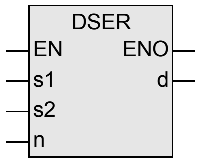
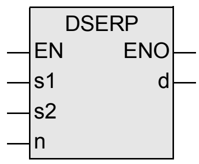
7.7.2 SER

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	△	×	×

Outline

This is a command for searching same data and maximum value, minimum value from the table of data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SER	16 bits	Continuous		SER(E, s1, s2, n, d);
SERP	16 bits	Pulse		SERP(EN, s1, s2, n, d);
DSER	32 bits	Continuous		DSER(EN, s1, s2, n, d);
DSERP	32 bits	Pulse		DSERP(EN, s1, s2, n, d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Beginning device for searching same data, maximum value, minimum value	ANY16	ANY32
	s2	Value for searching same data, maximum value, minimum value, or its storage destination device	ANY16	ANY32
	n	The number for searching same data, maximum value, minimum value	ANY16	
Output variable	ENO	Execution state	Bit	
	d	Beginning device for storing the numbers after searching same data, maximum value, minimum value	ARRAY [1..5] OF ANY16	ARRAY [1..5] OF ANY32

3. Applicable devices

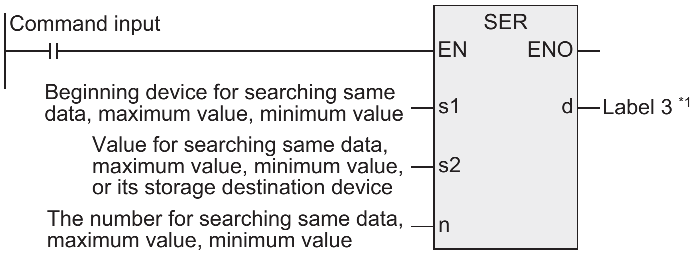
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲1	▲2			●					
s2								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
d									●	●	●	●	●	●	▲1	▲2			●					
n														●	▲1					●	●			

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation

In n pieces of data beginning from the device designated in (s1), same data as the device designated in (s2) is searched, and the result is stored in the device designated in (d).



*1. This is to define the beginning device for storing the numbers after searching same data, maximum value, minimum value.

1) Content and result of searched data
   a) When same data is found
      In five devices beginning from the device designated in (d), the number of same data, initial/final position, and positions of maximum value and minimum value are stored.
   b) When same data is not found
      In five devices beginning from the device designated in (d), the number of same data, initial/final position, and positions of maximum value and minimum value are stored.
      However, 0 is stored in three devices beginning from the device designated in (d) (number of same data, initial/final position).

2) Operation examples

a) Examples of composition of search result table and data

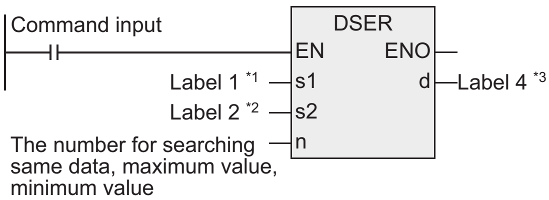
Searched device s1	Value of s1 of searched data (ex.)	Value of s2 of comparative data (ex.)	Position of data	Search result		
				Maximum value d +4	Coincide d	Minimum value d +3
s1	K100	K100	0		✓(Initial)	
s1 +1	K111		1			
s1 +2	K100		2		✓	
s1 +3	K 98		3			
s1 +4	K123		4			
s1 +5	K 66		5			✓
s1 +6	K100		6		✓(Final)	
s1 +7	K 95		7			
s1 +8	K210		8	✓		
s1 +9	K 88		9			

b) Search result table

Device No.	Content	Search result item
d	3	Number of same data
d +1	0	Position of same data (initial)
d +2	6	Position of same data (final)
d +3	5	Final position of minimum value
d +4	8	Final position of maximum value

2. 32-bit operation(DSER, DSERP)

In n pieces of data beginning from the device designated in s1, same data as the device designated in s2 is searched, and the result is stored in the device designated in d.



*1. This is to define the beginning device for searching same data, maximum value, minimum value.
*2. This is to define the value of searching same data, maximum value, minimum value, or its storing destination device.

1) Content and result of searched data

a) When same data is found

In five points of 32-bit data beginning from the device designated in d, the number of same data, initial/final position, and positions of maximum value and minimum value are stored.

b) When same data is not found

In five points of 32-bit data beginning from the device designated in d, the number of same data, initial/final position, and positions of maximum value and minimum value are stored.

However, 0 is stored in three points of 32-bit data beginning from the device designated in d (number of same data, initial/final position).

2) Operation examples

a) Examples of composition of search result table and data

Searched device (s1)+1	Value of (s1) of searched data (ex.)	Comparative data (s2)	Position of data	Search result		
				Maximum value (d)+4	Coincide (d)	Minimum value (d)+3
[(s1)+1, (s1)]	K100000	K100000	0		✓(Initial)	
[(s1)+3, (s1)+2]	K110100		1			
[(s1)+5, (s1)+4]	K100000		2		✓	
[(s1)+7, (s1)+6]	K 98000		3			
[(s1)+9, (s1)+8]	K123000		4			
[(s1)+11, (s1)+10]	K 66000		5			✓
[(s1)+13, (s1)+12]	K100000		6		✓(Final)	
[(s1)+15, (s1)+14]	K 95000		7			
[(s1)+17, (s1)+16]	K910000		8	✓		
[(s1)+19, (s1)+18]	K910000		9	✓		

b) Search result table

Device No.	Content	Search result item
[(d)+1, (d)]	3	Number of same data
[(d)+3, (d)+2]	0	Position of same data (initial)
[(d)+5, (d)+4]	6	Position of same data (final)
[(d)+7, (d)+6]	5	Final position of minimum value
[(d)+9, (d)+8]	9	Final position of maximum value

Cautions

1. Comparison of magnitude

To be calculated algebraically. (-10 < 2)

2. When minimum value and maximum value are present in a plurality

If there are a plurality of minimum value and maximum value in the data, the latter position is stored individually.

3. Number of bits occupied

The search result occupies the following number of devices when this command is driven.
Please be careful not to overlap with the devices used in the control of the machine.

1) In the case of 16-bit operation
   A total of five points will be occupied, that is, (d), (d)+1, (d)+2, (d)+3, (d)+4.

2) In the case of 32-bit operation
   A total of ten points will be occupied, that is, ((d)+1, (d)), ((d)+3, (d)+2), ((d)+5, (d)+4), ((d)+7, (d)+6), ((d)+9, (d)+8).

4. Designation of input and output variables

When handling array data or 32-bit data in structured program, you cannot designate the 16-bit device directly unlike simple project. Please use the label when handling array data or 32-bit data.
However, with the 32-bit counter, you can designate the 32-bit data directly because it is a device of 32 bits long.
When designating a device, use the global label.

5. FXU PLC supports the command by V3.07 or higher.

6. Object devices are limited.

▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

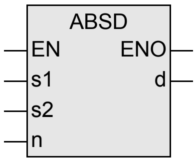
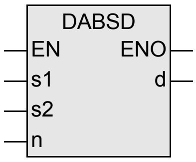
7.7.3 ABSD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This is a command for creating multiple output patterns corresponding to the present value of the counter.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ABSD	16 bits	Continuous		ABSD(EN, s1, s2, n, d);
DABSD	32 bits	Continuous		DABSD(EN, s1, s2, n, d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Beginning device for storing table data (rise point, fall point)	ANY16	ANY16
	s2	Counter for present value monitor to compare with table data	ANY16	ANY16
	n	Number of lines of table and number of points of bit device to be output	ANY16	
Output variable	ENO	Execution state	Bit	
	d	Beginning bit device to be output	Bit	

3. Applicable devices

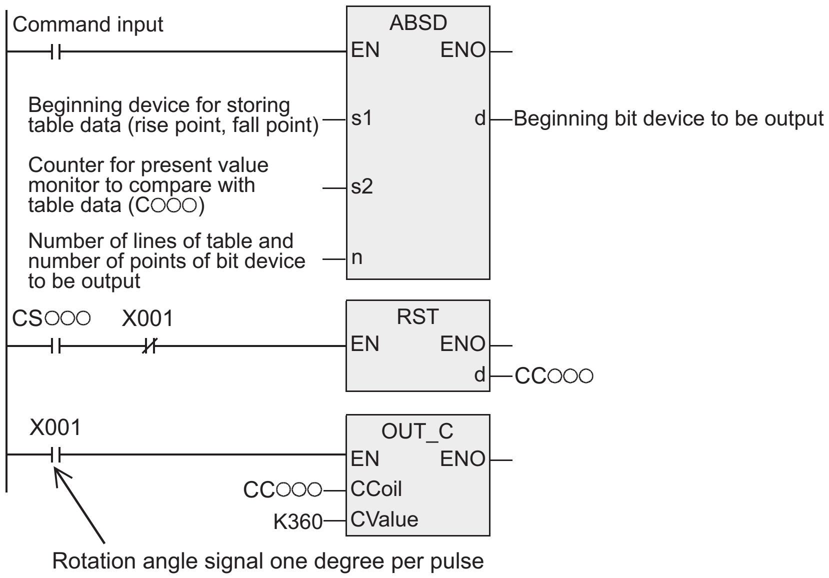
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲2	▲3			●					
s2													●						●					
d		●	●			●	▲1												●					
n																				●	●			

▲: Refer to "Cautions".

IST command equivalent circuit

1. 16-bit operation(ABSD)

This is an example for explaining the ON/OFF control of the output by one revolution of table (0 to 360 degrees).
(Rotation angle signal one degree per pulse)
Data table (occupying n lines × 2 points) of n lines from the device designated in (s1) is compared with the present value of the counter of the device designated in (s2), and outputs of n points continuous from the device designated in (d) are controlled to be ON/OFF during one revolution.

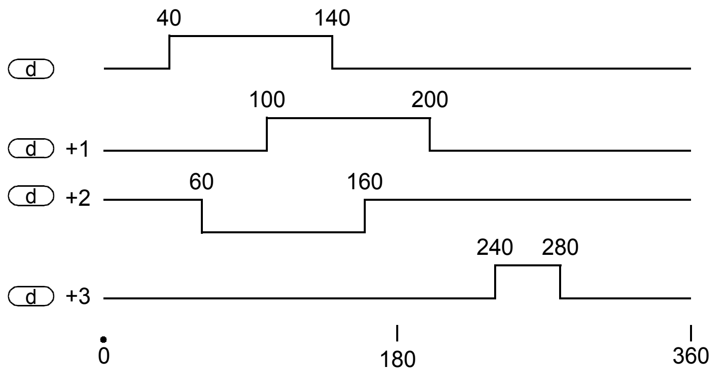


1) Write the following data in (s1) to (s1)+2n+1, preliminarily by using transfer command.

Rise point		Fall point		Object output
	Data value (example)		Data value (example)	
(s1)	40	(s1)+1	140	(d)
(s1)+2	100	(s1)+3	200	(d)+1
(s1)+4	160	(s1)+5	60	(d)+2
(s1)+6	240	(s1)+7	280	(d)+3
⋮	-	⋮	-	⋮
(s1)+2n		(s1)+2n+1		(d)+n-1

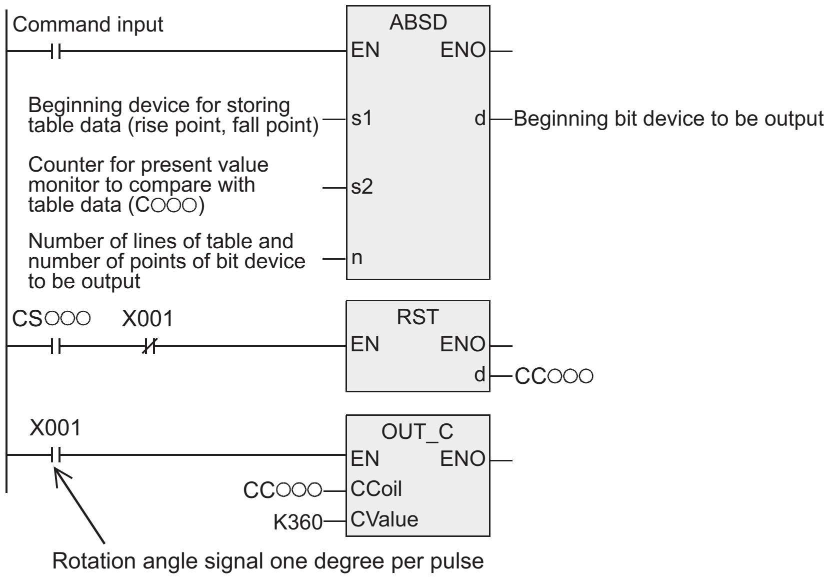
For example, rise point data is stored in even-number device, and fall point data in odd-number device, by 16-bit data.

2) Output pattern
When command input is turned ON, n points are changed as follows, starting from the device designated in (d).
Rise point and fall point may be individually changed by rewriting data in (s1) to (s1)+n×2.



2. 32-bit operation(DABSD)

This is an example for explaining the ON/OFF control of the output by one revolution of table (0 to 360 degrees).
(Rotation angle signal one degree per pulse)
Data table (occupying n lines × 4 points) of n lines from the device designated in (s1) is compared with the present value of the counter of the device designated in (s2), and outputs of n points continuous from the device designated in (d) are controlled to be ON/OFF.

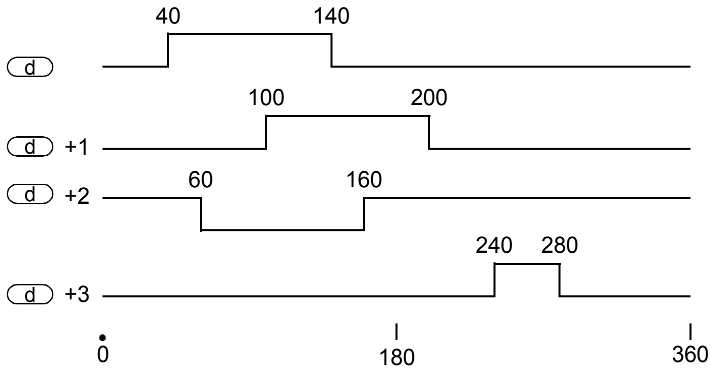


1) Write the following data in ((s1), (s1)+1) to ((s1)+4n+2, (s1)+4n+3), preliminarily by using transfer command.

Rise point	Data value (example)	Fall point	Data value (example)	Object output
[(s1)+1, (s1)]	40	[(s1)+3, (s1)+2]	140	(d)
[(s1)+5, (s1)+4]	100	[(s1)+7, (s1)+6]	200	(d)+1
[(s1)+9, (s1)+8]	160	[(s1)+11, (s1)+10]	60	(d)+2
[(s1)+13, (s1)+12]	240	[(s1)+15, (s1)+14]	280	(d)+3
⋮	-	⋮	-	⋮
[(s1)+4n+1, (s1)+4n]		[(s1)+4n+3, (s1)+4n+2]		(d)+n-1

For example, rise point data is stored in even-number device, and fall point data in odd-number device, by 32-bit data.

2) Output pattern
When command input is turned ON, n points are changed as follows, starting from the device designated in (d).
Rise point and fall point may be individually changed by rewriting data in ((s1)+1, (s1)) to ((s1)+(n×2)+3, (s1)+(n×2)+2).



Cautions

1. Designation of high speed counter

DABSD command can designate a high speed counter in the device designated in (s2).
In this case, as compared with the counter present value, the output pattern may have a response delay due to scan cycle.
When using FX3U, FX3UC PLCs, the response may be enhanced by using table high speed comparison function by DHSZ command or by using DHSCT command.

2. When designating the digits of bit device in (s1)

1) Device No.
   Designate a multiple of 16 (0, 16, 32, 64, ...).

2) Number of digits
   - K4 only in the case of ABSD (16-bit operation)
   - K8 only in the case of DABSD (32-bit operation)

3. Other cautions

- The number of output points of the object is determined by the value of n. ($1 \leq n \leq 64$)
- The output is not changed if the command input is turned OFF.

4. FXU PLC does not support 32-bit command at V2.30 or lower.

5. Object devices are limited.

▲1: FX3U, FX3UC PLCs only are applicable.
However, index modifier (V, Z) is not applicable.
▲2: FX3U, FX3UC, FX3G PLCs only are applicable.
▲3: FX3U, FX3UC PLCs only are applicable.

7.7.4 INCD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This is a command for creating multiple output patterns by using a pair of counters.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
INCD	16 bits	Continuous	INCD EN ENO s1 d s2 n	INCD(EN, s1, s2, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Beginning word device for storing the set value	ANY16
	s2	Beginning device of counter for present value monitor (2 points occupied)	ANY16
	n	Number of bit devices to be output	ANY16
Output variable	ENO	Execution state	Bit
	d	Beginning bit device to be output (n points occupied)	Bit

3. Applicable devices

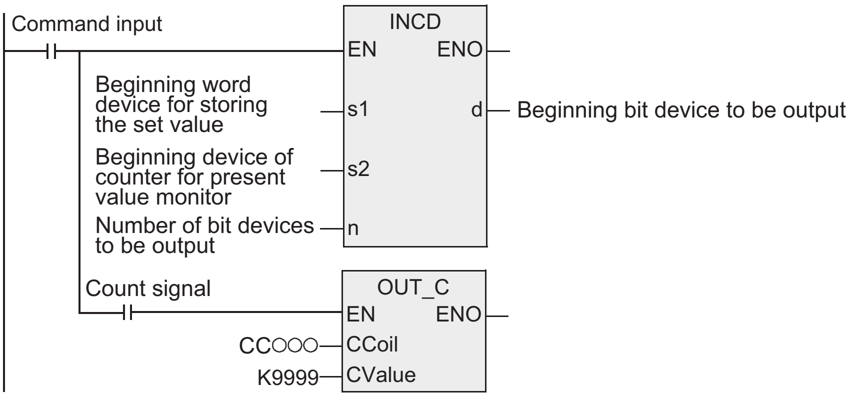
Operand type	Bit Devices							Word Devices													Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲2	▲3			●					
s2													●						●					
d		●	●			●	▲1												●					
n																				●	●			

▲: Refer to "Cautions".

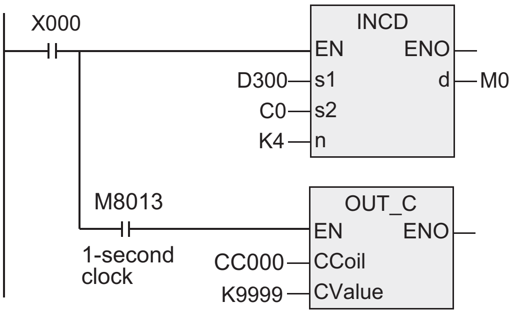
Function and operation explanation

1. 16-bit operation(INCD)

Data table (n lines × 1 point occupied) of n lines from the device designated in (s1) is compared with the present value of the counter of the device designated in (s2), and by resetting when coinciding, outputs are sequentially controlled to be ON/OFF.



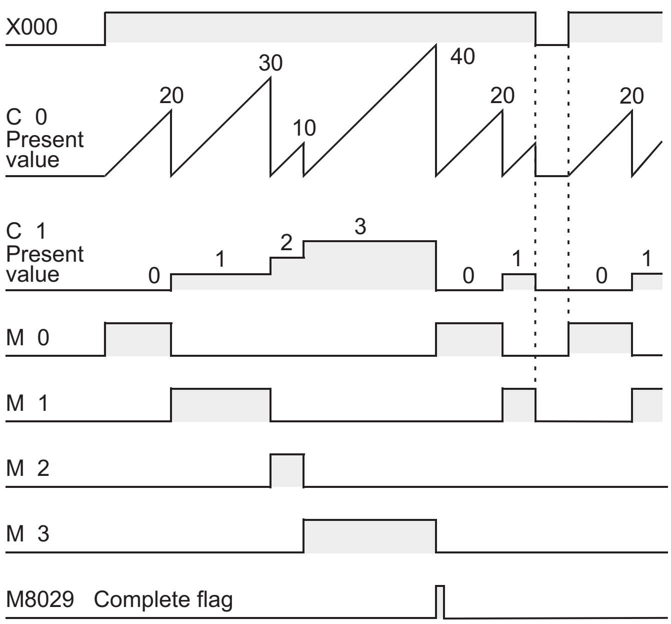
Operation



1) Timing chart

Write the following data beforehand by using transfer command.

Store device		Output	
	Data value (example)		Example
(s1)	D300=20	(d)	M0
(s1) +1	D301=30	(d) +1	M1
(s1) +2	D302=10	(d) +2	M2
(s1) +3	D303=40	(d) +3	M3
⋮	⋮	⋮	⋮
(s1) +n-1	-	(d) +n-1	-



2) M0 output is turned ON when command contact is turned ON.
3) Output (M0) is reset when the present value of C0 reaches comparative value D300, and the count value of the process counter C1 is incremented by +1, and the present value of the counter C0 is also reset.
4) Next output M1 is turned ON.
5) Output M1 is compared with comparative value D301 of C0 present value, and when reaching the comparative value, the count value of the process counter C1 is incremented by +1, and the present value of the counter C0 is also reset.
6) Similarly compared up to the number of points designated in n(K4). ($1 \leq n \leq 64$)
7) When the final process designated in n is over, execution complete flag M8029 is turned ON for one operation period.
   M8029 is a command execution complete flag used by a plurality of commands, and is used as the contact right after the command, and then you can set up a complete flag exclusively for this command.
8) Repeat outputs by returning to the beginning.

Cautions

1. When designating the digits of bit device in the device designated in (s1)
Designate a multiple of 16 (0, 16, 32, 64, ...) in the device number.

2. Object devices are limited.
▲1: FX3U, FX3UC PLCs only are applicable.
However, index modifier (V, Z) is not applicable.
▲2: FX3U, FX3UC, FX3G PLCs only are applicable.
▲3: FX3U, FX3UC PLCs only are applicable.

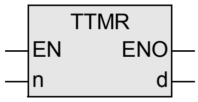
7.7.5 TTMR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This is a command for measuring the ON duration of TTMR command.
This is used when adjusting the timer setting time by pushbutton.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
TTMR	16 bits	Continuous		TTMR(EN, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	n	Multiplying factor number to be applied to teaching data	ANY16
Output variable	ENO	Execution state	Bit
	d	Device for storing teaching data (2 points occupied)	ANY16

3. Applicable devices

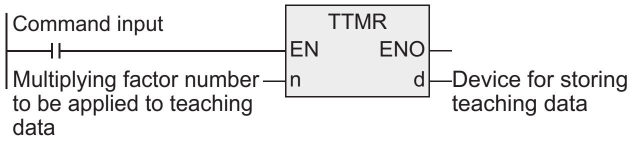
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
d														●	▲1				●					
n														●	▲1					●	●			

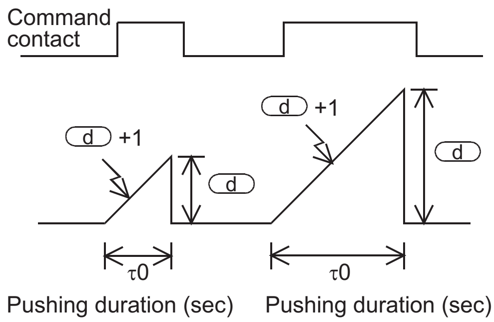
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(TTMR)

The pushing duration of command input (pushbutton) is measured in the unit of seconds, and is multiplied by multiplying factor (10^n), and transferred to the device designated in d.





The time to be transferred to the device designated in d depends on the multiplying factor of n, supposing the pushing duration to be $\tau 0$ (1 sec unit), and the value of the device actually designated in d is as follows.

n	Multiplying factor	d
K0	$\tau 0$	d ×1
K1	$10\tau 0$	d ×10
K2	$100\tau 0$	d ×100

Related command

You can use such a convenient command.

Command	Content
HOUR	The command to issue the ON time of HOUR command by the time added and designated in 1 hour unit.

Cautions

1. When command contact is turned OFF

The present value (d +1) of the pushing duration is reset, and the teaching time d is not changed.

2. Number of bits occupied

Two devices are occupied starting from the device (teaching time) designated in d.
Be careful not to overlap with the device used in control of the machine.

- d: Teaching time
- d +1: Present time of pushing duration

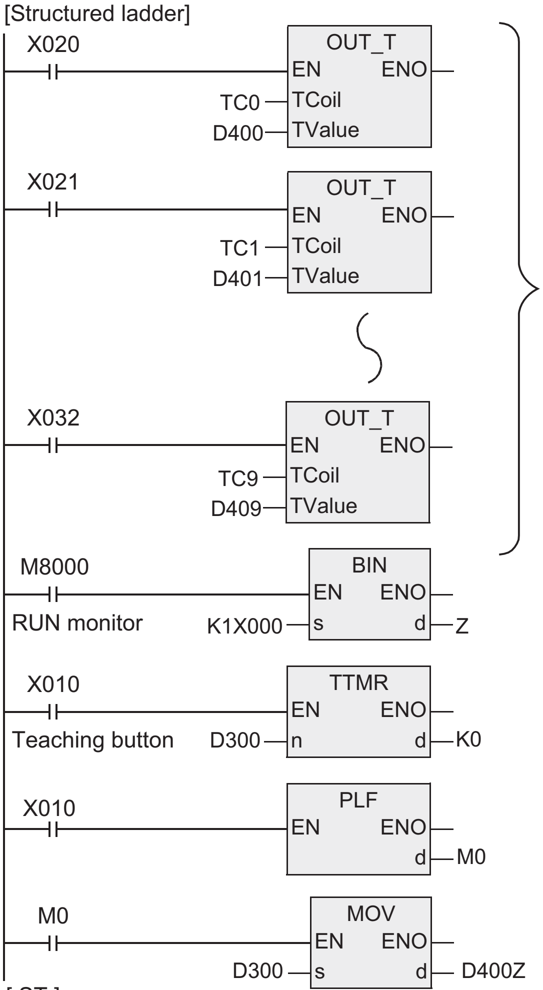
3. Object devices are limited.

▲1: FX3U, FX3UC PLCs only are applicable.

Program examples

Write teaching time in ten data registers.
Set values should be preliminarily written in D400 to D409.

[Structured ladder]



<u>Ten timers desired to be set</u>
Timers (T0 to T9) are 100 ms timers, and 1/10 of the teaching data is the actual operation time (in seconds).

<u>Timer selection by digital switch</u>
The input of one-digit digital switch connected to X000 to X003 is converted to BIN, and transferred to Z.

<u>Teaching measurement</u>
Pushing duration (seconds) of X010 is stored in D300.

Teaching button restore detection

<u>Writing set value of timer</u>
The teaching time (D300) is transferred to the setting register (D400Z) for the timer selected by digital switch.

[ST]

```
OUT_T (X020, TC0,D400);
OUT_T (X021, TC1, D401);
        ⁓
OUT_T (X032, TC9, D409);
BIN(M8000, K1X000, Z);
TTMR(X010, D300, K0);
PLF(X010, M0);
MOV(M0, D300, D400Z);
```

7.7.6 STMR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This is the command for creating the off-delay timer, one-shot timer or flicker timer easily.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
STMR	16 bits	Continuous	STMR block: inputs EN, s, m; outputs ENO, d	STMR(EN, s, m, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Timer to be used [100ms timer]	ANY16
	m	Set value of timer	ANY16
Output variable	ENO	Execution state	Bit
	d	Beginning bit device to be output [4 points occupied]	Bit

3. Applicable devices

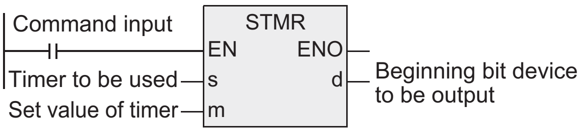
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s												●							●					
m														●	▲2					●	●			
d		●	●			●	▲1												●					

▲: Refer to "Cautions".

Function and operation explanation

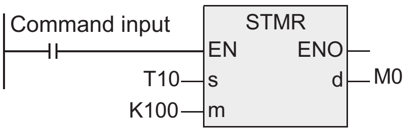
1. 16-bit operation(STMR)

The value designated in m is determined as the set value for the timer of the device designated in (s), and is issued to four points from the device designated in (d).
Create the program depending on the application by referring to the following example.

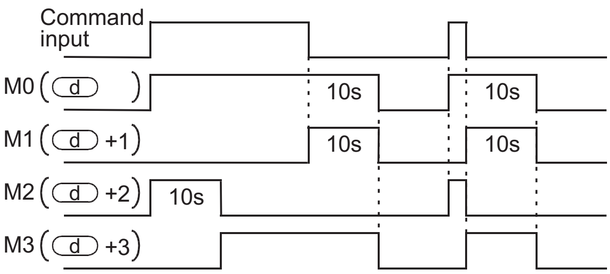


Off-delay timer, one-shot timer

When T10 is assigned in (s), K100 in m, and M0 in (d)

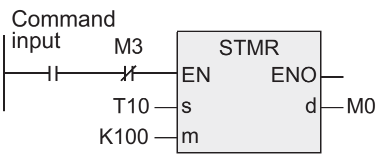


- M0[(d)] : This is an off-delay timer for turning OFF with set time delay of the timer after the command contact is turned OFF.
- M1[(d)+1] : This is a one-shot timer for turning OFF after the timer set time by turning ON after the command contact is changed from ON to OFF.
- M2[(d)+2] : Occupied. To be used for flicker application.
- M3[(d)+3] : Occupied.

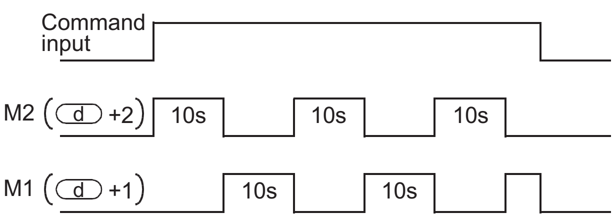


Flicker

The flicker is issued to (d)+1, (d)+2 by preparing the program for turning OFF this command by the b-contact of (d)+3 as shown below.
(d) and (d)+3 are occupied.



- M0[(d)] : Occupied. (To be used in off-delay timer application. See previous page.)
- M1[(d)+1] : This is the flicker (a-contact) for repeating ON/OFF at timer time intervals.
- M2[(d)+2] : This is the flicker (b-contact) for repeating ON/OFF at timer time intervals.
- M3[(d)+3] : Occupied.



Cautions

1. Handling of designated timer

The timer number designated by this command cannot be used in overlap with other general circuit (OUT command, etc.).
If overlapped, the timer does not operate correctly.

2. Number of bits occupied

Four devices are occupied from the one designated in (d).
Be careful not to overlap with the device used in control of the machine.

Device	Function	
	Off-delay timer One-shot timer	Flicker
(d)	Off-delay timer	Occupied
(d)+1	One-shot timer	Flicker(a-contact)
(d)+2	Occupied	Flicker(b-contact)
(d)+3	Occupied	Flicker(b-contact)

3. When command contact is turned OFF

(d), (d)+1, and (d)+3 are turned OFF after the set time. The timers designated in (d)+2 and (s) are reset immediately.

4. Object devices are limited.

▲1: FX3U, FX3UC PLCs only are applicable.
However, index modifier (V, Z) is not applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

7.7.7 ALT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This is the command for inverting the bit device (ON to OFF, OFF to ON) when the input is turned ON.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ALT	16 bits	Continuous	ALT (EN → ENO, d)	ALT(EN, d);
ALTP	16 bits	Pulse	ALTP (EN → ENO, d)	ALTP(EN, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	(d)	Bit device to be output alternately	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)		●	●			●	▲1												●					

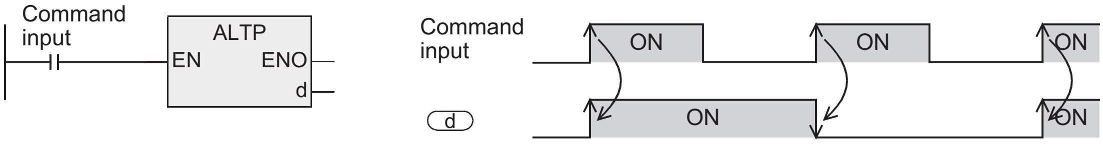
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(ALT, ALTP)

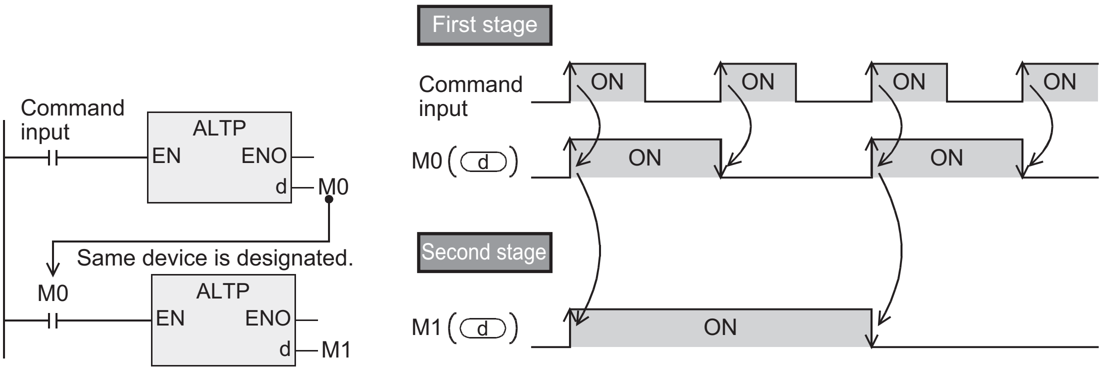
Alternate output (one stage)

Every time the command input is changed from OFF to ON, the bit device designated in (d) is inverted from ON to OFF, from OFF to ON.



Frequency dividing output (alternate output (two stages))

By combining and using a plurality of ALTP commands, the frequency can be divided and issued in multiple stages.



Cautions

1. When using ALT command (continuous execution form)

When programmed by ATL command, the operation is inverted in every operation period.
When inverting by ON/OFF switching of command, use the ALTP command (pulse execution form) or execute the command contact by LDP (pulse execution form).

2. FX0, FX0S, FX0N PLCs do not support the command of pulse execution form.

In the case of pulse execution form, convert the command execution form into pulses.

3. Object devices are limited.

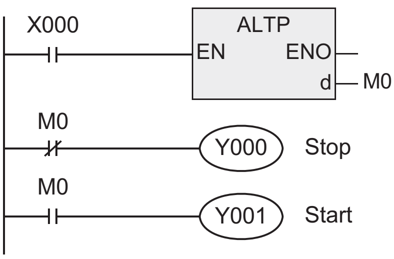
▲1: FX3U, FX3UC PLCs only are applicable.
However, index modifier (V, Z) is not applicable.

Program examples

1. Start/stop by one input

1) By pressing pushbutton X000, the start output Y001 is put in action.
2) By pressing pushbutton X000 again, the stop output Y000 is put in action.

[Structured ladder]



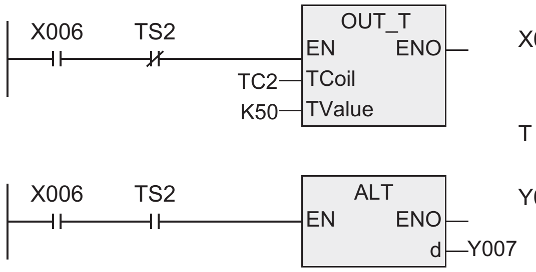
[ST]

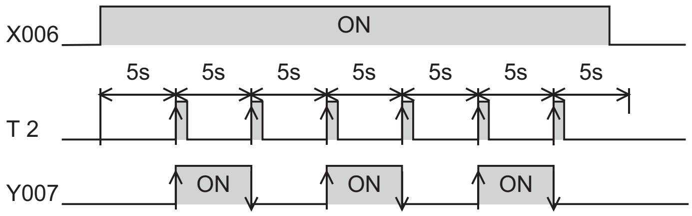
```
ALTP(X000, M0);
OUT(NOT M0, Y000);
OUT(M0, Y001);
```

2. Flicker operation

1) When the input X006 is turned ON, the contact of timer T2 operates momentarily in every 5 seconds.
2) Every time the contact of T2 is turned ON, the output Y007 is alternately turned ON/OFF.

[Structured ladder]





[ST]

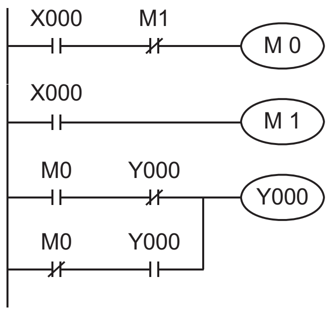
```
OUT_T(X006 AND NOT TS2, TC2, K50);
Y007:= ALT(X006 AND TS2);
```

3. Alternate output operation using auxiliary relay (M) (same operation as ALT command)

The following circuit shows an example of alternate operation by using the basic command and auxiliary relay (M) in the same operation as ALT command.

1) When the X000 is turned on, M0 is turned ON for one operation period only.
2) When M0 is turned ON for the first time, the Y000 operates by self-holding, and when turned ON second time, the self-holding function is canceled.

[Structured ladder]



[ST]

```
OUT(X000 AND NOT M1, M0);
OUT(X000);
OUT((M0 OR NOT Y000) OR (NOT M0 AND Y000), Y000));
```

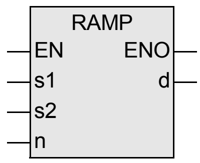
7.7.8 RAMP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	○	○

Outline

This is a command for obtaining data changing n times when designated between the beginning (initial value) and the end (target value).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RAMP	16 bits	Continuous		RAMP(EN, s1, s2, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Device storing initial value of ramp, setting of target value	ANY16
	s2	Device storing initial value of ramp, setting of target value	ANY16
	n	Number of transfer scan times of ramp	ANY16
Output variable	ENO	Execution state	Bit
	d	Device storing data of present value of ramp (2 points occupied)	ANY16(0..1)

3. Applicable devices

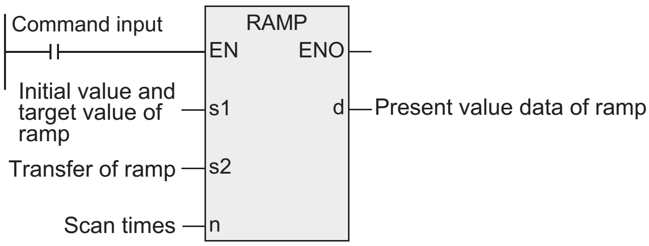
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1				●					
s2														●	▲1				●					
d														●	▲1				●					
n														●	▲1					●	●			

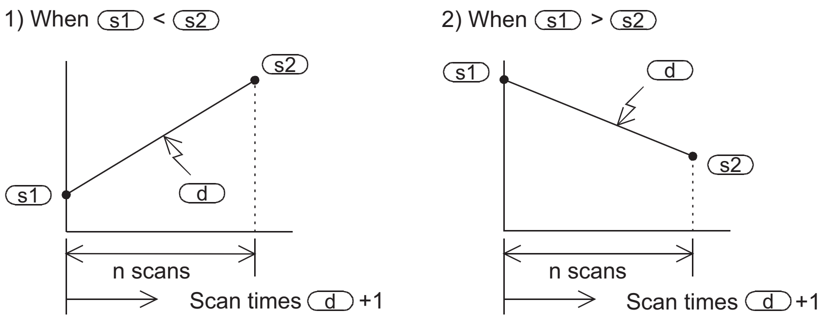
▲: Refer to "Cautions".

Function and operation explanation

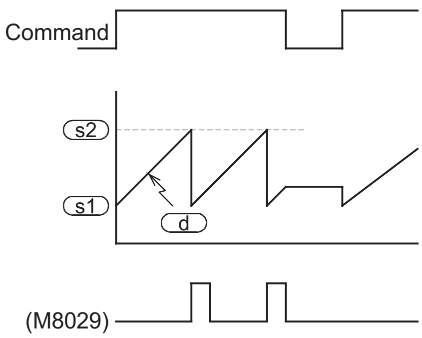
1. 16-bit operation(RAMP)

The starting value and the desired end value are designated in s1 and s2, and when the command input is turned ON, the value equally divided by the number of times designated in n is added to s1 sequentially in every operation period, and the sum is stored in the device designated in d.
This command and the analog output are combined, and the cushion start/stop command can be issued.





- In d+1, scan times (0 to n times) is stored.
- The time from start till end is scanned by operation period × n times.
- When the command input is turned OFF in the midst of operation, the execution is interrupted (present value data of d is held, and d+1 scan times is cleared), and when turned ON again, d is cleared, and the operation is resumed from s1.
- After completion of transfer, the command execution complete flag M8029 is active, and the value of d returns to the value of s1.

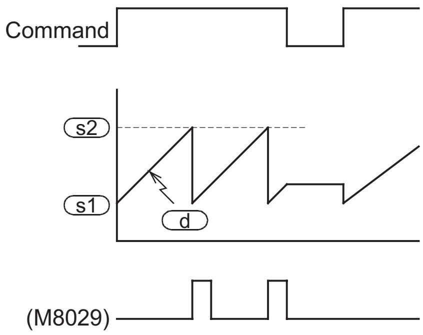


- When obtaining operation results at specific time intervals (constant scan mode)
  Write specified scan time in D8039 (a slightly longer value than actual scan time), and when the M8039 is turned ON, the PLC is set in constant scan mode.
  When this value is, for example, 20 ms and designated by n=100 times, the value of d changes from s1 to s2 in 20 seconds.

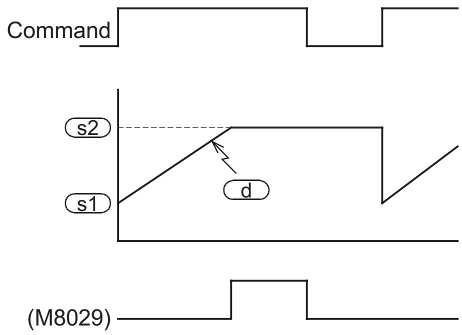
2. Operation of mode flag (M8026)

In FX3U, FX3UC, FX2N, FX2NC, FX2C, FXU (V1.20 or higher) PLCs, by ON/OFF switching of mode flag M8026, the content of (d)+1 is changed as follows.
FX3G, FX1S, FX1N, FX0, FX0S, FX0N PLCs operate same as when the M8026 is turned ON.
FXU (V1.1 or lower) PLC operates same as when the M8026 is turned OFF.

1) In the case of M8026=OFF



2) In the case of M8026=ON



Related devices

→ **As for the method of using the command execution complete flag, see paragraph 1.3.4.**

Device	Name	Content
M8029	Command execution complete	After n operation periods, when becoming (d) = (s2), the device is turned ON.
M8026*1	RAMP mode	See the Operation of mode flag (M8026) explained above.

*1. Cleared when changed from RUN to STOP (applicable only in FX3U, FX3UC, FX2N, FX2NC, FX2C, FXU PLCs).

Cautions

1. When designating power failure hold device (keep region) in the device designated in (d)

When desired to keep the PLC in RUN (start) state while the command input is ON, the device designated in (d) should be cleared beforehand.

2. Object devices are limited.

▲1: FX3U, FX3UC, FX3G PLCs only are applicable.

7.7.9 ROTC

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This is a command suited for turning the table by a shortcut route depending on the demanding window when putting on or taking out articles on the rotary table.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ROTC	16 bits	Continuous	ROTC EN ENO s d m1 m2	ROTC(EN, s, m1, m2, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Data register for counting (3 points occupied)	ANY16
	m1	Number of divisions	ANY16
	m2	Number of low speed sections	ANY16
Output variable	ENO	Execution state	Bit
	d	Beginning bit device to be driven (8 points occupied)	Bit

3. Applicable devices

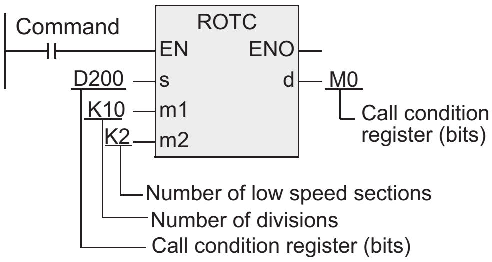
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	▲2				●					
m1																				●	●			
m2																				●	●			
d		●	●			●	▲1												●					

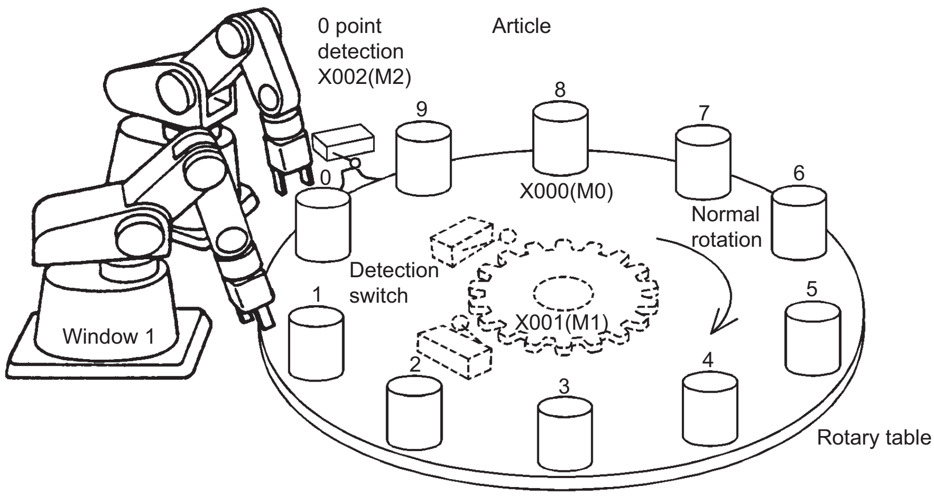
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(ROTC)

As shown below, in order to put on or take out articles on the rotary table divided into m1 sections (=10), depending on the demanding window, the table is controlled and moved by a shortcut route in the condition designated in m2 or (s) and (d).





1) Designation register of call condition (s)

(s)	Register for counting	To be set preliminarily by transfer command
(s) +1	Setting of calling window number	
(s) +2	Setting of calling article number	

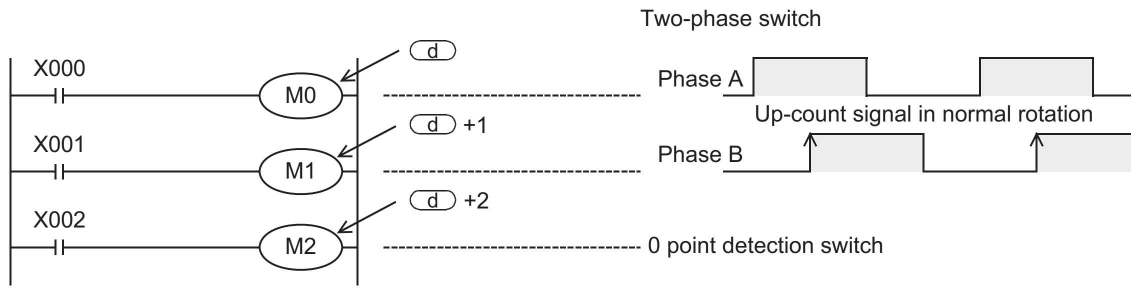
2) Designation bit of calling condition (d)

(d) :	Phase A signal	Internal contact circuit to be driven is preliminarily composed of input signal (X).
(d) +1	Phase B signal	
(d) +2	0 point detection signal	
(d) +3	High speed normal rotation	
(d) +4	Low speed normal rotation	
(d) +5	Stop	
(d) +6	Low speed reverse rotation	
(d) +7	High speed reverse rotation	

Operation condition

The condition necessary for using this command is as shown in the example below.

1) Rotation detection signal: X→(d)
   a) Please install two-phase switch (X000, X001) for detecting normal rotation/reverse rotation of table, and switch X002 for operating when article number 0 comes to window number 0.
   b) Create sequence program as shown in the example below.



By X000 to X002, you can exchange with internal contacts of (d) to (d)+2.
Beginning device number for designating × or (d) is arbitrary.

2) Designation of register for counting: (s)
   (s) is a counter for counting article of which number is coming to window number 0.
3) Designation register of calling condition: (s)+1, (s)+2
   a) In (s)+1, you can set the window number desired to be called.
   b) In (s)+2, you can set the article number desired to be called.
4) Division number m1 and low speed period m2
   To designate division number m1 of table and low speed operation section m2.

When the above condition is designated, the output of normal/reverse rotation, high speed/low speed/stop will be obtained in the output of (d)+3 to (d)+7 designated in the beginning device (d) of the command.

Cautions

1. Operation by ON/OFF of command input

- When this command is driven by turning ON the command input, results of (d)+3 to (d)+7 will be obtained automatically.
- By turning OFF the command input, (d)+3 to (d)+7 are turned OFF.

2. Operation of plural times in one division section of article of rotation detection signal ((d) to (d)+2)

For example, rotation detection signal ((d) to (d)+2) operates 10 times in one division section of article, setting of division number, setting of calling window number, and setting of article number should be all multiplied by 10.
As a result, the set value of low speed section can be set in an intermediate value of division number.

3. 0 point detection signal (d)

When the command input is ON and 0 point detection signal (M2) is turned ON, the content of register for counting (s) is cleared to 0. You must start operation by executing this clearing operation beforehand.

4. Object devices are limited.

▲1: FX3U, FX3UC PLCs only are applicable.
However, index modifier (V, Z) is not applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

7.7.10 SORT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	△	×	×

Outline

This command reshuffles the data table composed of data (columns) and group data (rows) in the ascending order in column unit on the basis of designated group data (rows). In this command, the group data (rows) is stored in continuous devices. Similarly, in SORT2 command, data (columns) is stored in continuous devices, so that the data (columns) may be added easily, and reshuffling is applicable in both ascending order and descending order.

→ **As for SORT2 command, see paragraph 7.13.7.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SORT	16 bits	Continuous	SORT block: inputs EN, s, m1, m2, n; outputs ENO, d	SORT(EN, s, m1, m2, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Beginning device storing data table (m1xm2 points occupied)	ANY16
	m1	Number of data (columns)	ANY16
	m2	Number of group data (rows)	ANY16
	n	Row of group data (rows) as basis of reshuffling	ANY16
Output variable	ENO	Execution state	Bit
	d	Beginning device storing arithmetic results (m1 × m2 points occupied)	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	▲1									
m1																				●	●			
m2																				●	●			
s3														●	▲1									
n														●	▲1					●	●			

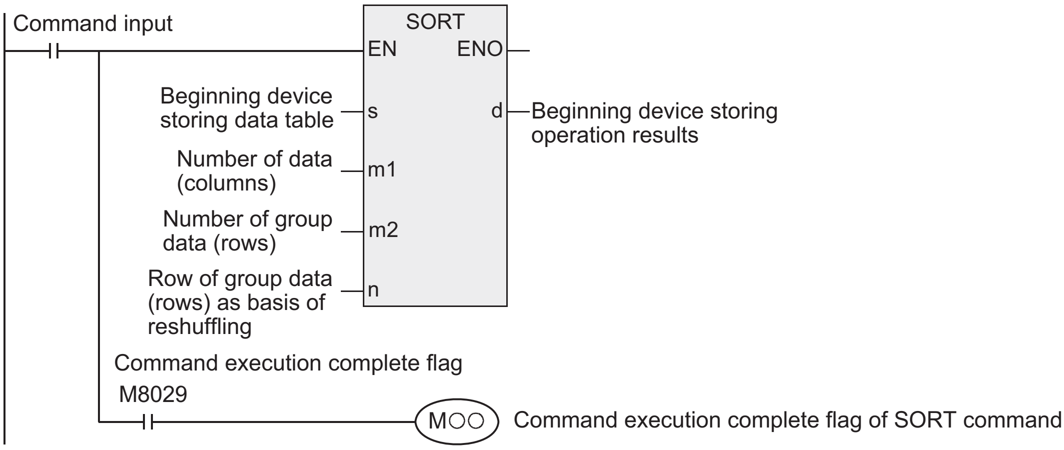
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(SORT)

This command reshuffles the data table (reshuffling origin) composed of (m1×m2) points from the device designated in (s) in the ascending order in data columns on the basis of n rows of group data, and stores in the data table (after reshuffling) of (m1×m2) points from the device designated in (d).

→ **As for operation examples, see next page.**



- The data table composition is explained in an example of m1=K3, m2=K4 at the reshuffling origin. After reshuffling, replace (s) with (d) in data table composition.

Column number \ Row number		Group number m2 (in the case of m2=K4)			
		1	2	3	4
		Management number	**Height**	**Weight**	**Age**
In the case of data number m1=3	**1**	(s)	(s)+3	(s)+6	(s)+9
	2	(s)+1	(s)+4	(s)+7	(s)+10
	3	(s)+2	(s)+5	(s)+8	(s)+11

- When the command input is turned ON, the data arraying is started, and after m1 scan, the data arraying is completed, and the command execution complete flag M8029 is turned ON.

→ **As for the method of using the command execution complete flag, see paragraph 1.3.4.**

2. Operation examples

When the following reshuffling origin data is executed in "n=K2 (row number 2)" and "n=K3 (row number 3)", the operation is as follows.

When a serial number such as management number is entered in the first row, it is convenient because you can judge the original column number by its content.

Reshuffling origin data

Column number \ Row number		Group number m2 (in the case of m2=K4)			
		1	2	3	4
		Management number	**Height**	**Weight**	**Age**
In the case of data number m1=5	**1**	(s)	(s)+5	(s)+10	(s)+15
		1	150	45	20
	2	(s)+1	(s)+6	(s)+11	(s)+16
		2	180	50	40
	3	(s)+2	(s)+7	(s)+12	(s)+17
		3	160	70	30
	4	(s)+3	(s)+8	(s)+13	(s)+18
		4	100	20	8
	5	(s)+4	(s)+9	(s)+14	(s)+19
		5	150	50	45

1) Reshuffling results when command is executed in n=K2 (row number 2)

Column number \ Row number	1 Management number	2 Height	3 Weight	4 Age
1	(s3)	(s3)+5	(s3)+10	(s3)+15
	4	100	20	8
2	(s3)+1	(s3)+6	(s3)+11	(s3)+16
	1	150	45	20
3	(s3)+2	(s3)+7	(s3)+12	(s3)+17
	5	150	50	45
4	(s3)+3	(s3)+8	(s3)+13	(s3)+18
	3	160	70	30
5	(s3)+4	(s3)+9	(s3)+14	(s3)+19
	2	180	50	40

2) Reshuffling results when command is executed in n=K3 (row number 3)

Column number \ Row number	1 Management number	2 Height	3 Weight	4 Age
1	(s3)	(s3)+5	(s3)+10	(s3)+15
	4	100	20	8
2	(s3)+1	(s3)+6	(s3)+11	(s3)+16
	1	150	45	20
3	(s3)+2	(s3)+7	(s3)+12	(s3)+17
	2	180	50	40
4	(s3)+3	(s3)+8	(s3)+13	(s3)+18
	5	150	50	45
5	(s3)+4	(s3)+9	(s3)+14	(s3)+19
	3	160	70	30

Related devices

→ **As for the method of using the command execution complete flag, see paragraph 1.3.4.**

Device	Name	Content
M8029	Command execution complete	To be ON when data arraying is complete.

Cautions

1) Do not change the content of operand or data during operation.
2) When resuming the execution, once turn OFF the command input.
3) Limit of number of times of use of command
   Usable only once in the program.
4) When designating same device in (s) and (d)
   The original data is rewritten in the data sequence after reshuffling.
   In particular, never change the content of the device designated in (s) until the completion of execution.
5) FXU PLC supports the command by V3.07 or higher.
6) Object devices are limited.
   ▲1: FX3U, FX3UC PLCs only are applicable.

7.8 External FX I/O Device

7.8.1 TKY

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This is a command for setting data in the timer or counter by the input of numeric keys 0 to 9.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
TKY	16 bits	Continuous	TKY (EN, s → ENO, d1, d2)	TKY(EN, s, d1, d2);
DTKY	32 bits	Continuous	DTKY (EN, s → ENO, d1, d2)	DTKY(EN, s, d1, d2);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Beginning bit device for entering numeric keys (10 points occupied)	Bit	
Output variable	ENO	Execution state	Bit	
	d1	Word device for storing data	ANY16	ANY32
	d2	Beginning bit device for turning ON key pushing information (11 points occupied)	Bit	

3. Applicable devices

Operand type	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
	Bit Devices: System user							Word Devices: Digit designation				System user				Special unit	Index			Others: Constant		Real number	Character string	Pointer
s	●	●	●			●	▲1												●					
d1									●	●	●	●	●	●	▲2	▲2	●	●	●					
d2		●	●			●	▲1												●					

▲: Refer to "Cautions".

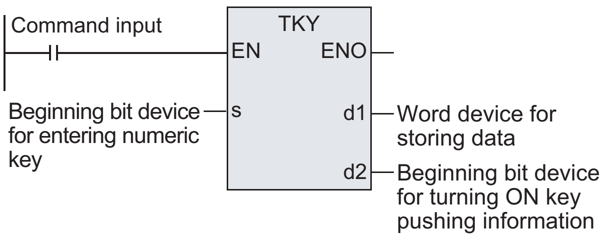
Function and operation explanation

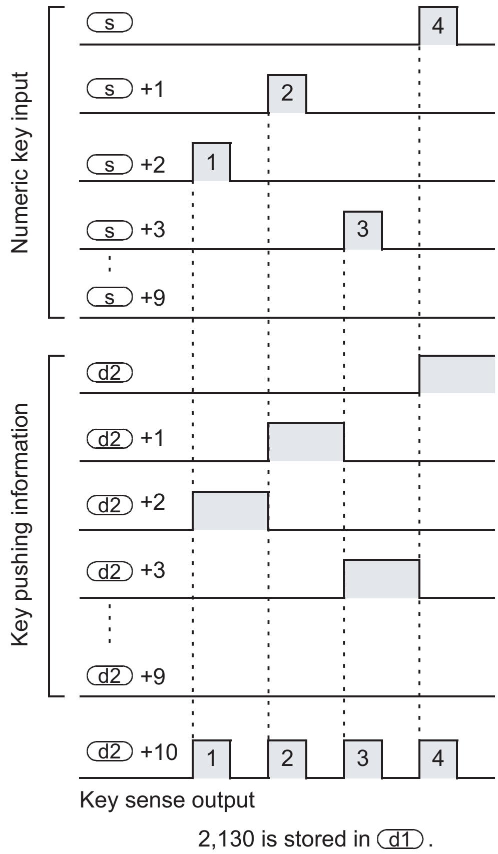
1. 16-bit operation(TKY)

By pressing key from the input for connecting the numeric keys (device designated in s), the entered numerical value is stored in the device designated in d1, and the key pushing information and key sense output are issued to the device designated in d2.

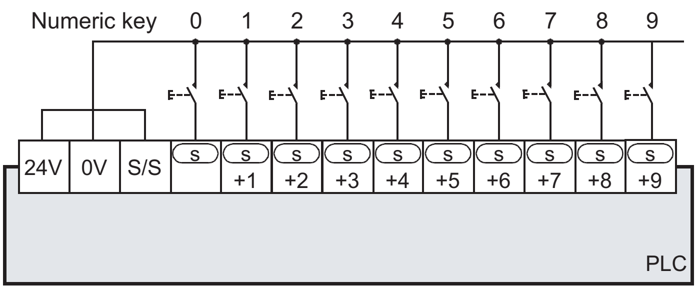
1) Entered numerical value
   - If more than 9,999, the value overflows from the higher digits.
   - The entered numerical value is stored in the BIN (binary) value.
   - In the diagram of next page, when numeric keys are pressed in the sequence of 1, 2, 3, 4, then "2,130" is stored in the device designated in d1.

2) Key pushing information
   - Key pushing information of d2 to d2+9 is turned ON/OFF depending on the pushed key.
   - Key sense output of d2+10 is ON when either key is being pushed.





The diagram below shows an example of FX3U PLC (sync input). As for the wiring, see the manual of each PLC.

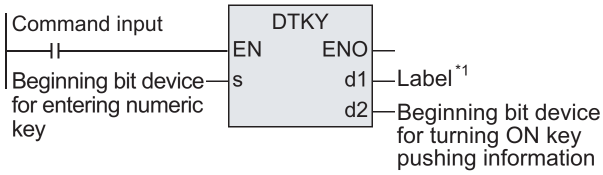


2. 32-bit operation(DTKY)

By pressing key from the input for connecting the numeric keys (device designated in (s)), the entered numerical value is stored in the device designated in (d1), and the key pushing information and key sense output are issued to the device designated in (d2).

1) Entered numerical value
- If more than 99,999,999, the value overflows from the higher digits.
- The entered numerical value is stored in the BIN (binary) value.

2) Key pushing information
- Key pushing information of (d2) to (d2)+9 is turned ON/OFF depending on the pushed key.
- Key sense output of (d2)+10 is ON when either key is being pushed.



*1.To define the word device for storing data.

As for numeric key connection example and key pushing information, see the above explanation of 16-bit operation (TKY).

Cautions

1. When keys are pressed simultaneously

When plural keys are pressed, only the first pressed key is valid.

2. When command contact is turned OFF

If turned OFF, the content of (d1) is not changed, but all of (d2) to (d2)+10 are turned OFF.

3. Number of bits occupied

1) When a numeric key is connected, 10 points are occupied from (s).
   When numeric key is not connected (not used), the devices are occupied and cannot be used in other applications.

2) A total of 11 points are occupied from the beginning device (d2) for output of key pushing information.
   Be careful not to overlap with the device used in control of machine.
   - (d2) to (d2)+9: Turned ON corresponding to the input of numeric keys 0 to 9.
   - (d2)+10: Kept ON while any one of 0 to 9 is being pressed (key sense output).

4. Limit of number of times of use of command

Only one of TKY command and DTKY command can be used in the program.
If desired to use plural times, you can program by the index modifier (V, Z) function.

5. Designation of input and output variables

When handling 32-bit data in a structured program, you cannot designate 16-bit device directly unlike in simple project. When handling 32-bit data, please use the label. However, the 32-bit counter is a device of 32 bits long, and the device can be designated directly. When designating the device, please use the global label.

6. Object devices are limited.

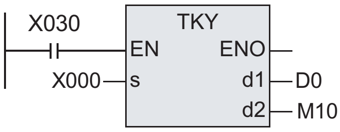
▲1: FX3U, FX3UC PLCs only are applicable.
However, index modifier (V, Z) is not applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

Program examples

This is an explanation of an example in which the input X000 is the beginning, and numeric keys 0 to 9 are connected.

1. Program

[Structured ladder]

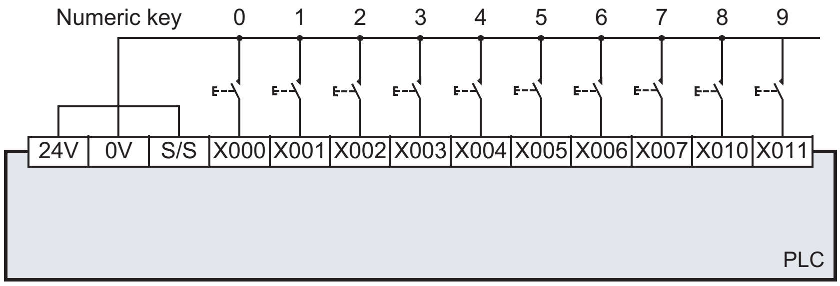


[ST]

```
TKY(X030, X000, D0, M10);
```

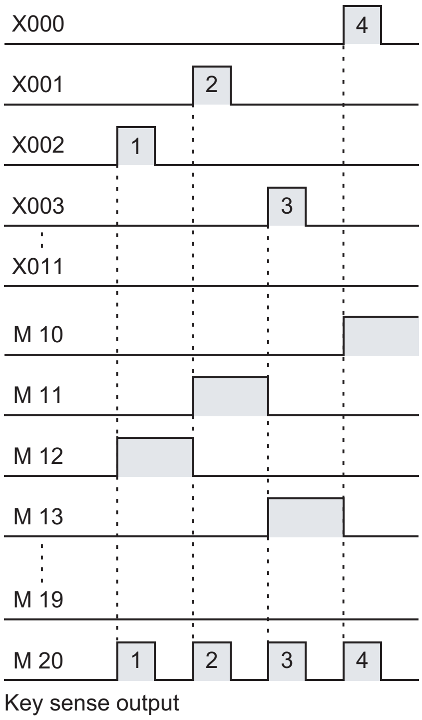
2. Wiring diagram

This wiring diagram is an example of FX3U PLC (sync input). As for the actual wiring connection, see the manual of the PLC.



3. Timing chart

1) When numeric keys are pressed in the sequence of 1, 2, 3, 4, the content of D0 is 2,130.
   Numerical value more than 9,999 overflows sequentially from the upper digits. (The actual content of D0 is BIN data.)
2) When X002 is pressed, M12 is set (ON) until other key is pressed. It is the same when other key is pressed.
   Thus, depending on the operation of input X000 to X011, operation of M10 to M19 is carried out.
3) When any key is pressed, the key sense output M20 is kept ON while being pressed.



7.8.2 HKY

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This is a command for setting the input data of numerical value (0 to 9) or operation condition (function keys A to F), by the input of keys from 0 to F (16 keys).
When the extension function is turned ON, the key input is entered in hexadecimal notation by keys 0 to F.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
HKY	16 bits	Continuous	HKY EN ENO s d1 d2 d3	HKY(EN, s, d1, d2, d3);
DHKY	32 bits	Continuous	DHKY EN ENO s d1 d2 d3	DHKY(EN, s, d1, d2, d3);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Beginning device for entering 16 keys (X) [4 points occupied].	Bit	
Output variable	ENO	Execution state	Bit	
	d1	Beginning device for sending output (Y) [4 points occupied]	Bit	
	d2	Device for storing numerical value entered from 16 keys	ANY16	ANY32
	d3	Beginning bit device for turning ON key pushing information [8 points occupied]	Bit	

3. Applicable devices

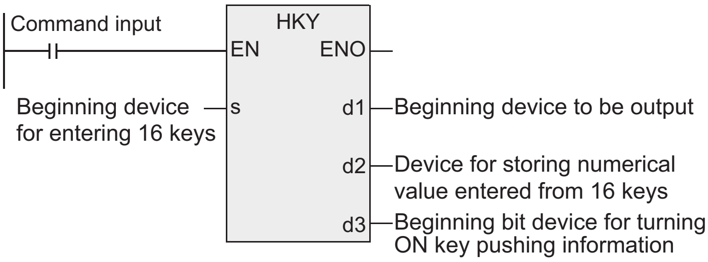
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●																		●					
d1		●																	●					
d2												●	●	●	▲2	▲2	●	●	●					
d3		●	●			●	▲1												●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(HKY)

The numerical value scanned by the signals of the input for connecting 16 keys (0 to F) (device designated in s) and the row output (device designated in d1) and by pressing keys 0 to 9 is stored in the device designated in d2, and the key sense output is issued to the device designated in d3.
When any one of keys A to F is pressed, the key pushing information corresponding to the key (device designated in d3) is turned ON, and the key sense output (device designated in d3) is issued.



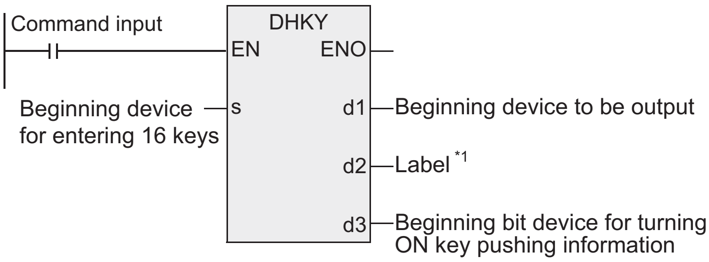
1) Numerical value input by keys 0 to 9
   If more than 9,999, the value overflows from the higher digits.
   - The entered numerical value is stored in d2 as BIN (binary) value.
   - The key sense output d3 +7 is turned ON by pressing any one of keys 0 to 9.
2) Key pushing information of keys A to F
   - To turn ON 6 points from d3 corresponding to any one of keys A to F.
   - The key sense output d3 +6 is turned ON by pressing any one of keys A to F.

Key	Key pushing information
A	d3
B	d3 +1
C	d3 +2

Key	Key pushing information
D	d3 +3
E	d3 +4
F	d3 +5

2. 32-bit operation(DHKY)

The numerical value scanned by the signals of the input for connecting 16 keys (0 to F) (device designated in s) and the row output (device designated in d1) and by pressing keys 0 to 9 is stored in the device designated in d2, and the key sense output is issued to the device designated in d3.
When any one of keys A to F is pressed, the key pushing information corresponding to the key (device designated in d3) is turned ON, and the key sense output (device designated in d3) is issued.



*1.To define the device for storing the numerical value entered from 16 keys.

1) Numerical value input by keys 0 to 9
   - If more than 99,999,999, the value overflows from the higher digits.
   - The entered numerical value is stored in (d2 +1, d2) as BIN (binary) value.
   - The key sense output d3 +7 is turned ON by pressing any one of keys 0 to 9.
2) Key pushing information of keys A to F
   As for the key pushing information, refer to the previous page on 16-bit operation (HKY).

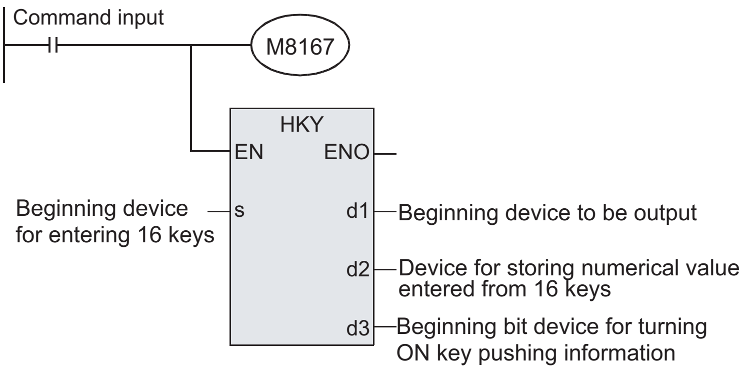
Extension function

When the extension function is validated by turning ON the M8167, the hexadecimal key pushing data of 0 to F is stored in BIN.
Except for the following, this is same as the "function and operation explanation" given above.
FXU PLC: V2.30 or lower has no extension function.

1. 16-bit operation(HKY)

Hexadecimal data entered by keys 0 to F is directly written into the device designated in (d2).

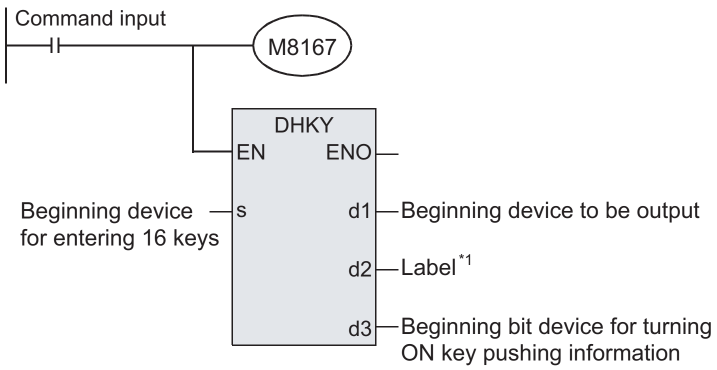
1) Numerical value input by keys 0 to F
   - If more than FFFF, the value overflows sequentially from the upper digits.
   - Example:
     In the case of input of 1 → 2 → 3 → B → F, "23BF" is stored in BIN in the device designated in (d2).
     That is, 1 overflows at the time of input of F.



2. 32-bit operation(DHKY)

Hexadecimal data entered by keys 0 to F is directly written into the device designated in (d2).

1) Numerical value input by keys 0 to F
   - If more than FFFFFFFF, the value overflows sequentially from the upper digits.
   - Example:
     In the case of input of 9 → 2 → 3 → B → F → A → F, "923BFAF" is stored in BIN.



*1.To define the device for storing the numerical value entered from 16 keys.

Related devices

→ **As for the method of using the command execution complete flag, see paragraph 1.3.4.**

Device	Name	Content
M8167	Function extension flag	HEX data handling function of HKY command OFF : numeric key + function key ON : hexadecimal key
M8029	Command execution complete	OFF : During scanning from (d1) to (d1)+3, or command not executed ON : To be turned ON after one-cycle operation of (d1) to (d1)+3 output (key scan of 0 to F).

Cautions

1. Limit of number of times of use of command

Only one of HKY command and DHKY command can be used in the program.
If desired to use plural times, you can program by the index modifier (V, Z) function.

2. When keys are pressed simultaneously

When plural keys are pressed, only the first pressed key is valid

3. When command contact is turned OFF

If turned OFF, the content of (d2) is not changed, but all of (d3) to (d3)+7 are turned OFF.

4. Number of bits occupied

1) To occupy 4 points for connecting 16 keys from the beginning device (s) of input (X).
2) To occupy 4 points for connecting 16 keys from the beginning device (d1) of output (Y).
3) To occupy 8 points from beginning device (d3) for output of key pushing information.
   Be careful not to overlap with the device used in control of the machine.
   - (d3) to (d3)+5 : Key pushing information of keys A to F
   - (d3)+6 : Key sense output of keys A to F.
   - (d3)+7 : Key sense output of keys 0 to 9

5. Intake timing of key input

HKY and DHKY commands are executed simultaneously with the operation period of PLC.
At the end of a series of key scan, a time of 8 scans is needed.
To prevent intake error due to filter delay of key input, you can utilize the function of "constant scan mode" or "timer interruption."

6. Output format

Select and use the PLC of transistor output type.

7. Designation of input and output variables

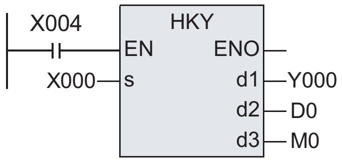
When handling 32-bit data in a structured program, you cannot designate 16-bit device directly unlike in simple project. When handling 32-bit data, please use the label. However, the 32-bit counter is a device of 32 bits long, and the device can be designated directly. When designating the device, please use the global label.

8. Object devices are limited.

▲1: FX3U, FX3UC PLCs only are applicable.
However, index modifier (V, Z) is not applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

Program examples

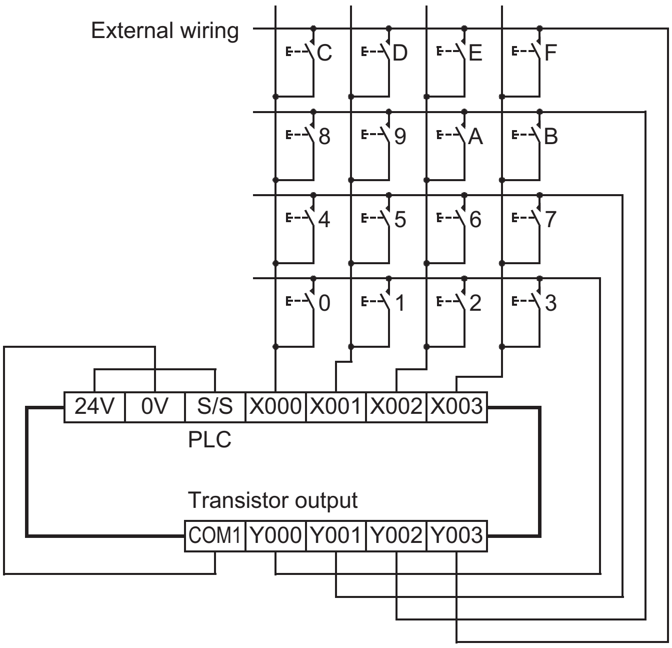
[Structured ladder]



[ST]

```
HKY(X004, X000, Y000, D0, M0);
```

The wiring diagram below is an example of basic unit (sync input/sync output) of FX3U series.
For the actual wiring connection, see the manual of the PLC.



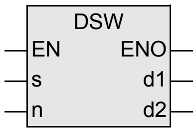
7.8.3 DSW

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This is a command for reading in the set value of digital switch.
You can read in the data of 4 digits and 1 set (n=K1), or 4 digits and 2 sets (n=K2).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DSW	16 bits	Continuous		DSW(EN, s, n, d1, d2);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Beginning device for connecting digital switch (X) (4Xn points occupied)	Bit
	(n)	Number of sets of digital switch (4 digits/1 set) [n=1 or 2]	ANY16
Output variable	ENO	Execution state	Bit
	(d1)	Beginning device of output of strobe signal (Y) (4 points occupied)	Bit
	(d2)	Device for storing numerical value of digital switch (n points occupied)	ANY16

3. Applicable devices

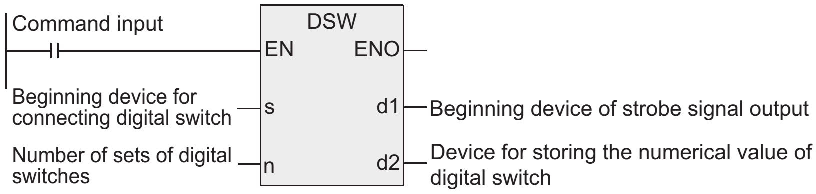
Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real number	Others: Character string	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)	●																		●					
(d1)		●																	●					
(d2)												●	●	●	▲1	▲2	●	●	●					
(n)																				●	●			

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(DSW)

The value of digital switch connected to the device designated in s is processed by time division (entered sequentially from the first digit by output signals at 100 ms intervals), and stored in the device designated in d2.



1) Data d1
   - You can read up to 4 digits from 0 to 9,999.
   - Data is stored in BIN (binary value).
   - First set is stored in d2, and second set in d2+1.
2) Designation of number of sets n
   - When using 4 digits and 1 set × 1 (n=K1)
     Digital switch of 4 digits in BCD connected to s to s+3 is sequentially read in by strobe signals d1 to d1+3, and stored in d2 as BIN value.
   - When using 4 digits and 1 set × 2 (n=K2)
     Digital switch of 4 digits in BCD connected to s to s+3 is sequentially read in by strobe signals d1 to d1+3, and stored in d2 as BIN value.
   - Digital switch of 4 digits in BCD connected to s+4 to s+7 is sequentially read in by strobe signals d1 to d1+3, and stored in d2+1 as BIN value.

Related devices

→ As for the method of using the command execution complete flag, see paragraph 1.3.4.

Device	Name	Content
M8029	Command execution complete	OFF : Scan continuous or command not executed from d1 to d1+3. ON : ON after one-cycle operation of d1 to d1+3 output (scan of 1 to 4 digits)

Cautions

1. When command contact is turned OFF

If turned OFF, the content of d2 is not changed, but all of d1 to d1+3 are turned OFF.

2. Number of bits occupied

1) When 4 digits and 2 sets (n=K2) are used, 2 points are occupied from d2.
2) s occupies 4 points in the case of 4 digits and 1 set, and occupies 8 points in the case of 4 digits and 2 sets.

3. When connecting a digital switch of less than 4 digits

The wiring of strobe signal <output for digit designation> d1 is not needed in the digit not in use, but the output for digit not in use is occupied by this command, and cannot be used in other application. The output not in use must be always kept vacant.

4. We recommend to use the transistor output type.

To take in the values of digital switch continuously, you must use the PLC of transistor output.

→ As for the relay output type, refer to the "Method of using by relay output type" shown later.

5. Digital switch

Use the digital switch of BCD output.

6. Object devices are limited.

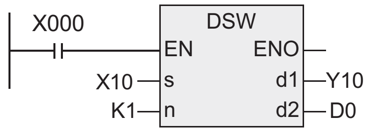
▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

Program examples

This is an example of explaining connection of digital switch starting from the input X010 and starting from the digit designation output Y010.

1. Program

[Structured ladder]

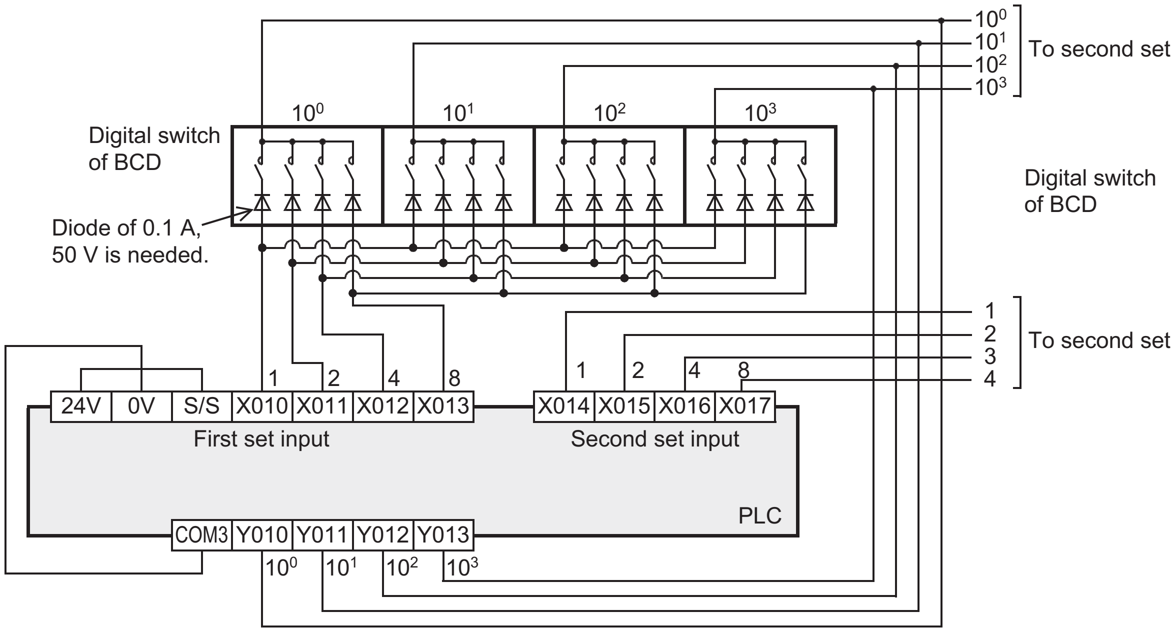


[ST]

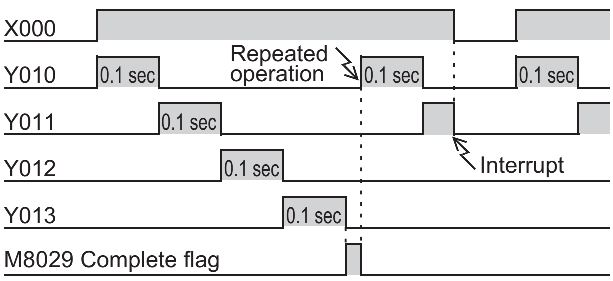
```
DSW(X000, X10, K1, Y10, D0);
```

2. Wiring diagram

This wiring connection diagram is an example of basic unit of FX3U series (sync input/sync output). As for the actual wiring, see the manual of the PLC.



3. Timing chart

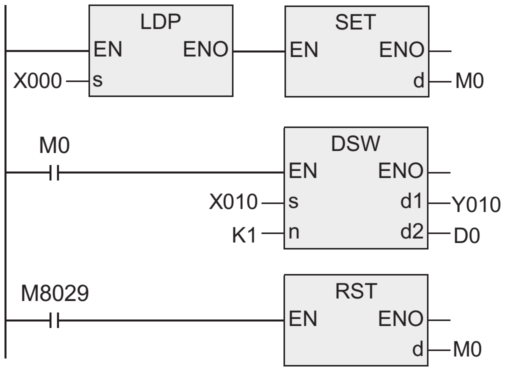


Y010 to Y013 are turned ON sequentially in every 100 ms while X000 is being turned ON, and when operation of one cycle is over, the execution complete flag M8029 is set up.

4. Method of using by relay output type

You can also use the relay output type PLC by providing with "digital switch reading input."
When pushbutton (X000) is pressed, the DSW performs a series of operations. In the case of this program, if Y010 to Y013 are relay outputs, there is no problem in relay contact life.

[Structured ladder]



[ST]

```
IF(LDP(TRUE, X000)) THEN SET(TRUE, M0);
DSW(M0, X010, K1, Y010, D0);
RST(M8029, M0);
```

1) DSW is operating while the M0 (digital switch reading input) is being ON.
2) DSW continues to operate until operation of one cycle is over and execution complete flag (M8029) is set up.

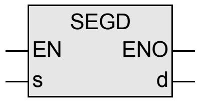
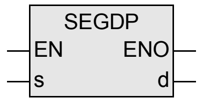
7.8.4 SEGD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This is a command for lighting up the 7-segement display unit (1 digit) by decoding the data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
SEGD	16 bits	Continuous		SEGD(EN, s, d);
SEGDP	16 bits	Pulse		SEGDP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Beginning word device to be decoded	ANY16
Output variable	ENO	Execution state	Bit
	d	Word device for storing 7-segment display data	ANY16

3. Applicable devices

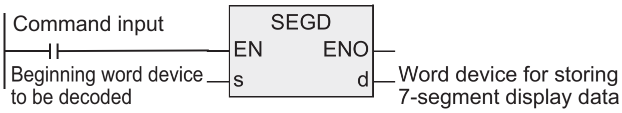
Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real number	Others: Character string	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲1	▲1	●	●	●	●	●			
d									●	●	●	●	●	●	▲1	▲1	●	●	●					

▲: Refer to "Cautions".

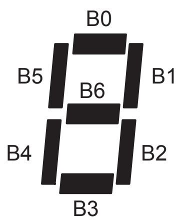
Function and operation explanation

1. 16-bit operation(SEGD, SEGDP)

0 to F (hexadecimal) of lower 4 bits (1 digit) of the device designated in (s) are decoded into 7-segment display data, and stored in lower 8 bits of the device designated in (d).



2. 7-segement decoding table

(s)					Composition of 7 segments	(d)											Display data
Hexadecimal	b3	b2	b1	b0		B15	•••	B8	B7	B6	B5	B4	B3	B2	B1	B0	
0	0	0	0	0		-		-	0	0	1	1	1	1	1	1	0
1	0	0	0	1		-		-	0	0	0	0	0	1	1	0	1
2	0	0	1	0		-		-	0	1	0	1	1	0	1	1	2
3	0	0	1	1		-		-	0	1	0	0	1	1	1	1	3
4	0	1	0	0		-		-	0	1	1	0	0	1	1	0	4
5	0	1	0	1		-		-	0	1	1	0	1	1	0	1	5
6	0	1	1	0		-		-	0	1	1	1	1	1	0	1	6
7	0	1	1	1		-		-	0	0	1	0	0	1	1	1	7
8	1	0	0	0		-		-	0	1	1	1	1	1	1	1	8
9	1	0	0	1		-		-	0	1	1	0	1	1	1	1	9
A	1	0	1	0		-		-	0	1	1	1	0	1	1	1	A
B	1	0	1	1		-		-	0	1	1	1	1	1	0	0	b
C	1	1	0	0		-		-	0	0	1	1	1	0	0	1	C
D	1	1	0	1		-		-	0	1	0	1	1	1	1	0	d
E	1	1	1	0		-		-	0	1	1	1	1	0	0	1	E
F	1	1	1	1		-		-	0	1	1	1	0	0	0	1	F

↑

B0 is the beginning of bit device or the lowest bit of the word device.

Cautions

1) Number of bits occupied
   Lower 8 bits from the output of the device designated in (d) are occupied, and the upper 8 bits are not changed.
2) Object devices are limited.
   ▲1: FX3U, FX3UC PLCs only are applicable.

7.8.5 SEGL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This is a command for controlling the 7-segment display unit with latch of 4 digits and 1 set or 4 digits and 2 sets.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SEGL	16 bits	Continuous	SEGL: EN, s, n → ENO, d	SEGL(EN, s, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Beginning word device for BCD conversion • n=K(H)0 to K(H)3: 1 point occupied • n=K(H)4 to K(H)7: 2 points occupied	ANY16
	(n)	Parameter	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Beginning device to be output (Y) • n=K(H)0 to K(H)3: 8 points occupied • n=K(H)4 to K(H)7: 12 points occupied	Bit

3. Applicable devices

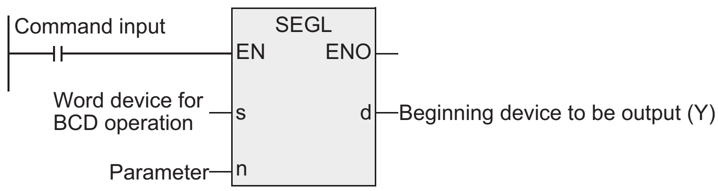
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user			Special unit	Index				Cons tant		Real number	Character string	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)		●																	●					
(n)																				●	●			

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(SEGL)

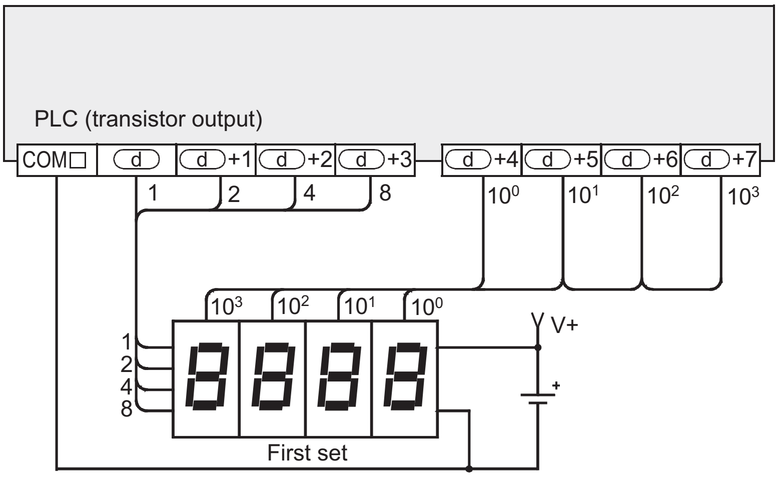
The 4-digit numerical value of the device designated in (s) is converted into BCD data, and is sequentially divided in time digit by digit, and is sent into the 7-segment display unit with BCD decoder.



When using 4 digits and 1 set(n=K0 to K3)

→ **As for selection of "n", see the second item below.**

1) Data and strobe signal
   The 4-digit numerical value of (s) is converted from BIN to BCD, and is issued sequentially by time division digit by digit from (d) to (d)+3.
   The strobe signal output ((d)+4 to (d)+7) is also issued sequentially by time division, and the 7-segment display of 4 digits and 1 set is latched.
2) In (s), BIN data in a range of 0 to 9,999 is valid.
3) Connection example of 7-segment display unit
   The following diagram show an example of FX3U series basic unit (sync output). As for the actual wiring, see the manual of the PLC.

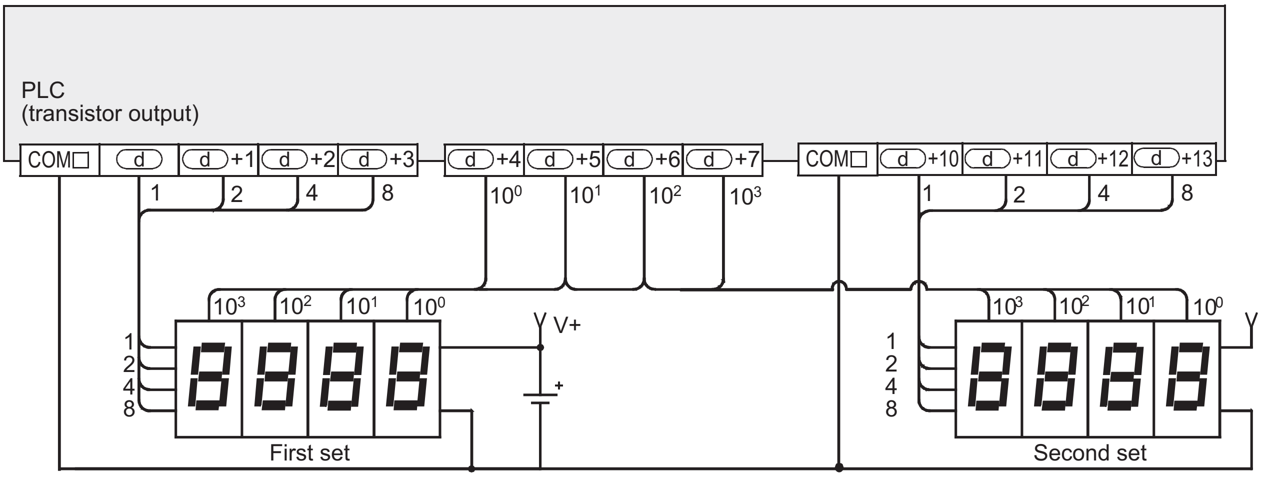


When using 4 digits and 2 sets(n=K4 to K7)

→ **As for selection of "n", see the item below.**

1) Data and strobe signal
   a) 4 digits, first set
      The 4-digit numerical value of (s) is converted from BIN to BCD, and is issued sequentially by time division digit by digit from (d) to (d)+3.
      The strobe signal output ((d)+4 to (d)+7} is also issued sequentially by time division, and the 7-segment display of 4 digits and 1 set is latched.
   b) 4 digits, second set
      The 4-digit numerical value of (s)+1 is converted from BIN to BCD, and is issued sequentially by time division digit by digit from (d)+10 to (d)+13.
      The strobe signal output ((d)+4 to (d)+7) is also issued sequentially by time division, and the 7-segment display of 4 digits and 2 sets is latched. (The strobe signal output ((d)+4 to (d)+7) is common in each set.)
2) In (s) and (s)+1, BIN data in a range of 0 to 9,999 is valid.

3) Connection example of 7-segment display unit
The following diagram show an example of FX3U series basic unit (sync output).
As for the actual wiring, see the manual of the PLC.



Related devices

→ **As for the method of using the command execution complete flag, see paragraph 1.3.4.**

Device	Name	Content
M8029	Command execution complete	To be turned ON when output of 4 digits is over.

Cautions

1. Time for updating 7-segement 4-digit display

The time for updating the display of 4 digits (1 set or 2 sets) is required by 12 times of the scan time (operation time).

2. Operation when command input is turned OFF

While the command input is ON, the operation is repeated, but once the command contact is turned OFF during operation, the operation is interrupted, and when turned ON again, the operation is resumed from the beginning.

3. Number of bits occupied

When using 4 digits and 1 set : 1 point is occupied from the beginning device designated in (s).
8 points are occupied from the beginning device designated in (d). If the number of digits is smaller, the remainder cannot be used in other application.

When using 4 digits and 2 sets: 2 points are occupied from the beginning device designated in (s).
12 points are occupied from the beginning device designated in (d). If the number of digits is smaller, the remainder cannot be used in other application.

4. Scan time (operation period) and display timing

SEGL command is executed in synchronism with the scan time (operation period) of the PLC.
For a series of display, the scan time of the PLC is required by 10 ms or more.
If less than 10 ms, you can operate in the scan time of 10 ms or more by using the constant scan mode.

5. Output format of PLC

Use the PLC of transistor output type.

6. Object devices are limited.

▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

Selection procedure of 7-segment display unit

You can select the 7-segment display unit depending on the electrical content by referring to the example below.

→ **As for the actual wiring, see the hardware manual of the PLC main body.**

1. Check points by 7-segment specification

1) Check if the data input, and input voltage and current characteristics of strobe signal are satisfying the output specification of the PLC or not.
   - Check if the input signal voltage (Lo) is about 1.5 V or less.
   - Check if the input voltage is DC 5 V to DC 30 V.
2) Check if the BCD decoding or latching function is provided or not.

Selection procedure of parameter n by specification of 7-segment display

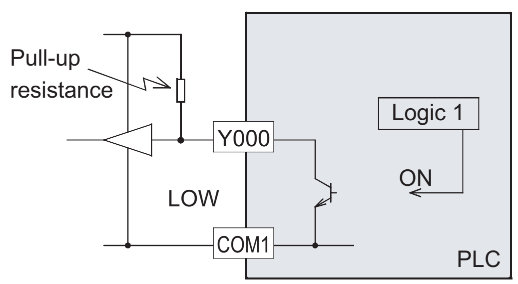
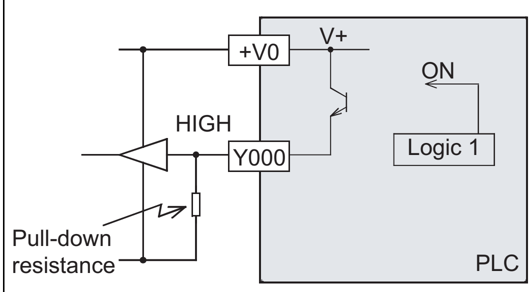
The value to be set in parameter n varies with the signal logic of 7-segment display.
Select in the following procedure.
A check column is provided in the final line of the table. Check the corresponding positive or negative logic, and select the parameter accordingly.

1. Role of parameter n

Parameter n is a number selected depending on the logic (positive or negative) of 7-segment data input, the logic (positive or negative) of strobe signal, or control of 4 digits and 1 set or control of 2 sets.

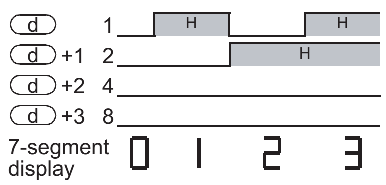
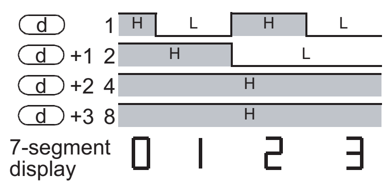
2. Check the output logic of the PLC

The transistor output of the PLC is available in two types, sync output and source output. The specification is different as shown below.

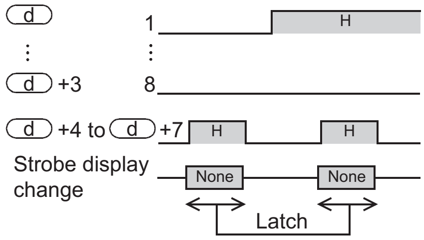
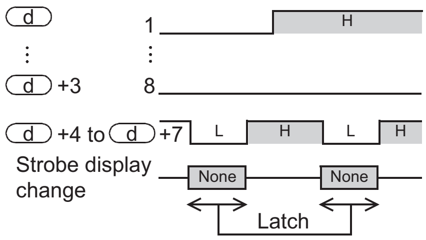
Logic	Negative logic	Positive logic
Output format	Sync output [- common]	Source output [+ common]
Output circuit		
Explanation	Because of transistor output (sync), when the internal logic is 1 (ON output), the output is LOW level (0V). This is called the negative logic.	Because of transistor output (source), when the internal logic is 1 (ON output), the output is HIGH level (V+). This is called the positive logic.
Logic check		

3. Check the logic of the 7-segment display unit.

1) Data input

Logic	Negative logic	Positive logic
Timing chart		
Explanation	Becoming BCD data at LOW level	Becoming BCD data at HIGH level
Logic check		

2) Strobe signal

Logic	Negative logic	Positive logic
Timing chart		
Explanation	Data latched at LOW level is held	Data latched at HIGH level is held
Logic check		

4. Selection of parameter n

Select by referring to the table below depending on the positive or negative logic of the PLC side, and the positive or negative logic of the 7-segment display side.

PLC output logic	Data input	Strobe signal	Parameter n	
			4 digits × 1 set	4 digits × 2 sets
Negative logic	Negative logic (coinciding)	Negative logic (coinciding)	0	4
		Positive logic (not coinciding)	1	5
	Positive logic (not coinciding)	Negative logic (coinciding)	2	6
		Positive logic (not coinciding)	3	7
Positive logic	Positive logic (coinciding)	Positive logic (coinciding)	0	4
		Negative logic (not coinciding)	1	5
	Negative logic (not coinciding)	Positive logic (coinciding)	2	6
		Negative logic (not coinciding)	3	7

5. Explanation of selection method of parameter n by exemplary cases

When the following 7-segment display is connected, n=1 in the case of 4 digits × 1 set, and n=5 in the case of 4 digits × 2 sets.

1) Transistor output of PLC
- Sync output = negative logic
- Source output = positive logic

2) 7-segment display
- Data input = negative logic
- Strobe signal = positive logic

PLC output logic	Data input	Strobe signal	Parameter n	
			4 digits × 1 set	4 digits × 2 sets
Negative logic	Negative logic (coinciding)	Negative logic (coinciding)	0	4
		Positive logic (not coinciding)	1	5
	Positive logic (not coinciding)	Negative logic (coinciding)	2	6
		Positive logic (not coinciding)	3	7

7.8.6 ARWS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This instruction enters data by arrow switches for digit move and increase and decrease of numerical value of each digit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ARWS	16 bits	Continuous	ARWS EN ENO s d1 n d2	ARWS(EN, s, n, d1, d2);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head bit device to be entered [4 points occupied]	Bit
	n	Digit number specification of 7-segment display.	ANY16
Output variable	ENO	Execution state	Bit
	d1	Word device in which BCD converted data is stored	ANY16
	d2	Head bit device (Y) for connecting 7-segment display unit [8 points occupied]	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices													Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●	●	●			●	▲1												●					
d1												●	●	●	▲2	▲2	●	●	●					
d2		●																	●					
n																				●	●			

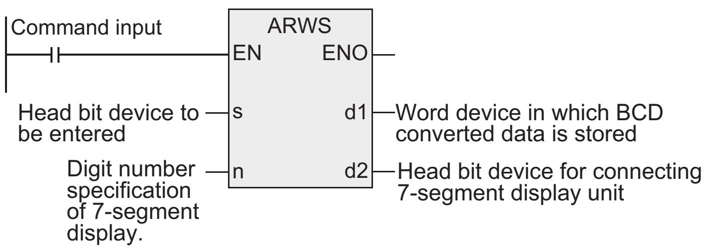
▲: Refer to "Cautions".

Function and operation explanation

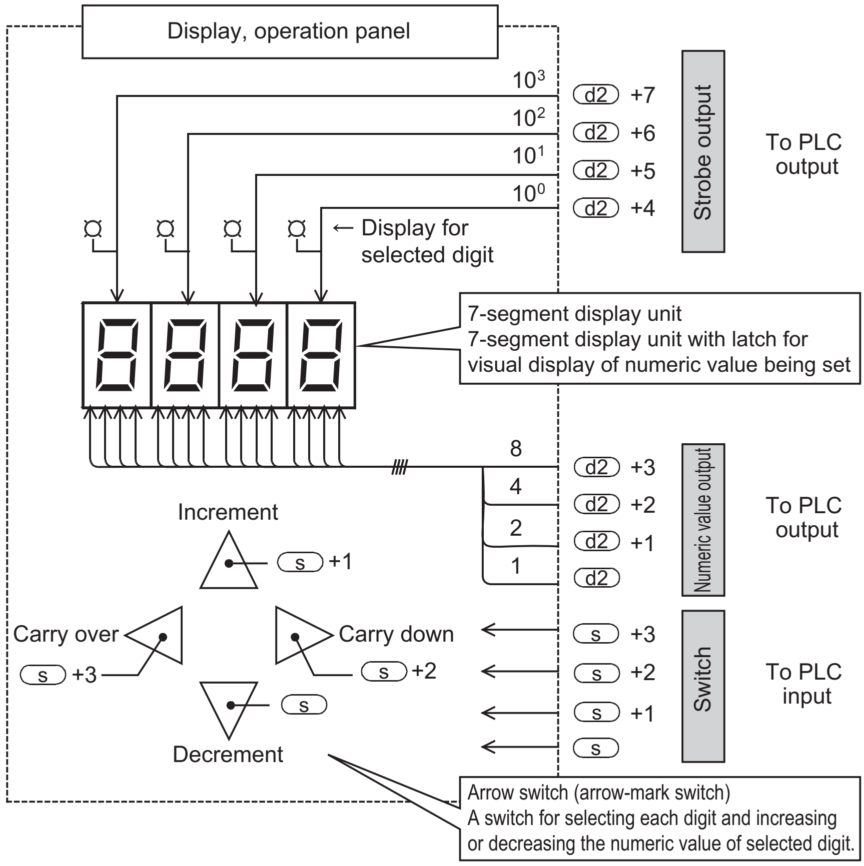
Four arrow switches are connected to the input of the device specified by s, the 7-segment display unit with BCD decoder is connected to the output of the device specified by d2, and the numeric value is entered in the device specified by d1.

1. 16-bit operation (ARWS)

In the device specified by d1, 16-bit BIN value of 0 to 9,999 is stored, but for the convenience of explanation, BCD converted value is expressed.
When the command input is ON, then ARWS instruction operates as follows.



Content of display and operation unit



1) Digit specification n of 7-segment display unit with BCD decoder
   In the following explanation of operation, four digits (10^3 digits) are supposed.
2) Operation of digit select switch ((s)+2, (s)+3)
   - Operation when carry-down switch (s)+2 is ON
     Every time the switch is pressed, the digit specification changes in the sequence of $10^3 \rightarrow 10^2 \rightarrow 10^1 \rightarrow 10^0 \rightarrow 10^3$.
   - When carry-over switch (s)+3 is ON
     Every time the switch is pressed, the digit specification changes in the sequence of $10^3 \rightarrow 10^0 \rightarrow 10^1 \rightarrow 10^2 \rightarrow 10^3$.
3) Operation of LED for selected digit display ((d2)+4 to (d2)+7)
   The specified digit is displayed by LED by strobe signals (d2)+4 to (d2)+7.
4) Operation of data change switch of digit unit ((s), (s)+1)
   Data changes in the digit specified by the "digit select switch."
   - Operation when increment input is ON
     Every time the switch is pressed, the content of (d1) changes in the sequence of 0 → 1 → 2 → ... → 8 → 9 → 0 → 1.
   - Operation when decrement input is ON
     Every time the switch is pressed, the content of (d1) changes in the sequence of 0 → 9 → 8 → 7 ... → 1 → 0 → 9.

   The content can be displayed in 7-segment display unit.
   Thus, by a series of operations, desired numeric values can be written into (d1) while observing the 7-segment display unit.

Cautions

1. Setting of parameter n
Refer to the parameter setting of SEGL instruction. However, the setting range is 0 to 3.

2. Output format of PLC
Use the PLC of transistor output type.

3. Scan time (operation period) and display timing
ARWS instruction is executed in synchronism with the scan time (operation cycle) of the PLC.
To execute a series of displays, 10 ms or more is needed in the scan time of the PLC. If less than 10 ms, you must use the constant scan mode and operate in scan time of 10 ms or more.

4. Number of bits occupied in device
1) The input of the device specified by (s) occupies 4 bits.

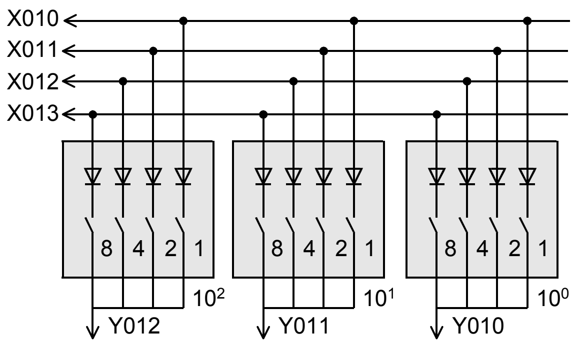
2) The output of the device specified by (d2) occupies 8 bits.

5. Limit of times of use of instruction
ARWS instruction can be used only once in the program. When using this instruction two or more times, use index function to program.

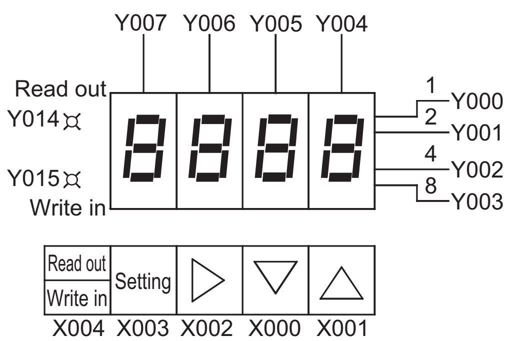
6. Object devices are limited.
▲1: FX3U FX3UC PLCs only are applicable.
However, index modifier (V, Z) is not applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

Program example

1. When changing the timer setting and displaying the current value
1) Specify the timer number by 3-digit digital switch



2) Setting of constant value of timer by arrow switch

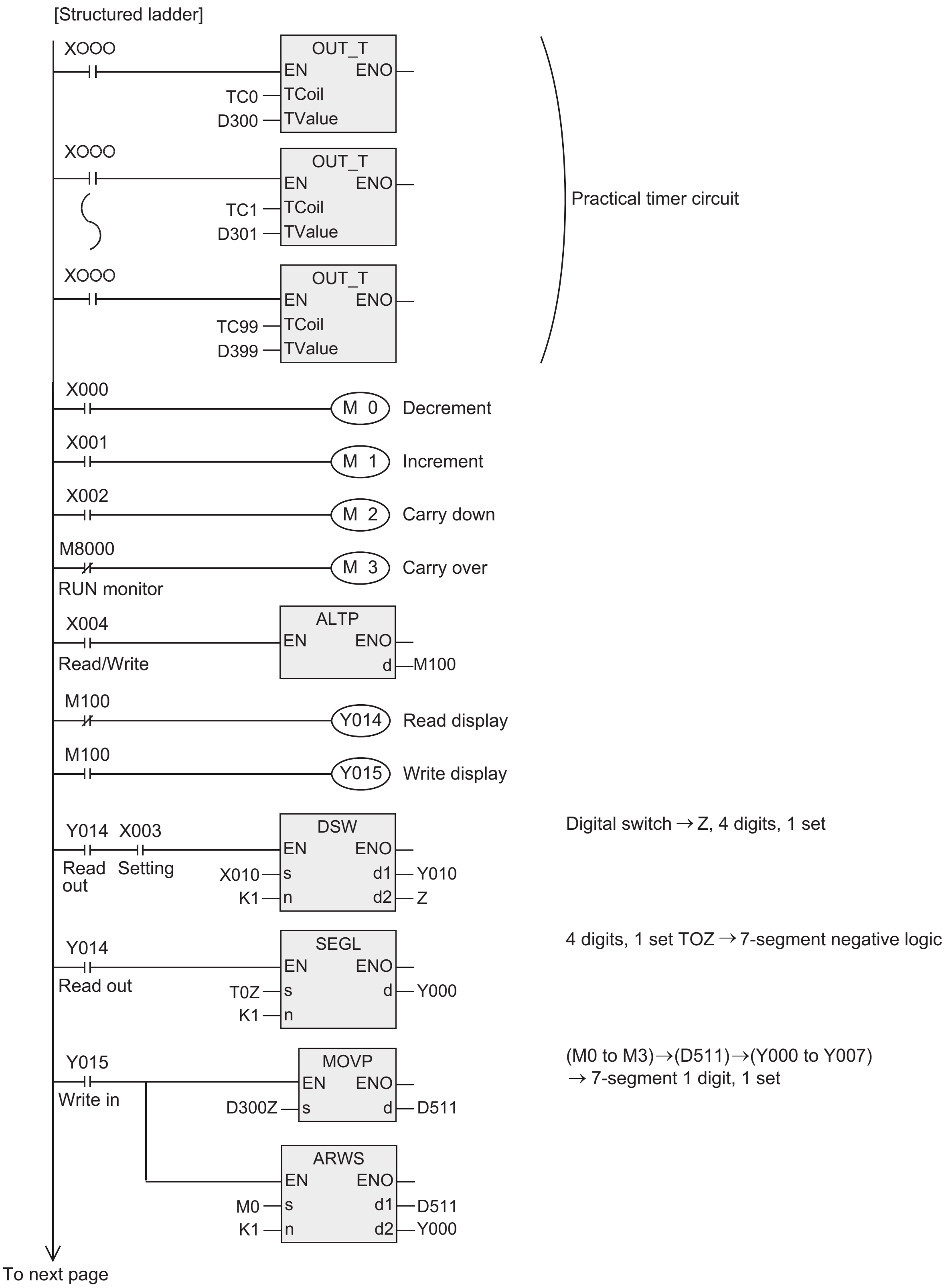


Operation explanation
Every time read/write key is pressed, the read and write LED is changed over and displayed.

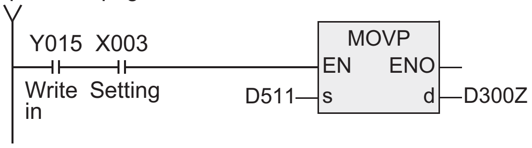
- When reading out
  Press the setting switch (X003) after setting the timer number by digital switch.

- When writing in
  Press the X003 by setting the numeric value while observing the 7-segment display by the arrow switch.

Program



From previous page



[ST]

```
OUT_T(XOOO, TC0, D300);
OUT_T(XOOO, TC1, D301);
    ⟆
OUT_T(XOOO, TC99, D399);

M0:= X000;
M1:= X001;
M2:= X002;
M3:= NOT M8000;
ALTP(X004, M100);
Y014:= NOT M100;
Y015= M100;
IF(Y014 AND X003) THEN DSW (TRUE, X010, K1, Y010, Z);
SEGL(Y014, T0Z, K1, Y000);
MOVP(Y015, D300Z, D511);
ARWS(Y015, M0, K1, D511, Y000);
IF(Y015 AND X003) THEN MOVP(TRUE, D511, D300Z);
```

7.8.7 ASC

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This instruction converts the 1-byte alphanumeric character string into ASCII code.
This is used when selecting and displaying plural messages in the external display unit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ASC	16 bits	Continuous	ASC: EN ENO, s d	ASC(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	1-byte alphanumerics of 8 characters entered from personal computer	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Head device for storing ASCII code	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices													Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)																							●*1	
(d)												●	●	●	▲1	▲1			●					

*1. In ASC instruction, double quotation marks (") are not needed before and after the character string specified by (s).

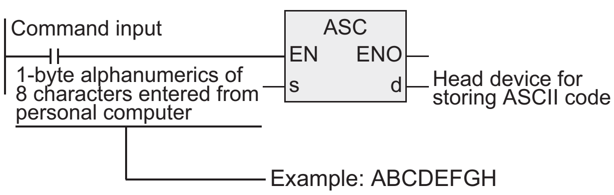
▲: Refer to "Cautions".

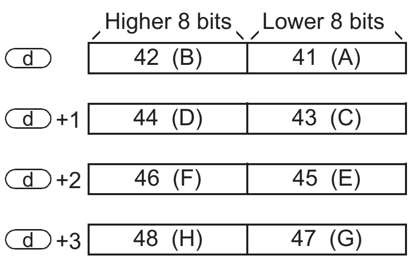
Function and operation explanation

1. 16-bit operation (ASC)

1-byte alphanumeric character string specified by (s) is converted into ASCII code, and transferred sequentially to the device specified by (d).

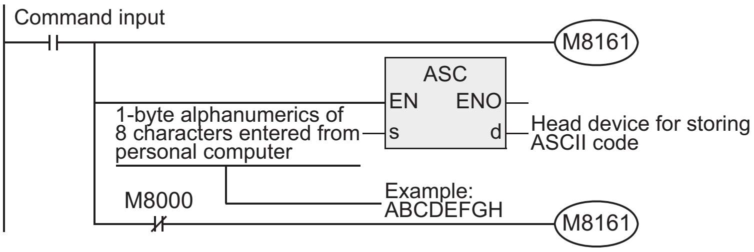
- In the device specified by (s), 1-byte characters of A to Z, 0 to 9, and symbols can be handled. (2-byte characters cannot be handled.)
  By the programming tool, the character string is entered when programming.
- In the device specified by (d), converted ASCII code is stored by 2 characters/1 byte each in the sequence of lower 8 bits and higher 8 bits.





Extension function

When the extension function is validated by turning ON M8161, the 1-byte alphanumeric character string of the device specified by (s) is converted into ASCII code, and sequentially transferred into the device specified by (d) only in the lower 8 bits (1 byte).
The extension function is not available when FXU PLC is V2.30 or earlier.



Higher 8 bits are H00.

	(d)		(d)
	Higher 8 bits	Lower 8 bits	Character string
(d)	00	41	A
(d)+1	00	42	B
(d)+2	00	43	C
(d)+3	00	44	D
(d)+4	00	45	E
(d)+5	00	46	F
(d)+6	00	47	G
(d)+7	00	48	H

Related devices

Device	Name	Content
M8161	Function extension flag	8-bit processing mode of ASC, RS, ASCI, HEX, CCD instruction OFF : Every 2 characters are stored in the sequence of lower 8 bits, higher 8 bits (2 characters/ word). ON : Characters are stored in lower 8 bits character by character (1 character/word).

Cautions

1. Number of bits occupied in device

1) When extension function is OFF
   - The device specified by (d) occupies the value of number of characters ÷ 2 bits. (The remainder is carried over.)
2) When extension function is ON
   - The device specified by (d) occupies the same number as the number of characters.

2. When using RS, ASCI, HEX, CCD

The extension function flag M8161 is a common flag used in other instructions.
When above instructions are used together with ASC instruction, you must execute the program of turning ON or OFF the M8161 immediately before the ASC instruction so as to avoid its effect.

3. Object devices are limited.

▲1: FX3U, FX3UC PLCs only are applicable.

7.8.8 PR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	○	×	×

Outline

This instruction performs parallel output of ASCII code data to the output (Y).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
PR	16 bits	Continuous	PR EN ENO s d	PR(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device for storing data of ASCII code.	String
Output variable	ENO	Execution state	Bit
	(d)	Head device for output data of ASCII code.	Bit

3. Applicable devices

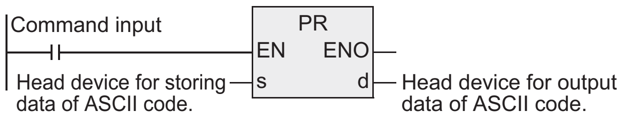
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)												●	●	●	▲1				●					
(d)		●																	●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (PR)

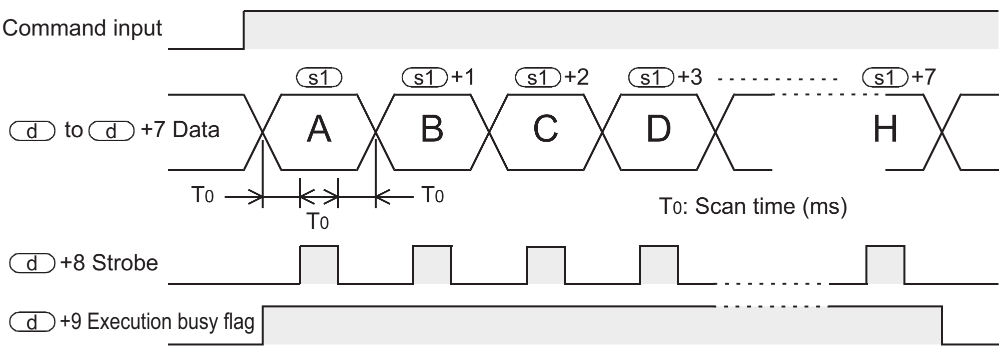
ASCII code stored in lower 8 bits (1 byte) of (s) to (s)+7 is issued to (d) to (d)+7 sequentially in time division character by character.



The following timing chart explains the ASCII code stored in the device specified by (s).
The transmission sequence begins from (s)="A", and ends with (s)+7="H", and 8 bytes are transmitted in total.

(s)	(s)+1	(s)+2	(s)+3	(s)+4	(s)+5	(s)+6	(s)+7
A(H41)	B(H42)	C(H43)	D(H44)	E(H45)	F(H46)	G(H47)	H(H48)

2. Timing chart



Type of output signal

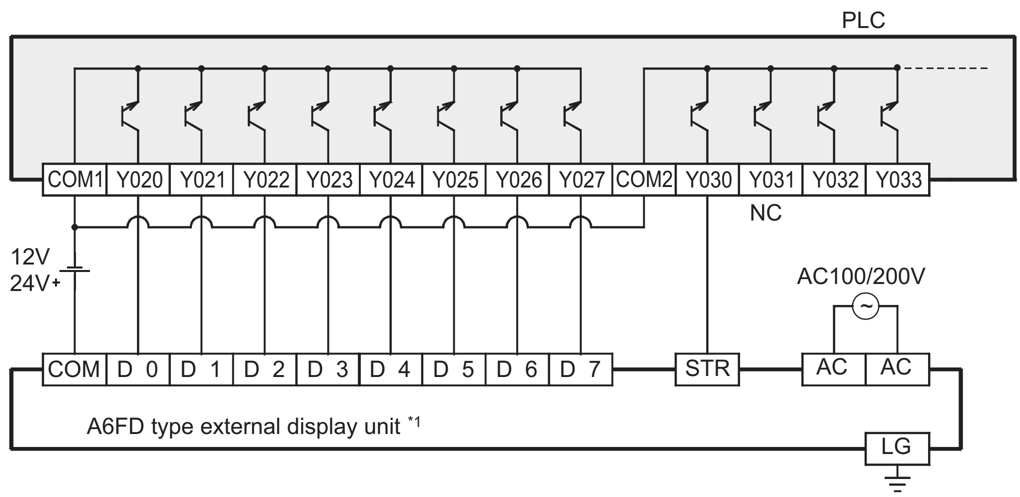
- (d) to (d)+7 : Transmission output (d) is the lower bit side, and (d)+7 is the higher bit side.
- (d)+8 : Strobe signal
- (d)+9 : Execution busy flag Operation conforms to the timing chart above.

Extension function

1. 16-byte serial output

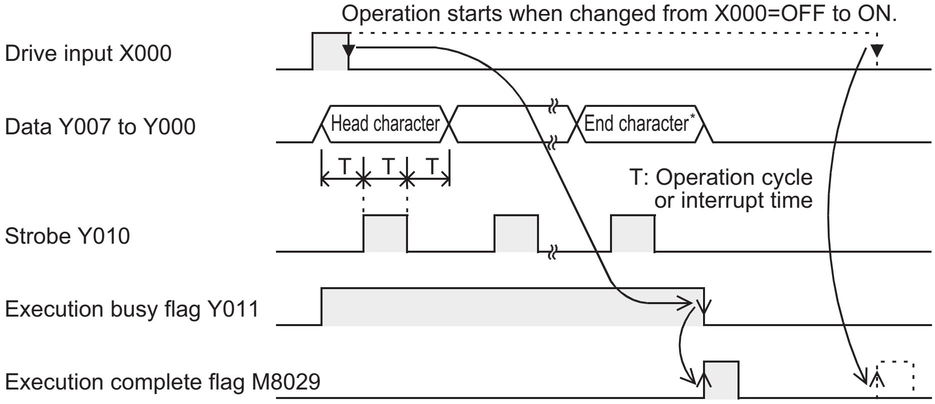
By ON/OFF control of special auxiliary relay M8027, the number of characters of output varies in every two times of instruction drive. In the case of M8027=OFF, the operation is 8-byte serial output (fixed in 8 characters), and in the case of M8027=ON, it is 16-byte serial output (1 to 16 characters).
An example of display of 16 characters or less (1 character/byte) is explained by referring to a display device (example: A6FD type external display unit*1).
The display data is supposed to be stored in hexadecimal code in D300 to D307, for example.

1) A6FD type external display unit*1 connection example
The PLC shown below is an example of FX2N-16EYT (sink output) connected to FX3U-32M□.



*1. A6FD type external display unit is out of production since November 2002.

2) Timing chart (When M8027=ON)



* If H00 (NUL code) exists in the data (in 16 characters), one character before H00 (NUL code) is the end character.

Related devices

Device	Name	Content
M8027*1	PR mode	OFF : 8-byte serial output (fixed in 8 characters) ON : 16-byte serial output (1 to 16 characters)

*1. Cleared when changed from RUN to STOP.

Cautions

1. Command input and operation of instruction

Command input=ON : Even in the midst of continuous ON or pulse instruction execution, when the output of one cycle is over, the execution is terminated.
M8029 operates only when M8027=ON.
Command input=OFF: All outputs are OFF.

2. Relation with scan time (operation time)

This instruction is executed in synchronism with the scan time.
If the scan time is short, it is driven by constant scan mode, or if too long, it is driven by using the timer interrupt mode.

3. Output of PLC

Use the transistor output for the output of the PLC.

4. If 00H (NUL) exists in the data (when M8027=ON)

The instruction execution is completed, and the remaining data is not issued.
M8029 remains ON during 1 operation cycle.

5. Object devices are limited.

▲1: FX3U, FX3UC PLCs only are applicable.

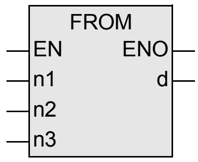
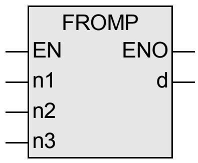
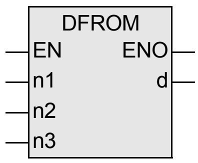
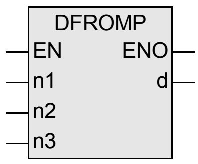
7.8.9 FROM

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	×	△	○	×

Outline

This instruction reads out the content of buffer memory (BFM) of special extension unit/block to the PLC.
If a large quantity of buffer memory (BFM) data is read out in batch by using this instruction, a watchdog timer error may occur. When there is no bad influence for the control if the data is divided and read out, you can use RBFM instruction.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
FROM	16 bits	Continuous		FROM(EN, n1, n2, n3, d);
FROMP	16 bits	Pulse		FROMP(EN, n1, n2, n3, d);
DFROM	32 bits	Continuous		DFROM(EN, n1, n2, n3, d);
DFROMP	32 bits	Pulse		DFROMP(EN, n1, n2, n3, d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	n1	Unit No. of special extension unit/block	ANY16	ANY32*1
	n2	Transfer origin buffer memory (BFM) number	ANY16	ANY32*1
	n3	Number of transfer points	ANY16	ANY32*1
Variable	ENO	Execution state	Bit	
	d	Transfer destination device • 16-bit operation: n^3 points occupied • 32-bit operation: $2 \times n^3$ points occupied	ANY16	ANY32

*1. In the case of FXU, FX1N, FX2N, FX2C, FX1NC, FX2NC series, the data type of DFROM, DFROMP of 32-bit operation is ANY16.

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
n1														●	▲1					●	●			
n2														●	▲1					●	●			
d									●	●	●	●	●	●	▲1		●	●	●					
n3														●	▲1					●	●			

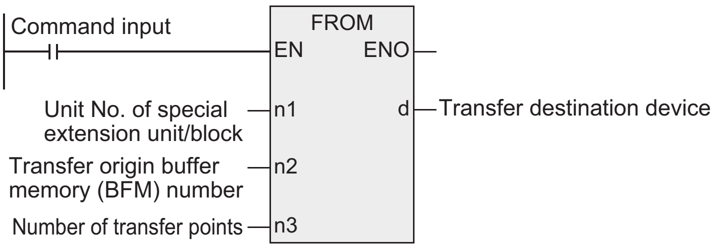
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (FROM, FROMP)

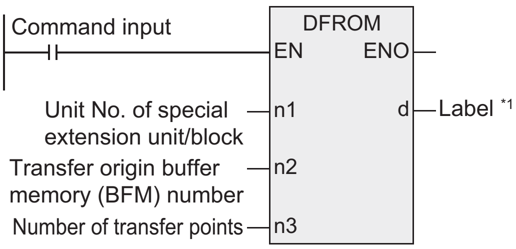
→ **As for the common terms of FROM/TO instruction, refer to the Common terms of FROM/TO instruction (detail)**

From special extension (BFM) to PLC (word device)
16-bit data of n3 points starting from buffer memory (BFM) n2 in the special extension unit/block of unit number n1 is transferred (read out) by the portion of n3 points starting from the device specified by d in the PLC.



2. 32-bit operation (DFROM, DFROMP)

From special extension (BFM) to PLC (word device)
32-bit data of n3 points starting from buffer memory (BFM) [n2+1, n2] in the special extension unit/block of unit number n1 is transferred (read out) by the portion of n3 points starting from the device specified by d in the PLC.



*1. This defines the transfer destination device.

Related devices

Device	Name	Content
M8028	Interrupt permit flag	Prohibit/permit interruption during execution of FROM/TO instruction. → **As for the detail, refer to the following pages "Interrupt accept during execution of FROM/TO (M8028)."** OFF : Interrupt prohibit (interrupt executed after FROM/TO instruction process) ON : Interrupt permit

Cautions

1) Bit device digits to be specified by (d) should be K1 to K4 in the case of 16-bit operation instruction, and K1 to K8 in the case of 32-bit operation.
2) When handling 32-bit data in structured program, unlike the simple project, you cannot specify the 16-bit device directly. Use the label when handling 32-bit data.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   Use the global label when specifying the device.
3) FXU PLC supports the instruction in V2.10 or later.
4) FX0N PLC does not support the instruction of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
5) Object devices are limited.
   ▲1: FX3U, FX3UC, FX3G PLCs only are applicable.

Program example

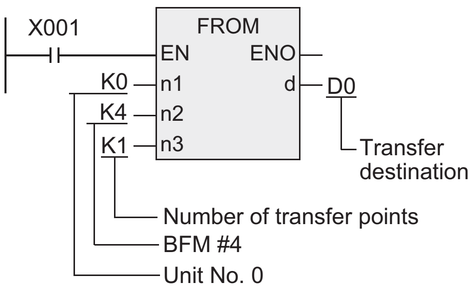
By using direct specification[*1] of FROM instruction or buffer memory, you can transfer (read out) the content of buffer memory (BFM) of special extension unit/block to the digit specification of data register (D), extension register (R), or auxiliary relay (M).

*1. FX3U, FX3UC PLCs only are available.

Example) Program for reading out BFM#4 (abnormal station information) of CC-Link/LT master (unit No. 0 fixed) built in FX3UC-32MT-LT(-2) to D0.

- In the case of FROM instruction

[Structured ladder]

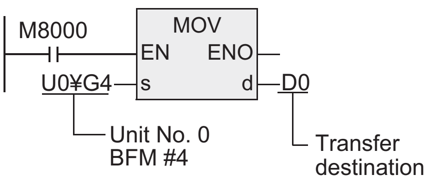


[ST]

FROM(X001, K0, K4, K1, D0);

- In the case of MOV instruction

[Structured ladder]



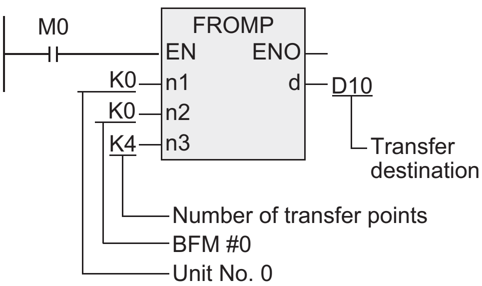
[ST]

MOV(M8000, U0¥G4, D0);

Program for reading out BFM#0 to #3 (remote station connection information) of CC-Link/LT master (unit No. 0 fixed) built in FX3UC-32MT-LT(-2) to D10 to D13.

- In the case of FROM instruction

[Structured ladder]

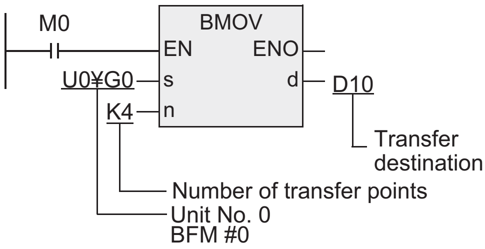


[ST]

FROMP(M0, K0, K0, K4, D10);

- In the case of BMOV instruction

[Structured ladder]



[ST]

BMOV(M0, U0¥G0, K4, D10);

Common terms of FROM/TO instruction (detail)

Specification content of operand

1. Unit number n1 of special extension unit/block

Unit number is used for specifying which equipment is the object of working for FROM/TO instruction.
Setting range: K0 to K7

Unit No. 0 built-in CC-Link/LT		Unit No. 1	Unit No. 2		Unit No. 3
FX3UC-32MT-LT(-2) Basic unit	Input/output extension block	Special extension block	Special extension block	Input/output extension block	Special extension block

The unit number is assigned automatically to the special extension unit/block connected to the PLC.
The unit number is given from the one closest to the basic unit in the sequence of No. 0, No. 1, No.2, etc.
In the case of the FX3UC-32MT-LT(-2) PLC, since the CC-Link/LT master is built in, the unit number is given from the one closest to the basic unit in the sequence of No. 1, No.2, No. 3, etc.

2. Buffer memory (BFM) number "n2"

In the special extension unit/block, 16-bit RAM is built in, and it is used as the buffer memory.
The buffer memory numbers are #0 to #32766, and the content is determined depending on the control purpose of each equipment.
Setting range: K0 to K32766

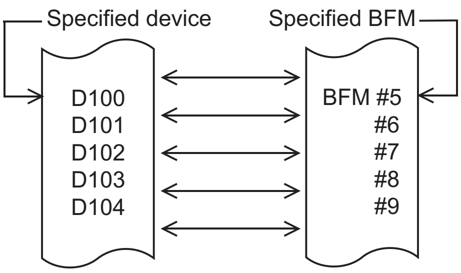
- When BFM is handled in 32-bit instruction, the specified BFM is lower 16 bits, and the BFM of the next number is higher 16 bits.

Higher 16 bits	Lower 16 bits	
BFM #10	BFM #9	←Specified BFM number

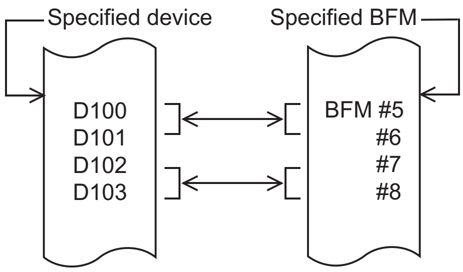
3. Transfer points "n3"

Setting range: K1 to K32767

Number of transfer words is specified by n3.
"n =2" in a 16-bit instruction indicates the same meaning with "n = 1" in a 32-bit instruction.



16-bit operation when n=5



32-bit operation when n=2

Interrupt accept during execution of FROM/TO (M8028)

1. When M8028=OFF

Interrupt is prohibited automatically during execution of FROM/TO instruction, and input interrupt or timer interrupt is not executed.
An interrupt occurring in this period is executed immediately after completion of execution of FROM/TO instruction.
FROM/TO instruction can be used also during the interrupt program.

2. When M8028=ON

When an interrupt occurs during execution of FROM/TO instruction, the execution is suspended, and the interrupt program is executed.
However, FROM/TO instruction cannot be used during interrupt program.

Handling in the event of occurrence of watchdog timer error

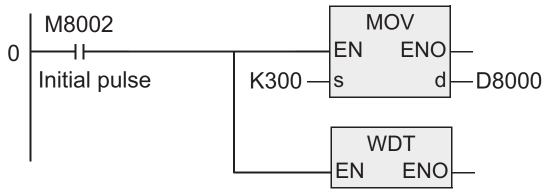
1. Cause of occurrence of watchdog timer error

Watchdog timer error may occur in the following cases.

1) When many special extension equipments are connected.
   In a configuration of a large number of connected units of special extension equipments (positioning, cam switch, link, analog, etc.), the initializing time of the buffer memory executed in PLC RUN mode is long, and the operation time is extended, and a watchdog timer error may occur.
2) When FROM/TO instructions are driven simultaneously.
   When many FROM/TO instructions are executed, or when multiple buffer memories are transferred, the operation time is extended, and a watchdog timer error may occur.

2. Countermeasure

1) Method of using RBFM, WBFM instruction
   → **As for BFM divided reading [RBFM], refer to section 7.25.1.**
   → **As for BFM divided writing [WBFM], refer to section 7.25.2.**
2) Method of changing the time of watchdog timer
   The detection time of the watchdog timer can be changed by rewriting the content of D8000 (watchdog timer time).
   By entering the following program, the subsequent sequence programs are monitored by the new watchdog timer time.



Watchdog timer time 300ms

Watchdog timer refresh
When WDT instruction is not programmed, the value of D8000 is valid during END process.

3) Change of execution timing of FROM/TO instruction
   Please shorten the operation time by deviating the execution of FROM/TO instruction.

Handling of special extension unit/block

As for the connection method of special extension unit/block, the number of units allowed to be connected, and handling of input and output numbers, please refer to the manual of the PLC main body or the manual of each special extension unit/block.

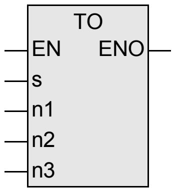
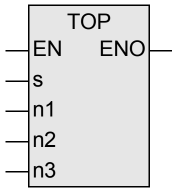
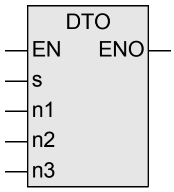
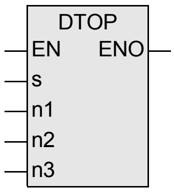
7.8.10 TO

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	×	△	○	×

Outline

This instruction writes data from PLC into the buffer memory (BFM) of special extension unit/block.
By this instruction, when data is written into multiple buffer memories (BFM) in batch a watchdog timer error may occur. When there is no bad influence for the control if the data is divided and written in, you can use WBFM instruction.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
TO	16 bits	Continuous		TO(EN, s, n1, n2, n3);
TOP	16 bits	Pulse		TOP(EN, s, n1, n2, n3);
DTO	32 bits	Continuous		DTO(EN, s, n1, n2, n3);
DTOP	32 bits	Pulse		DTOP(EN, s, n1, n2, n3);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Transfer source data or device • 16-bit operation: n3 points occupied • 32-bit operation: 2 × n3 points occupied	ANY16	ANY32
	n1	Unit No. of special extension unit/block (Sequentially from the right side of basic unit, K0 to K7)	ANY16	ANY32*1
	n2	Transfer destination buffer memory (BFM) number	ANY16	ANY32*1
	n3	Number of transfer points	ANY16	ANY32*1
Output variable	ENO	Execution state	Bit	

*1. In the case of FXU, FX1N, FX2N, FX2C, FX1NC, FX2NC series, the data type of DTO, DTOP of 32-bit operation is ANY16.

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
n1														●	▲1					●	●			
n2														●	▲1					●	●			
s								●	●	●	●	●	●	●	▲1		●	●	●	●	●			
n3														●	▲1					●	●			

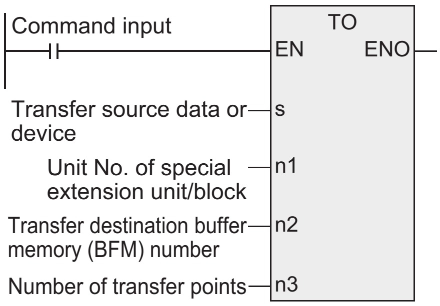
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (TO, TOP)

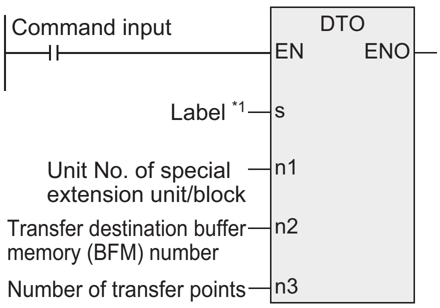
→ **As for common terms of FROM/TO instruction, refer to section 7.8.9.**

From PLC (word device) to special extension (BFM)
You can transfer (write) 16-bit data for the portion of n3 points starting from buffer memory (BFM) n2 in special extension unit/block of unit number n1, and for the portion of n3 points starting from the device specified by (s) in PLC.



2. 32-bit operation (DTO, DTOP)

From PLC (word device) to special extension (BFM)
You can transfer (write) 32-bit data for the portion of n3 points starting from buffer memory (BFM) [n2+1, n2] in special extension unit/block of unit number n1, and for the portion of n3 points starting from the device specified by (s) in PLC.



*1. This defines transfer destination data or device.

Related devices

Device	Name	Content
M8028	Interrupt permit flag	Prohibit/permit interruption during execution of FROM/TO instruction. → **As for detail, refer to Subsection 7.8.9 "Interrupt accept during execution of FROM/TO (M8028)"** OFF : Interrupt prohibit (interrupt executed after FROM/TO instruction process) ON : Interrupt permit

Cautions

1) About bit device digit specification to be specified by (s)
   Specify K1 to K4 in the case of 16-bit operation instruction, or K1 to K8 in the case of 32-bit operation.
2) When handling 32-bit data in structured program, unlike the simple project, you cannot specify the 16-bit device directly. Use the label when handling 32-bit data.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   Use the global label when specifying the device.
3) FXU PLC supports the instruction in V2.10 or later.
4) FX0N PLC does not support the instruction of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
5) Object devices are limited.
   ▲1: FX3U, FX3UC, FX3G PLCs only are applicable.

Program example

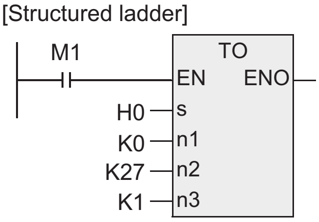
By using direct specification[*1] of TO instruction or buffer memory, you can write in (transfer) the digit specification of data register (D), extension register (R), or auxiliary relay (M) or constants (K, H) to the buffer memory (BFM) of special extension unit/block.

*1. FX3U, FX3UC PLCs only are available.

Example) Program for writing "H0" in BFM #27 (instruction) of CC-Link/LT master (unit No. 0 fixed) built in the FX3UC-32MT-LT(-2).

- In the case of TO instruction

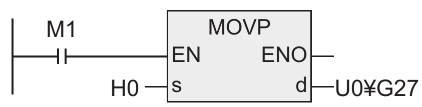
[Structured ladder]



[ST]
TO(M1, H0, K0, K27, K1);

- In the case of MOV instruction

[Structured ladder]



[ST]
MOVP(M1, H0, U0¥G27);

7.9 External Device (optional device)

7.9.1 RS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	△	△	×

Outline

This instruction sends and receives data in no-protocol communication by way of a serial port (only the ch1) in accordance with RS-232C or RS-485 provided in the main unit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RS	16 bits	Continuous	RS —EN ENO— —s d— —m —n	RS(EN, s, m, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device storing data to be sent	ANY16
	m	Number of bytes of data to be sent	ANY16
	n	Head device storing received data when receiving is completed	ANY16
Output variable	ENO	Execution state	Bit
	d	Number of bytes to be received	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	▲1				●					
m														●	▲1					●	●			
n														●	▲1				●					
d														●	▲1					●	●			

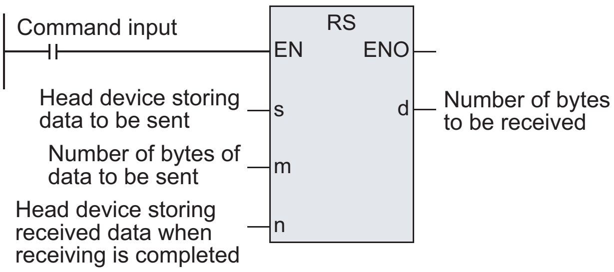
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (RS)

This instruction sends and receives data in no-protocol communication by way of serial ports in accordance with RS-232C or RS-485 provided in the main unit.

→ **For detailed explanation, refer to the Data Communication Edition manual.**



Related devices

→ **For detailed explanation, refer to the Data Communication Edition manual.**

Device	Name
M8063	Serial communication error 1
M8121	Sending wait flag
M8122	Sending request
M8123	Receiving complete flag
M8124*1	Carrier detection flag
M8129*2	Time-out check flag
M8161	8-bit processing mode

Device	Name
D8120	Communication format setting
D8122	Remaining number of data to be sent
D8123	Monitor for number of received data
D8124	Header
D8125	Terminator
D8129*2	Time-out time setting
D8063	Error code number of serial communication error 1
D8405*3	Communication parameter display
D8419*3	Operation mode display

*1. Not supported by the FX0N PLC.

*2. FXU, FX2C PLCs are applicable at Ver.3.30 or later.

*3. Not supported by the FX1S, FX1N, FX2N, FX1NC, FX2NC, FXU, FX2C and FX0N PLCs.

System configuration

To use this instruction, it is necessary to attach one of the products shown in the table below to the main unit.

→ **For the system configuration, refer to the respective PLC Hardware Edition manual.**
→ **For detailed explanation, refer to the Data Communication Edition manual.**

Differences between RS instruction and RS2 instruction

RS2 instruction is not supported by the FX1S, FX1N, FX2N, FX1NC, FX2NC, FXU and FX0N PLCs.

Item	RS2 instruction	RS instruction	Remarks
Header size	1 to 4 characters (bytes)	Up to 1 character (byte)	For the RS2 instruction, up to 4 characters (bytes) can be specified as a header or terminator.
Terminator size	1 to 4 characters (bytes)	Up to 1 character (byte)	
Attachment of check sum	The check sum can be automatically attached.	The check sum should be attached by a user program.	For the RS2 instruction, the check sum can be automatically attached to the sent and received data. In this case, however, make sure to use a terminator with the communication frame to be sent and received.
Used channel number	ch1	ch0, ch1, ch2	In RS2 instruction is as follows: Ch2 is not available in 14-point and 24-point type FX3G PLCs. Ch0 is available only in FX3G PLCs.

Cautions

→ **For other cautions, refer to the Data Communication Edition manual.**

1. For FX3U, FX3UC and FX3G PLCs.

1) RS instruction can be used for ch1 only (cannot be used for ch2).
Ch2 is not provided for the FX1S, FX1N, FX2N, FX1NC, FX2NC, FXU, FX2C and FX0N PLCs.

2) Do not drive two or more RS and/or RS2 instructions for the same port at the same time.
FX1S, FX1N, FX2N, FX1NC, FX2NC, FXU, FX2C and FX0N PLCs does not support the RS2 instruction.

2. FXU PLC supports the instruction at V3.07 or later.

3. FX0N PLC supports the instruction at V1.20 or later.

4. Number of bytes of data to be sent(m), Number of bytes to be received(n)

1) For FX3U, FX3UC, FX3G, FX2N and FX2NC PLCs
m, n: 0 to 4096 points
(However, "m + n" should not be more than 8000 points in FX2N and FX2NC PLCs.)

2) For FX1S, FX1N, FX1NC, FX0N, FXU and FX2C PLCs
m, n: 0 to 256 points

5. Object devices are limited.

▲1: FX3U, FX3UC, FX3G PLCs only are applicable.

7.9.2 PRUN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	○	×	×

Outline

This instruction handles the device number specified by (s) and (d) specified by digits as octal number, and transfers data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
PRUN	16 bits	Continuous	PRUN EN ENO s d	PRUN(EN, s, d);
PRUNP	16 bits	Pulse	PRUNP EN ENO s d	PRUNP(EN, s, d);
DPRUN	32 bits	Continuous	DPRUN EN ENO s d	DPRUN(EN, s, d);
DPRUNP	32 bits	Pulse	DPRUNP EN ENO s d	DPRUNP(EN, s, d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Specification of digits (the lowest digit of specified device number is 0.)	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Transfer destination device (the lowest digit of specified device number is 0.)	ANY16	ANY32

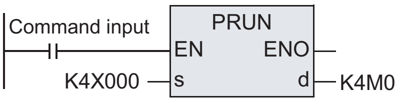
3. Applicable devices

Operand type	Bit Devices: System User							Word Devices: Digit designation				Word Devices: System User				Special Unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●		●									●					
(d)									●	●									●					

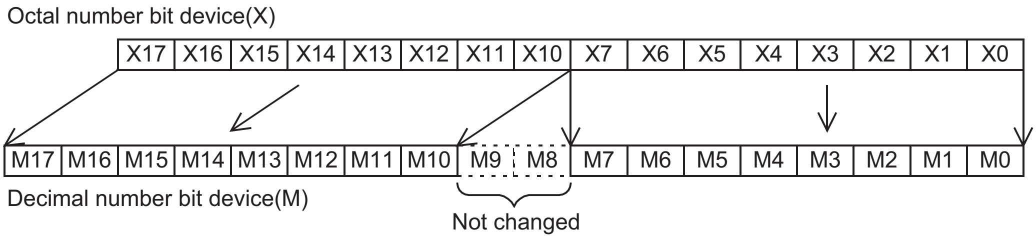
Function and operation explanation

1. 16-bit operation(PRUN, PRUNP)

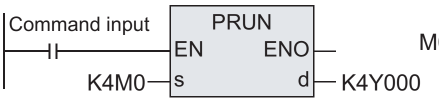
From octal number bit device to decimal number bit device



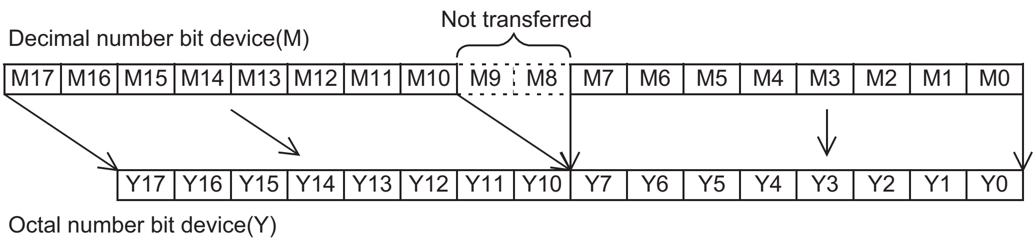
X000 ~ X017 → M0 ~ M7, M10 ~ M17



From decimal number bit device to octal number bit device

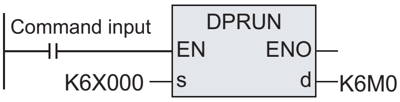


M0 ~ M7, M10 ~ M17 → Y0 ~ Y17

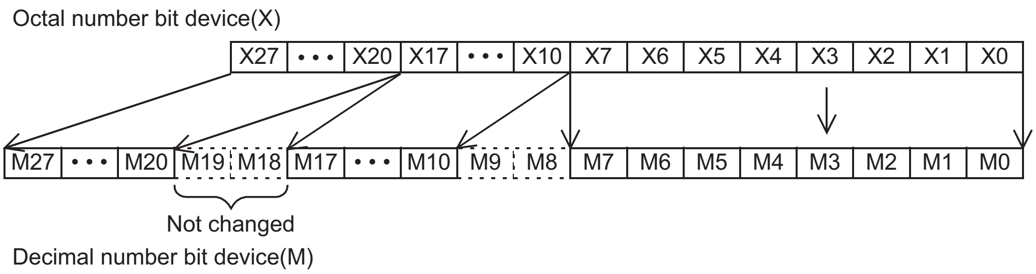


2. 32-bit operation(DPRUN, DPRUNP)

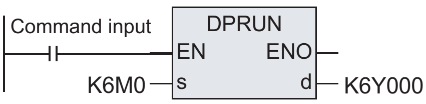
From octal number bit device to decimal number bit device



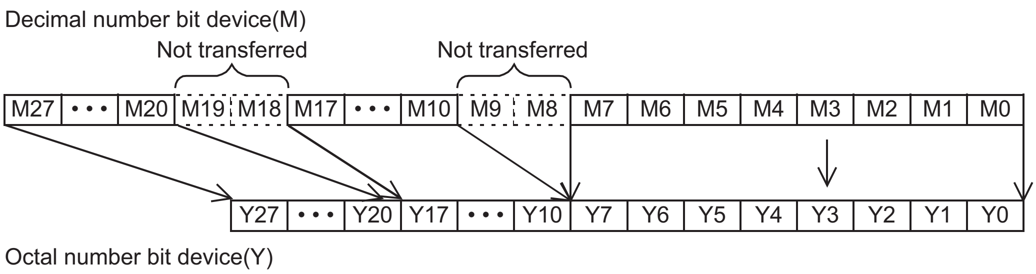
X000 ~ X027 → M0 ~ M7, M10 ~ M17, M20 ~ M27



From decimal number bit device to octal number bit device



M0 ~ M7, M10 ~ M17, M20 ~ M27 →Y000 ~ Y027



7.9.3 ASCI

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	△	△	×

Outline

This instruction converts HEX code into ASCII code.
Also available are BINDA instruction for converting BIN data into ASCII code, and DESTR instruction for converting binary floating decimal point data into ASCII code.

→ **As for BINDA instruction, refer to section 7.23.6.**
→ **As for DESTR instruction, refer to section 7.12.4.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
ASCI	16 bits	Continuous	ASCI EN ENO s d n	ASCI(EN, s, n, d);
ASCIP	16 bits	Pulse	ASCIP EN ENO s d n	ASCIP(EN, s, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device in which HEX code to be converted is stored	ANY16
	(n)	Number of characters in HEX code to be converted (number of digits)	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Head device for storing converted ASCII code	ANY16

3. Applicable devices

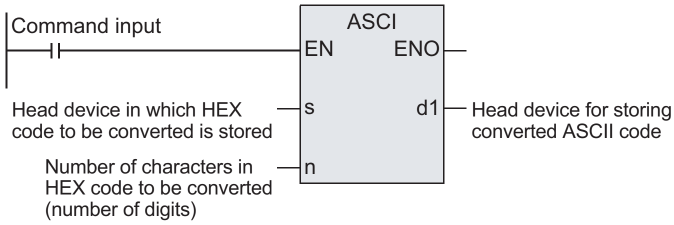
Operand type	Bit Devices: System User							Word Devices: Digit designation				Word Devices: System User				Word Devices: Special Unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2			●					
(n)														●	▲1					●	●			

▲: Refer to "Cautions".

Function and operation

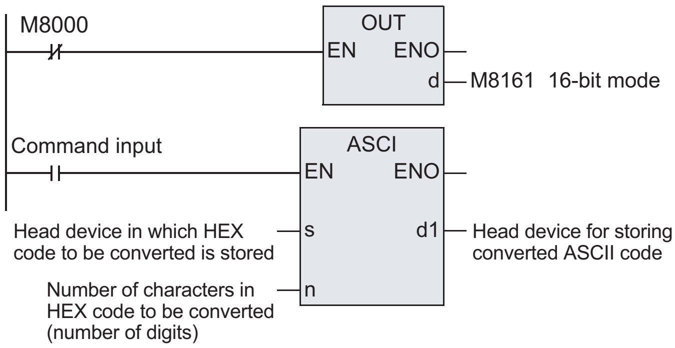
1. **16-bit operation(ASCI/ASCIP)**

   Of the HEX code stored after the device specified by (s), n characters (digits) are converted into ASCII code, and stored in the device after the one specified by (d).
   In this instruction, the mode usable when converting includes 16-bit mode and 8-bit mode. As for the operation of each mode, refer to the following pages.

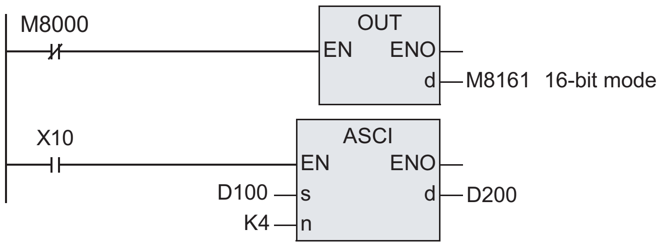


2. **<16-bit conversion mode> When M8161=OFF(M8161 is used commonly with RS, HEX, CCD, CRC instructions)**

   Each digit of HEX data stored after the device specified by (s) is converted into ASCII code, and transferred to lower and higher 8 bits (bytes) each in each device after the one specified by (d). The number of digits (characters) to be converted is specified by n.
   The device specified by (d) is divided into lower 8 bits and higher 8 bits, and ASCII data is stored.
   M8161 is used commonly with RS, HEX, CCD, CRC instructions. When using in 16 bits, you must keep always OFF.
   M8161 is cleared when changed from RUN to STOP.



Operation
In the case of the program shown below, the conversion is executed as follows.



Devices after (s)
(D100)=0ABCH
(D101)=1234H
(D102)=5678H

Number of digits (characters) specified and result of conversion

n / d	K1	K2	K3	K4	K5	K6	K7	K8	K9
D 200 lower	C)	B)	A)	0)	4)	3)	2)	1)	8)
D 200 higher		C)	B)	A)	0)	4)	3)	2)	1)
D 201 lower			C)	B)	A)	0)	4)	3)	2)
D 201 higher				C)	B)	A)	0)	4)	3)
D 202 lower					C)	B)	A)	0)	4)
D 202 higher						C)	B)	A)	0)
D 203 lower			Not changed				C)	B)	A)
D 203 higher								C)	B)
D 204 lower									C)

Bit composition in the case of n=K4

D 100=ABCH

0	0	0	0	1	0	1	0	1	0	1	1	1	1	0	0
0				A				B				C			

D 200

0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	0
A)→41H								0)→30H							

D 201

0	1	0	0	0	0	1	1	0	1	0	0	0	0	1	0
C)→43H								B)→42H							

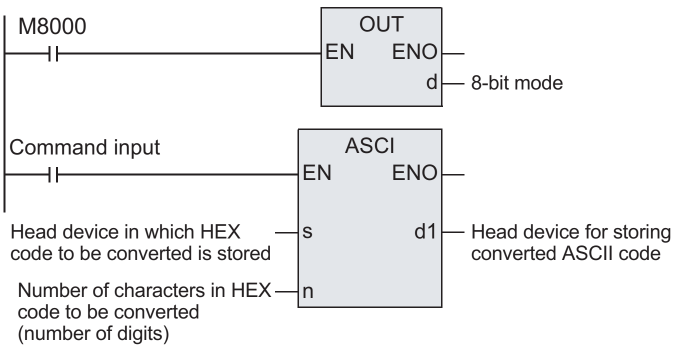
ASCII code

0)=30H	1)=31H	5)=35H
A)=41H	2)=32H	6)=36H
B)=42H	3)=33H	7)=37H
C)=43H	4)=34H	8)=38H

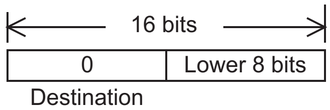
- When issuing as BCD data by printer or the like, before execution of this instruction, you must convert from BIN to BCD.

3. <8-bit conversion mode> When M8161=ON (M8161 is used commonly with RS, HEX, CCD, CRC instructions)

Each digit of HEX data stored after the device specified by (s) is converted into ASCII, and stored in lower 8 bits (bytes) in each device after the one specified by (d). The number of digits (characters) to be converted is specified by n.
In the device specified by (d), higher 8 bits are 0.
M8161 is used commonly with RS, HEX, CCD, CRC instructions. When using in 8 bits, you must keep always ON.
M8161 is cleared when changed from RUN to STOP.

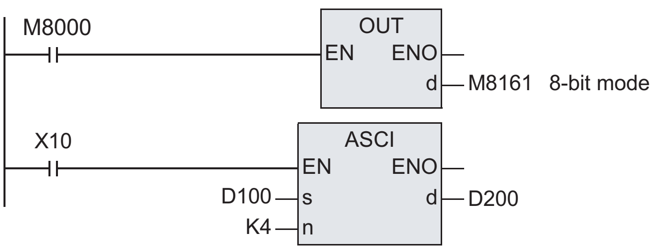


When M8161=ON, 8-bit mode is established, and the conversion is executed as follows.



Operation

In the case of the program shown below, the conversion is executed as follows.



Devices after (s)
(D100)=0ABCH
(D101)=1234H
(D102)=5678H

Number of digits (characters) specified and result of conversion

(n) / (d)	K1	K2	K3	K4	K5	K6	K7	K8	K9
D 200	C)	B)	A)	0)	4)	3)	2)	1)	8)
D 201		C)	B)	A)	0)	4)	3)	2)	1)
D 202			C)	B)	A)	0)	4)	3)	2)
D 203				C)	B)	A)	0)	4)	3)
D 204					C)	B)	A)	0)	4)
D 205						C)	B)	A)	0)
D 206		Not changed					C)	B)	A)
D 207								C)	B)
D 208									C)

Bit composition in the case of n=K2

D 100=0ABCH

0	0	0	0	1	0	1	0	1	0	1	1	1	1	0	0
0				A				B				C			

ASCII code of D200=B=42H

0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
								4				2			

ASCII code of D201=C=43H

0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1
								4				3			

ASCII code

0)=30H	1)=31H	5)=35H
A)=41H	2)=32H	6)=36H
B)=42H	3)=33H	7)=37H
C)=43H	4)=34H	8)=38H

- When issuing as BCD data by printer or the like, before execution of this instruction, you must convert from BIN to BCD.

Cautions

1) FXU PLC supports the instruction at V3.07 or later.
   FX0N PLC supports the instruction at V1.20 or later.
2) FX0N PLC does not support the instruction of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
3) Object devices are limited.
   ▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

7.9.4 HEX

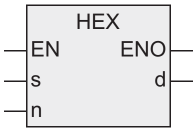
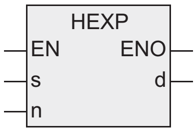
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	△	△	×

Outline

This instruction converts ASCII code into HEX code.
Also available are DABIN instruction for converting ASCII code into BIN data, and DEVAL instruction for converting ASCII code into binary floating decimal point data.

→ As for DABIN instruction, refer to section 7.23.5.
→ As for DEVAL instruction, refer to section 7.12.5.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
HEX	16 bits	Continuous		HEX(EN, s, n, d);
HEXP	16 bits	Pulse		HEXP(EN, s, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device in which ASCII code to be converted is stored	ANY16
	n	Number of characters in ASCII code to be converted (number of bytes)	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device for storing converted HEX code	ANY16

3. Applicable devices

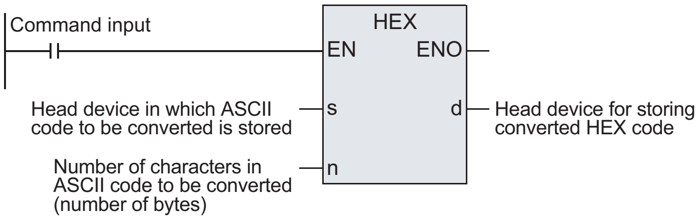
Operand type	Bit Devices: System User							Word Devices: Digit designation				Word Devices: System User				Word Devices: Special Unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲1	▲2			●	●	●			
d									●	●	●	●	●	●	▲1	▲2	●	●	●					
n														●	▲1					●	●			

▲: Refer to "Cautions".

Function and operation explanation

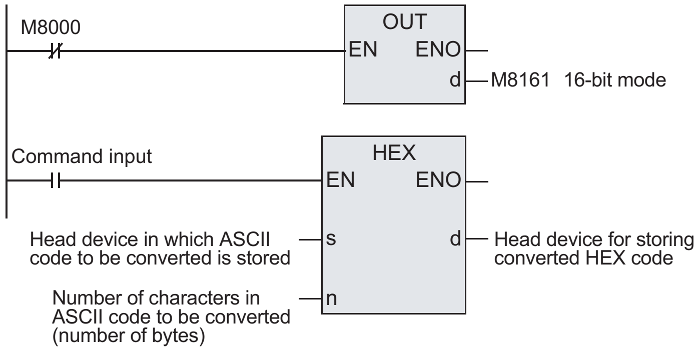
1. 16-bit operation(HEX/HEXP)

Of the ASCII code stored after the device specified by (s), n characters are converted into HEX code, and stored in the device after the one specified by (d). In this instruction, the mode usable when converting includes 16-bit mode and 8-bit mode. As for the operation of each mode, refer to the following pages.

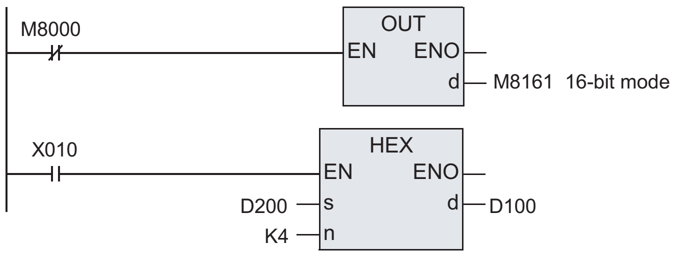


2. <16-bit conversion mode> When M8161=OFF(M8161 is used commonly with RS, ASCI, CCD, CRC instructions)

ASCII characters stored in higher and lower 8 bits (bytes) in the device specified by (s) are converted into HEX data, and transferred to the device specified by (d) in every four digits. The number of characters to be converted is specified by n.
M8161 is used commonly with RS, ASCI, CCD, CRC instructions. When using in 16 bits, you must keep always OFF.
M8161 is cleared when changed from RUN to STOP.



Operation
In the case of the program shown below, the conversion is executed as follows.



Conversion source data

(s)	ASCII code	HEX conversion
D 200 lower	30H	0
D 200 higher	41H	A
D 201 lower	42H	B
D 201 higher	43H	C
D 202 lower	31H	1
D 202 higher	32H	2
D 203 lower	33H	3
D 203 higher	34H	4
D 204 lower	35H	5

Number of characters specified and result of conversion

"." is 0.

(d) / (n)	D 102	D 101	D 100
1	Not changed	Not changed	...0H
2			..0AH
3			.0ABH
4			0ABCH
5		...0H	ABC1H
6		..0AH	BC12H
7		.0ABH	C123H
8		0ABCH	1234H
9	...0H	ABC1H	2345H

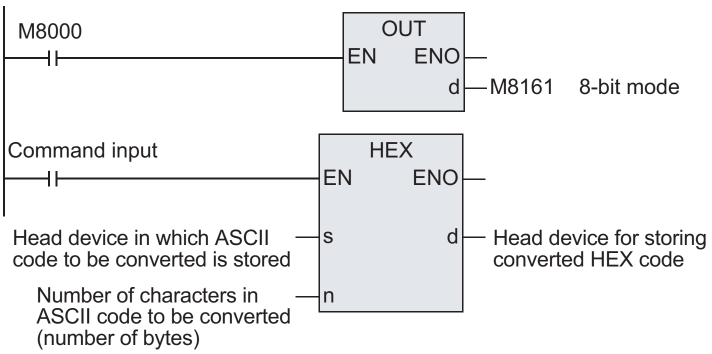
In the case of n=K4

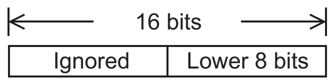
	b15															b0
D 200	0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	0
	41H→ A)								30H→ 0)							
D 201	0	1	0	0	0	0	1	1	0	1	0	0	0	0	1	0
	43H→ C)								42H→ B)							
D 100	0	0	0	0	1	0	1	0	1	0	1	1	1	1	0	0
	0				A				B				C			

- When the input data is BCD, after execution of this instruction, you must convert from BCD to BIN.
- In the case of HEX instruction, if the data stored in the device specified by (s) is not ASCII code, it is an operation error, and HEX conversion is disabled. In particular, if M8161 is OFF, you must store the ASCII code also in the higher 8 bits of the device specified by (s).

3. <8-bit conversion mode> When M8161=ON(M8161 is used commonly with RS, ASCI, CCD, CRC instructions)

ASCII characters stored in lower 8 bits in (s) are converted into HEX data, and transferred to (d) in every four digits. The number of characters to be converted is specified by n.
M8161 is used commonly with RS, ASCI, CCD, CRC instructions. When using in 8 bits, you must keep always ON.
M8161 is cleared when changed from RUN to STOP.

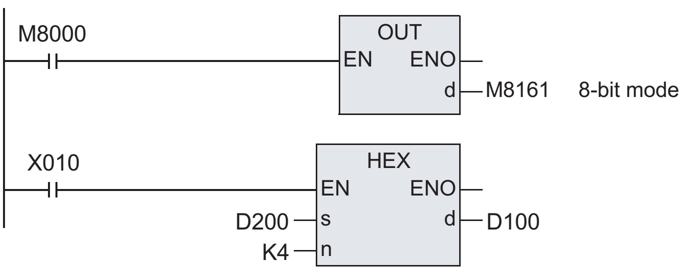




Source data

Operation

In the case of the program shown below, the conversion is executed as follows.



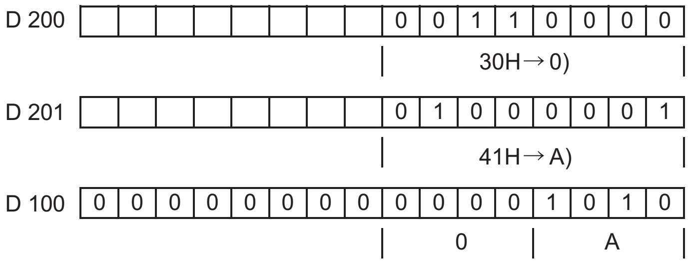
Conversion source data

s	ASCII code	HEX conversion
D 200	30H	0
D 201	41H	A
D 202	42H	B
D 203	43H	C
D 204	31H	1
D 205	32H	2
D 206	33H	3
D 207	34H	4
D 208	35H	5

Number of characters specified and result of conversion
"." is 0.

<table>
<tr><th>d
n</th><th>D 102</th><th>D 101</th><th>D 100</th></tr>
<tr><td>1</td><td rowspan="8" colspan="1">Not changed</td><td rowspan="4">Not changed</td><td>...0H</td></tr>
<tr><td>2</td><td>..0AH</td></tr>
<tr><td>3</td><td>.0ABH</td></tr>
<tr><td>4</td><td>0ABCH</td></tr>
<tr><td>5</td><td>...0H</td><td>ABC1H</td></tr>
<tr><td>6</td><td>..0AH</td><td>BC12H</td></tr>
<tr><td>7</td><td>.0ABH</td><td>C123H</td></tr>
<tr><td>8</td><td>0ABCH</td><td>1234H</td></tr>
<tr><td>9</td><td>...0H</td><td>ABC1H</td><td>2345H</td></tr>
</table>

In the case of n=K2



- When the input data is BCD, after execution of this instruction, you must convert from BCD to BIN.

Cautions

1) FXU PLC supports the instruction at V3.07 or later.
   FX0N PLC supports the instruction at V1.20 or later.
2) FX0N PLC does not support the instruction of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
3) Object devices are limited.
   ▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

7.9.5 CCD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	△	△	×

Outline

This instruction calculates the horizontal parity value or check sum value of error check method used in communication or the like. The error check method also includes cyclic redundancy check (CRC). Use the CRC instruction when determining the CRC value.

→ As for CRC instruction, refer to section 7.18.4.
→ As for complementary number [NEG instruction], refer to section 7.3.10.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	ST
CCD	16 bits	Continuous	CCD block (inputs: EN, s, n; outputs: ENO, d)	CCD(EN, s, n, d);
CCDP	16 bits	Pulse	CCDP block (inputs: EN, s, n; outputs: ENO, d)	CCDP(EN, s, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head object device	ANY16
	n	Number of data (n=1 to 256)	ANY16
Output variable	ENO	Execution state	Bit
	d	Head storage destination device of calculated data	ANY16

3. Applicable devices

Operand type	Bit Devices: System User							Word Devices: Digit designation				System User				Special Unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲1	▲2			●					
d									●	●	●	●	●	●	▲1				●					
n														●	▲1					●	●			

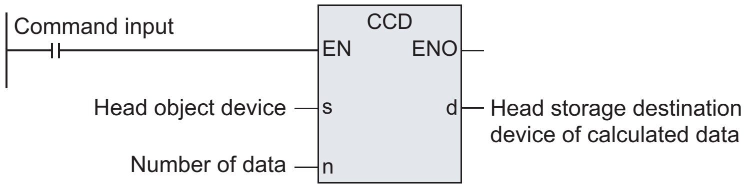
▲: Refer to "Cautions".

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

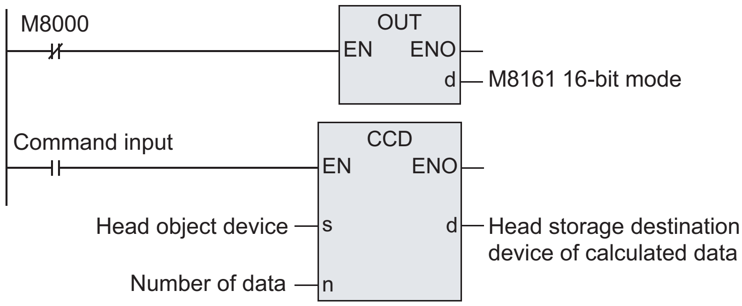
1. 16-bit operation(CCD/CCDP)

The addition data and horizontal parity of the data stored in the device specified by (s) are calculated, and the addition data and the horizontal parity are stored in the device specified by (d).
In this instruction, the mode usable when calculating includes 16-bit mode and 8-bit mode. As for the operation of each mode, refer to the following pages.

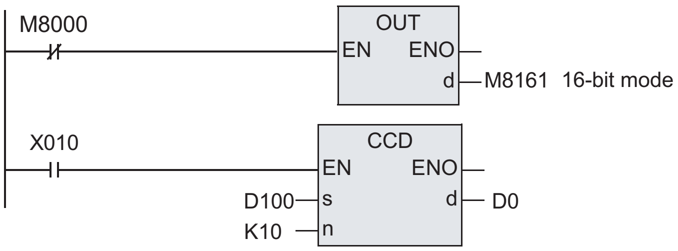


2. <16-bit conversion mode> When M8161=OFF(M8161 is used commonly with RS, ASCI, HEX, CRC instructions)

Of the data of n points starting from the device specified by (s), the addition data and the horizontal parity data of higher and lower 8 bits are stored in the device specified by (d).
M8161 is used commonly with RS, ASCI, HEX, CRC instructions. When using in 16 bits, you must keep always OFF.
M8161 is cleared when changed from RUN to STOP.



Example of 16-bit conversion
In the case of the program shown below, the conversion is executed as follows.

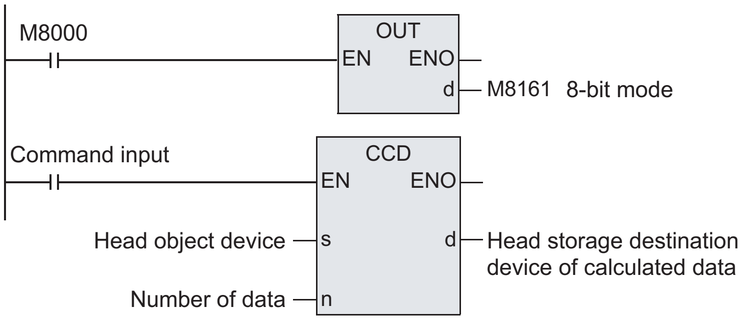


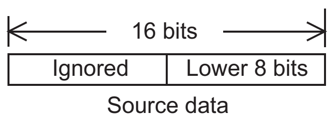
(s)	Example of data content	
D 100 lower	K100 = 01100100	
D 100 higher	K111 = 0110111①	←
D 101 lower	K100 = 01100100	
D 101 higher	K 98 = 01100010	
D 102 lower	K123 = 0111101①	←
D 102 higher	K 66 = 01000010	
D 103 lower	K100 = 01100100	
D 103 higher	K 95 = 0101111①	←
D 104 lower	K210 = 11010010	
D 104 higher	K 88 = 01011000	
Total	K1091	
Horizontal parity	1000010①	←When the number of "1" is an odd number, the horizontal parity is 1. When the number of "1" is an even number, the horizontal parity is 0.

D 0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	1	⟸ 1091 in BCD.
D 1	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	⟸ Horizontal parity

3. <8-bit conversion mode> When M8161=ON(M8161 is used commonly with RS, ASCI, HEX, CRC instructions)

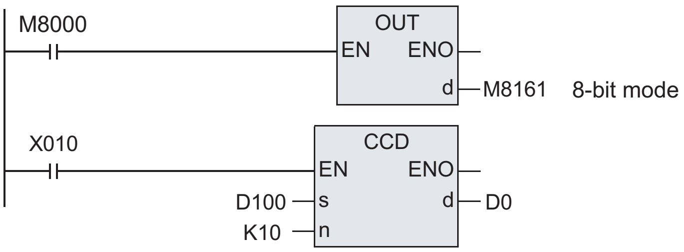
Of the data of n points (lower 8 bits only) starting from the device specified by (s), the addition data and the horizontal parity data are stored in the device specified by (d).
M8161 is used commonly with RS, ASCI, HEX, CRC instructions. When using in 8 bits, you must keep always ON.
M8161 is cleared when changed from RUN to STOP.





Example of 8-bit conversion

In the case of the program shown below, the conversion is executed as follows.



(s)	Example of data content	
D 100	K100 = 01100100	
D 101	K111 = 0110111①	←
D 102	K100 = 01100100	
D 103	K 98 = 01100010	
D 104	K123 = 0111101①	←
D 105	K 66 = 01000010	
D 106	K100 = 01100100	
D 107	K 95 = 0101111①	←
D 108	K210 = 11010010	
D 109	K 88 = 01011000	
Total	K1091	
Horizontal parity	1000010①	←

When the number of "1" is an odd number, the horizontal parity is 1.
When the number of "1" is an even number, the horizontal parity is 0.

D 0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	1	⟸ 1091 in BCD.
D 1	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	⟸ Horizontal parity

Cautions

1) FXU PLC supports the instruction at V3.07 or later.
   FX0N PLC supports the instruction at V1.20 or later.
2) FX0N PLC does not support the instruction of pulse operation type.
   To execute pulse operation, make the instruction execution condition pulse type.
3) Object devices are limited.
   ▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

7.9.6 VRRD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	△	△	△	○	○	×	×

Outline

This instruction reads out the value determined by the variable resistor.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
VRRD	16 bits	Continuous	VRRD (EN → ENO; s → d)	VRRD(EN, s, d);
VRRDP	16 bits	Pulse	VRRDP (EN → ENO; s → d)	VRRDP(EN, s, d);

2. Set data

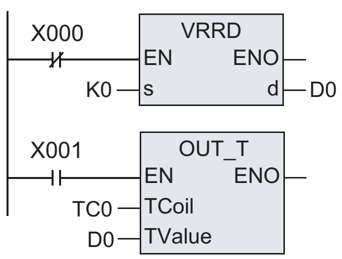
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Variable resistor No. to be read out	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Storage destination of variable resistor value	ANY16

3. Applicable devices

Operand type	Bit Devices: System User							Word Devices: Digit designation				Word Devices: System User				Special Unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														▲1	▲1				●	●	●			
(d)		●	●			●			●	●	●	●	●	●	▲1		●	●	●					

▲: Refer to "Cautions".

Function and operation explanation



The analog value of variable resistor No. is converted into BIN 8 bits, and 0 to 255 are transferred to D0.
As an application example, D0 is used as timer preset value.
The analog timer is obtained by this operation.
As the timer constant value, if a value of more than 256 is necessary, the product of take-in value multiplied by the constant by MUL instruction is set indirectly as the timer constant.

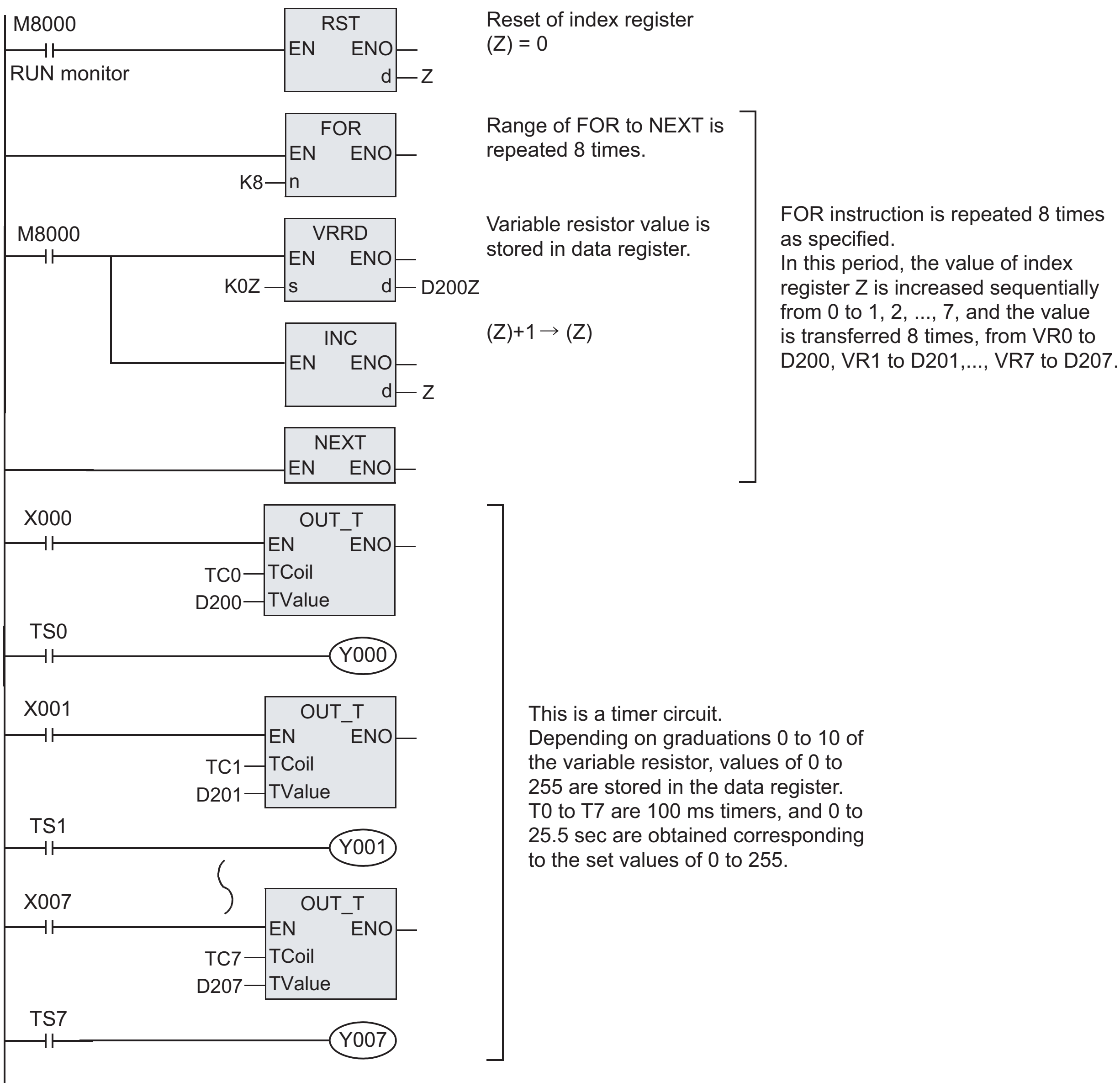
Cautions

1) FX1NC, FX2NC PLCs are not provided with variable resistors for reading out by this instruction, and hence do not function even if programmed.
2) FX3G PLC supports the instruction at V1.10 or later.
3) Object devices are limited
▲1: FX3G PLCs only are applicable.

Program example

Variable resistor values are read out sequentially.
Depending on variable resistors VR0 to VR7, the specified values of VRRD instruction are K0 to K7.
In the example below, being decorated by index (Z = 0 to 7), K0Z is K0 to K7.

[Structured ladder]



[ST]

```
RST(M8000, Z);
FOR(TRUE, K8);
VRRD(M8000, K0Z, D200Z);
INC(M8000, Z);
NEXT(TRUE);
OUT_T(X000, TC0, D200);
OUT(TS0, Y000);
OUT_T(X001, TC1, D201);
OUT(TS1, Y001);
        ⟆
OUT_T(X007, TC7, D207);
OUT(TS7, Y007);
```

7.9.7 VRSC

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	△	△	△	○	○	×	×

Outline

This instruction reads out the value determined in the variable resistor graduations.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
VRRD	16 bits	Continuous	VRSC EN ENO s d	VRSC(EN, s, d);
VRRDP	16 bits	Pulse	VRSCP EN ENO s d	VRSCP(EN, s, d);

2. Set data

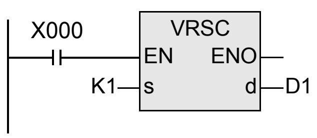
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Variable resistor number for reading out the graduations	ANY16
Output variable	ENO	Execution state	Bit
	d	Storage destination of variable resistor graduations	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System User							Digit designation				System User				Special Unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														▲1	▲1				●	●	●			
d		●	●			●			●	●	●	●	●	●	▲1		●	●	●					

▲: Refer to "Cautions".

Function and operation explanation



This instruction reads the value of a variable analog potentiometer on the variable analog potentiometer board attached to the main unit as a numeric value in the range from 0 to 10.
However, the actual scale value does not always correspond to the switching position of the variable analog potentiometer scale (0 to 10).
This instruction converts into a binary value the scale value of a variable analog potentiometer specified in s, and transfers the converted binary value to d.

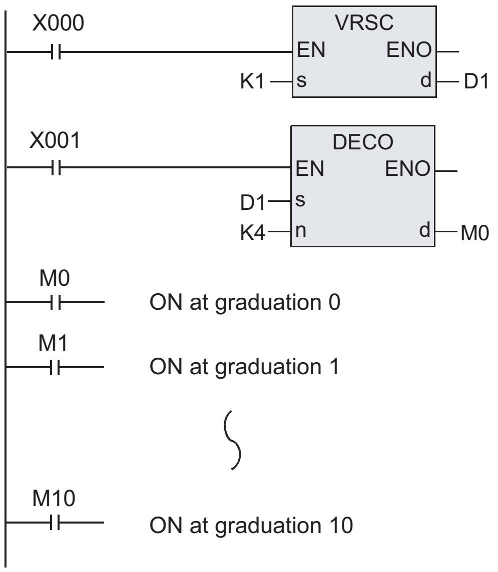
Cautions

1) FX1NC, FX2NC PLCs are not provided with variable resistors for reading out by this instruction, and hence do not function even if programmed.
2) FX3G PLC supports the instruction at V1.10 or later.
3) Object devices are limited
▲1: FX3G PLCs only are applicable.

Program example

This is an example of use as rotary switch.
Depending on variable resistor graduations 0 to 10, any one of auxiliary relays M0 to M10 is turned ON.
By DECO instruction, the auxiliary relays are occupied from M0 to M15.

[Structured ladder]



[ST]

```
VRSC(X000, K1, D1);
DECO(X001, D1, K4, M0);
OUT(M0, ···);
OUT(M1, ···);
      ⸾
OUT(M10, ···);
```

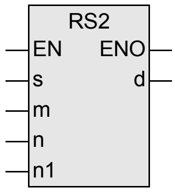
7.9.8 RS2

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	×	×	×	×	×	×

Outline

This instruction transmits and receives data by no-procedure communication via serial port of RS-232C or RS-485 installed in the basic unit.
In the case of FX3G PLC, data can be transmitted and received by no-procedure communication also via standard built-in port (RS-422).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
RS2	16 bits	Continuous		RS2(EN, s, m, n, n1, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device in which transmission data is stored • 16-bit processing mode: m÷2 points *1 occupied	ANY16
	(m)	Number of transmission data bytes [Setting range: 0 to 4096]	ANY16
	(n)	Number of reception data bytes [Setting range: 0 to 4096]	ANY16
	(n1)	Used channel number [Setting content: K0: ch0, K1: ch1, K2: ch2]*2	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Head device for storing reception data upon completion of reception • 16-bit processing mode: n÷2 points *1 occupied	ANY16

*1. Rounding up below the decimal point

*2. In the case of 14-point or 24-point type of FX3G PLC, ch2 cannot be used. Ch0 is applicable to FX3G PLC only.

3. Applicable devices

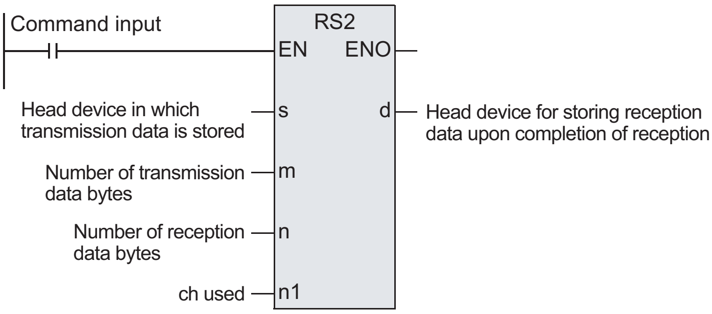
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit designation				System User				Special Unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	●				●					
(m)														●	●					●	●			
(d)														●	●				●					
(n)														●	●					●	●			
(n2)																				●	●			

Function and operation explanation

1. 16-bit operation(RS2)

This instruction transmits and receives data by no-procedure communication via serial port of RS-232C or RS-485 installed in the basic unit.

→ **As for the detailed explanation, refer to the communication control manual.**



Related devices

→ **As for the detailed explanation, refer to the communication control manual.**

Device			Name
ch0[*1]	ch1	ch2[*1]	
M8371	M8401	M8421	Transmission waiting flag[*2]
M8372	M8402	M8422	Transmission request[*2]
M8373	M8403	M8423	Reception complete flag[*2]
-	M8404	M8424	Carrier detection flag
-	M8405	M8425	Data set ready (DSR) flag[*3]
-	-	-	-
M8379	M8409	M8429	Time out judging flag
-	-	-	-
M8062	M8063	M8438	Serial communication error

Device			Name
ch0[*1]	ch1	ch2[*1]	
D8370	D8400	D8420	Communication format setting
-	-	-	-
D8372	D8402	D8422	Transmission data remainder points[*2]
D8373	D8403	D8423	Reception points monitor[*2]
D8375	D8405	D8425	Communication parameter display
D8379	D8409	D8429	Time out time setting
D8380	D8410	D8430	Headers 1, 2
D8381	D8411	D8431	Headers 3, 4
D8382	D8412	D8432	Terminators 1, 2
D8383	D8413	D8433	Terminators 3, 4
D8384	D8414	D8434	Reception sum (reception data)
D8385	D8415	D8435	Reception sum (calculation result)
D8386	D8416	D8436	Transmission sum
D8389	D8419	D8439	Operation mode display
D8062	D8063	D8438	Error code number of serial communication error

*1. ch0 is applicable to FX3G PLC only.
In the case of 14-point or 24-point type of FX3G PLC, ch2 cannot be used.

*2. Cleared when changed from RUN to STOP.

*3. FX3U, FX3UC PLCs are applicable at Ver.2.30 or later.

System configuration

In order to use this instruction, you must install any one of the following products in the basic unit.

→ **As for the system configuration, refer to the hardware manual of the corresponding PLC main unit.**

→ **As for the detailed explanation, refer to the communication control manual.**

PLC	Type of communication	Option
FX3U, FX3UC-32MT-LT(-2)	RS-232C communication	FX3U-232-BD or FX3U-232ADP(-MB)
	RS-485 communication	FX3U-485-BD or FX3U-485ADP(-MB)
FX3UC(D, DSS)	RS-232C communication	FX3U-232ADP(-MB)
	RS-485 communication	FX3U-485ADP(-MB)
FX3G	RS-232C communication	FX3G-232-BD or FX3U-232ADP(-MB) (FX3G-CNV-ADP is needed) RS-232C/RS-422 converter*1 (FX-232AW, FX-232AWC, FX-232AWC-H)
	RS-485 communication	FX3G-485-BD or FX3U-485ADP(-MB) (FX3G-CNV-ADP is needed)

*1. Required to use ch0 (standard built-in RS-422 port) in FX3G PLCs.

Difference between RS instruction and RS2 instruction

Item	RS2 instruction	RS instruction	Remarks
Header size	1 to 4 characters (bytes)	Up to 1 character (byte)	For the RS2 instruction, up to 4 characters (bytes) can be specified as a header or terminator.
Terminator size	1 to 4 characters (bytes)	Up to 1 character (byte)	
Attachment of check sum	The check sum can be automatically attached.	The check sum should be attached by a user program.	For the RS2 instruction, the check sum can be automatically attached to the sent and received data. In this case, however, make sure to use a terminator with the communication frame to be sent and received.
Used channel number	ch1	ch0, ch1, ch2	In RS2 instruction is as follows: Ch2 is not available in 14-point and 24-point type FX3G PLCs. Ch0 is available only in FX3G PLCs.

Function change by version

Corresponding version			Item	Outline of function
FX3G	FX3U	FX3UC		
Ver. 1.00 or later	Ver. 2.30 or later	Ver. 2.30 or later	ch1 Data set ready (DSR) flag	When DR (DSR) signal of ch1 is ON, special device M8405 is turned ON.
			ch2 Data set ready (DSR) flag	When DR (DSR) signal of ch2 is ON, special device M8425 is turned ON.

Cautions

→ **As for other cautions, refer to the communication control manual.**

1) With RS, RS2 instructions, do not drive the same port simultaneously by plural instructions.
2) You cannot use RS, RS2 instructions, and IVCK, IVDR, IVRD, IVWR, IVBWR instructions on the same port.
3) When using the header or terminator, please set the data of the header or terminator in the special D before driving the RS2 instruction. Do not change the values of the header or terminator during driving of RS2 instruction.

7.9.9 PID

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	△	×	×

Outline

This instruction executes PID control for changing the output values depending on the change value of the input.

→ **As for the detail, refer to the analog control manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
PID	16 bits	Continuous	PID EN ENO s1 d s2 s3	PID(EN, s1, s2, s3, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s1)	Data register for storing the target value (SV)	ANY16
	(s2)	Data register for storing the measured value (PV).	ANY16
	(s3)	Data register for storing the parameter [29 points occupied][*1]	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Data register for storing the output value (MV)	ANY16

*1. Variable depending on the setting content of the parameter.

3. Applicable devices

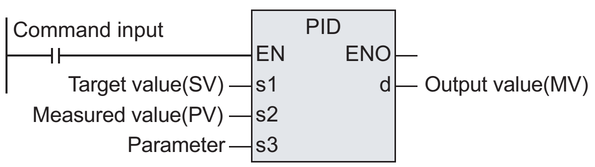
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit designation				System User				Special Unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)														●	▲1	▲2								
(s2)														●	▲1	▲2								
(s3)														●	▲1									
(s3)														●	▲1	▲2								

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation(PID)

When the program is executed by setting the target value of the device specified by (s1), the measured value of the device specified by (s2), and the parameter of the device specified by (s3), at every sampling time of the device specified by (s3), the calculation result (MV) is stored in the output value of the device specified by (d).



2. Setting items

Setting items		Content	No. of points occupied
s1	Target value(SV)	• To set the target value (SV). • PID instruction does not change the contents of setting. • Cautions when using auto-tuning (limit cycle method) If the target value for auto-tuning and the target value for PID control are different, the value to which the bias value is added is set, and the actual target value must be stored when the auto-tuning flag is turned OFF.	1 point
s2	Measured value(PV)	Input value of PID operation.	1 point
s3	Parameter[*1]	1) Auto-tuning: In the case of limit cycle method The devices are occupied by 29 points from the head device specified by s3. 2) Auto-tuning: In the case of step response method a) Action setting (ACT) setting: When all of bit1, bit2, bit5 are other than "0" The devices are occupied by 25 points from the head device specified by s3. b) Action setting (ACT) setting: When all of bit1, bit2, bit5 are "0" The devices are occupied by 20 points from the head device specified by s3.	29 points 25 points 20 points
s3	Output value(MV)	1) In the case of PID control (in ordinary processing) The initial output value is set at the user side before instruction drive. Thereafter, operation results are stored. 2) Auto-tuning: In the case of limit cycle method ULV value or LLV value is automatically issued during auto-tuning, and specified MV value is set after auto-tuning. 3) Auto-tuning: In the case of step response method Please set the step output value at the user side before instruction drive. During auto-tuning, the MV output cannot be changed at the PID instruction side.	1 point

*1. When the auto-tuning is not used, the same number of points as in the step response method are occupied.

3. List of parameters s3 to s3 +28

Setting items			Content of setting	Remarks
s3	Sampling time(TS)		1 to 32767 [ms]	Value shorter than operation cycle cannot be executed.
s3 +1	Action setting (ACT)	bit0	0: Normal action 1:Reverse action	Direction of action
		bit1	0: Input change amount alarm absent 1: Input change amount alarm valid	
		bit2	0: Output change amount alarm absent 1: Output change amount alarm valid	Do not turn ON bit2 and bit5 simultaneously.
		bit3	Not usable.	
		bit4[*2]	0: Auto-tuning inaction 1: Auto-tuning execute	
		bit5[*2]	0: Output value upper and lower limit setting absent 1: Output value upper and lower limit setting valid	Do not turn ON bit2 and bit5 simultaneously.
		bit6[*2,*3]	0: Step response method 1: Limit cycle method	Selection of auto-tuning mode
		bit7 to bit15	Not usable.	
s3 +2	Input filter constant (α)		0 to 99 [%]	No input filter in the case of 0
s3 +3	Proportional gain (KP)		1 to 32767 [%]	
s3 +4	Integral time(TI)		0 to 32767 [×100ms]	Handled as ∞ in the case of 0 (No integral)
s3 +5	Differential gain (KD)		0 to 100 [%]	No differential gain in the case of 0
s3 +6	Differential time(TD)		0 to 32767 [×10ms]	No differentiation in the case of 0
s3 +7 : s3 +19	Occupied by the internal processing of PID operation. Do not change the data.			
s3 +20[*1]	Input change amount (increase side) alarm setting value		0 to 32767	Direction of action(ACT): valid if s3 + 1 bit1 is 1
s3 +21[*1]	Input change amount (decrease side) alarm setting value		0 to 32767	Direction of action(ACT): valid if s3 + 1 bit1 is 1

*1. s3 + 20 to + 24 will be occupied in the case of bit1 = 1, bit2 = 1, or bit5 = 1 of s3 + 1 action setting (ACT)

*2. FXU, FX2C PLCs are not usable.

*3. FX1S, FX1N, FX2N, FX1NC, FX2NC PLCs are not usable.

Setting items			Content of setting	Remarks
s3 +22*1	Output change amount (increase side) alarm setting value		0 to 32767	Direction of action (ACT): s3 +1 bit2 = 1, bit5 = 0; valid
	Output upper limit setting value		-32768 to 32767	Direction of action (ACT): s3 +1 bit2 = 0, bit5 = 1; valid
s3 +23*1	Output change amount (decrease side) alarm setting value		0 to 32767	Direction of action (ACT): s3 +1 bit2 = 1, bit5 = 0; valid
	Output lower limit setting value		-32768 to 32767	Direction of action (ACT): s3 +1 bit2 = 0, bit5 = 1; valid
s3 +24*1	Alarm output	bit0	0: Input change amount (increase side) not over 1: Input change amount (increase side) over	Direction of action (ACT): s3 +1 bit1 = 1 or bit2 = 1; valid
		bit1	0: Input change amount (decrease side) not over 1: Input change amount (decrease side) over	
		bit2	0: Output change amount (increase side) not over 1: Output change amount (increase side) over	
		bit3	0: Output change amount (decrease side) not over 1: Output change amount (decrease side) over	
The following setting is required when using the limit cycle method (in the case of action direction (ACT) b6: ON).				
s3 +25*2	PV value threshold (hysteresis) width (SHPV)		To be set according to fluctuation of measured value (PV).	Action setting (ACT) b6: Occupied when limit cycle method (ON) is selected.
s3 +26*2	Output value upper limit (ULV)		Setting of maximum output value (ULV) of output value (MV)	
s3 +27*2	Output value lower limit (LLV)		Setting of minimum output value (LLV) of output value (MV)	
s3 +28*2	Weight setting parameter from end of tuning cycle to start of PID control (KW)		-50 to 32717%	

*1. s3 + 20 to + 24 will be occupied in the case of bit1 = 1, bit2 = 1, or bit5 = 1 of s3 + 1 action setting (ACT)

*2. FX2N, FX2NC, FX1N, FX1NC, FX1S, FXU, FX2C PLCs are not usable.

Cautions

1. Cautions when using a plurality of instructions

Possible to execute plural times simultaneously (the number of loops is not limited), but you must be careful so that the device numbers may not be overlapped in the devices used in s3 or d used in operation.

2. Number of parameters s3 occupied

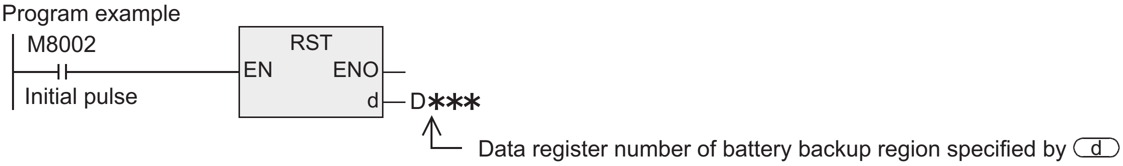
1) In the case of limit cycle method
   - Devices are occupied by 29 points from the head device specified by s3.

2) In the case of step response method
   - Action setting (ACT) setting: when all of bit1, bit2, bit5 are other than "0"
     Devices are occupied by 25 points from the head device specified by s3.
   - Action setting (ACT) setting: when all of bit1, bit2, bit5 are "0"
     Devices are occupied by 20 points from the head device specified by s3.

3. When specifying a device in power failure hold region

As for the output value (MV) of PID instruction, specify the data register d excluding the power failure hold region.
(When specifying the data register in the power failure hold region, you must clear the content of backup while the PLC is ON by the following program.)

Program example



4. FXU, FX2C PLCs support the instruction at V3.30 or later.

5. FX2N PLC supports the upper and lower limit setting functions of the auto-tuning and output value at V3.00 or later.

6. Object devices are limited.

▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

Error

If an operation error occurs, special auxiliary relay M8067 is turned ON, and the error code is stored in special data register D8067.

7.10 External Device

7.10.1 MNET

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	△	×	×

Outline

This instruction exchanges ON/OFF signals between the FXU, FX2C PLCs, and F-16NP/NT type interface unit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
MNET	16 bits	Continuous	MNET EN ENO s d	MNET(EN, s, d);
MNETP	16 bits	Pulse	MNETP EN ENO s d	MNETP(EN, s, d);

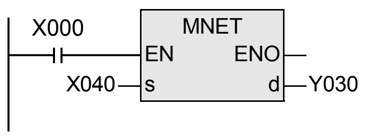
2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head input number of FX2-24EI connected to F-16NP/NT (16 points occupied)	Bit
Output variable	ENO	Execution state	Bit
	d	Head output number of FX2-24EI connected to F-16NP/NT (8 points occupied)	Bit

3. Applicable devices

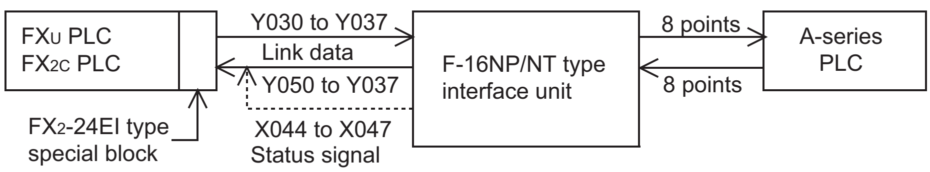
Operand type	Bit Devices							Word Devices												Others				
	System User							Digit designation				System User				Special Unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●																							
d		●																						

Function and operation explanation



- To exchange signals with F-16NP/NT.
- Head input and output numbers are determined by the connection position of FX2-24EI type special block.

The following signals are exchanged by the above instruction.



Cautions

1) In the case of FX-16NP/NT, FX-16NP/NT-S3 type interface block, this instruction is not used, and FX2-24EI is not needed.
2) FXU, FX2C PLCs do not support the instruction at V3.30 or later.
This instruction is disused from V3.30 and later.

7.10.2 ANRD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	△	×	×

Outline

This instruction writes the analog input of F2-6A type analog input and output unit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ANRD	16 bits	Continuous	ANRD EN ENO s d1 n d2	ANRD(EN, s, n, d1, d2);
ANRDP	16 bits	Pulse	ANRDP EN ENO s d1 n d2	ANRDP(EN, s, n, d1, d2);

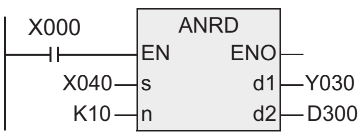
2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head input number of FX2-24EI connected to F2-6A (16 points occupied)	Bit
	n	Channel number of analog input (n=10,11,12,13)	ANY16
Output variable	ENO	Execution state	Bit
	d1	Head output number of FX2-24EI connected to F2-6A (8 points occupied)	Bit
	d2	Device storing analog input value (8-bit binary)	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●																							
n																				●	●			
d1		●																						
d2									●	●	●	●	●	●			●	●	●					

Function and operation explanation



To read out from analog input CH10 to D300.

- To read in analog input of F2-6A type analog input and output unit.
- The content of the device specified by s, d1 is determined by the connection position of FX2-24EI type special adapter.
- In the device specified by d2, 8-bit binary analog data is stored.

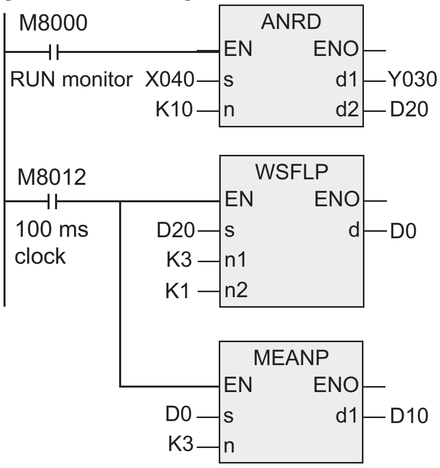
Caution

FXU, FX2C PLCs do not support the instruction at V3.30 or later.
This instruction is disused from V3.30 and later.

Program example

This is intended to determine the average of three points of time series data in 100 ms unit in order to suppress fluctuations of the analog input.

[Structured ladder]



To store the data of input channel 10 of F_2-6A type analog input and output unit connected from X040, Y030, in the D20.

The content of D20 is shifted to D0, D1, D2 in 100 ms unit.

The average of D0, D1, D2 is stored in the D10.

[ST]

```
ANRD(M8000, X040, K10, Y030, D20);
WSFLP(M8012, D20, K3, K1, D0);
MEANP(M8012, D0, K3, D10);
```

7.10.3 ANWR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	△	×	×

Outline

This instruction writes data from the PLC in the F2-6A type analog input and output unit, and issues as analog data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
ANWR	16 bits	Continuous	ANWR EN ENO s1 d s2 n	ANWR(EN, s1, s2, n, d);
ANWRP	16 bits	Pulse	ANWRP EN ENO s1 d s2 n	ANWRP(EN, s1, s2, n, d);

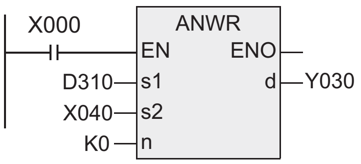
2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Device for storing analog output data (8-bit binary)	ANY16
	s2	Head input number of FX2-24EI connected to F2-6A	Bit
	n	Channel number of analog output	ANY16
Output variable	ENO	Execution state	Bit
	d	Head output number of FX2-24EI connected to F2-6A	Bit

3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●			●	●	●	●	●			
s2	●																		●					
n																				●	●			
d	●																							

Function and operation explanation



To write in from D310 into analog input CN0

- Data is written in from PLC into F2-6A type analog input and output unit, and issued as analog data.
- The content of the device specified by s2, d is determined by the connection position of FX2-24EI type special adapter.
- In the device specified by s1, 8-bit binary data is stored.

Caution

FXU, FX2C PLCs do not support the instruction at V3.30 or later.
This instruction is disused from V3.30 and later.

7.10.4 RMST

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	○	×	×

Outline

This instruction gives start signal from the PLC or receives status information, in the F2-32RM type programmable cam switch.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
RMST	16 bits	Continuous	RMST EN ENO s d1 n d2	RMST(EN, s, n, d1, d2);

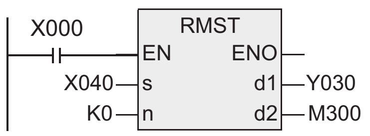
2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head input number of FX2-24EI connected to F2-32RM (16 points occupied)	Bit
	(n)	Program (bank) number of F2-32RM (n = 0, 1).	ANY16
Output variable	ENO	Execution state	Bit
	(d1)	Head output number of FX2-24EI connected to F2-32RM (8 points occupied)	Bit
	(d2)	Head of device storing status information (8 points occupied)	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)	●																							
(n)																				●	●			
(d1)		●																						
(d2)		●	●			●													●					

Function and operation explanation



FXU→FX2-32RM Start command
F2-32RM→FXU Status information

- This instruction gives start command from the PLC or receives status information, in the F2-32RM type programmable cam switch.
- The content of the device specified by (s), (d1) is determined by the connection position of FX2-24EI type special adapter.
- In the device specified by (d2), the status information is stored as follows.

		M307	M306	M305	M304	M303	M302	M301	M300
ON	→	Normal	Normal	CW	Normally ON	1.0°	START	―	BANK1
OFF	→	S/W error	H/W error	CCW	―	0.5°	STOP	Normally OFF	BANK0

As for the meaning of each status, please refer to the user's manual of F2-32RM type programmable cam switch.

7.10.5 RMWR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	○	×	×

Outline

This instruction sends output prohibit information from the PLC to the F2-32RM type programmable cam switch.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RMWR	16 bits	Continuous	RMWR EN ENO s1 d s2	RMWR(EN, s1, s2, d);
RMWRP	16 bits	Pulse	RMWRP EN ENO s1 d s2	RMWRP(EN, s1, s2, d);
DRMWR	32 bits	Continuous	DRMWR EN ENO s1 d s2	DRMWR(EN, s1, s2, d);
DRMWRP	32 bits	Pulse	DRMWRP EN ENO s1 d s2	DRMWRP(EN, s1, s2. d);

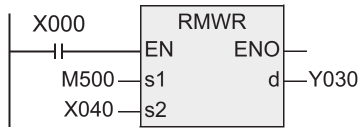
2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Head bit device of output prohibit table • 16-bit operation: 16 points occupied • 32-bit operation: 32 points occupied	Bit
	s2	Head input number of FX2-24EI connected to F2-32RM (16 points occupied)	Bit
Output variable	ENO	Execution state	Bit
	d	Head output number of FX2-24EI connected to F2-32RM (8 points occupied)	Bit

3. Applicable devices

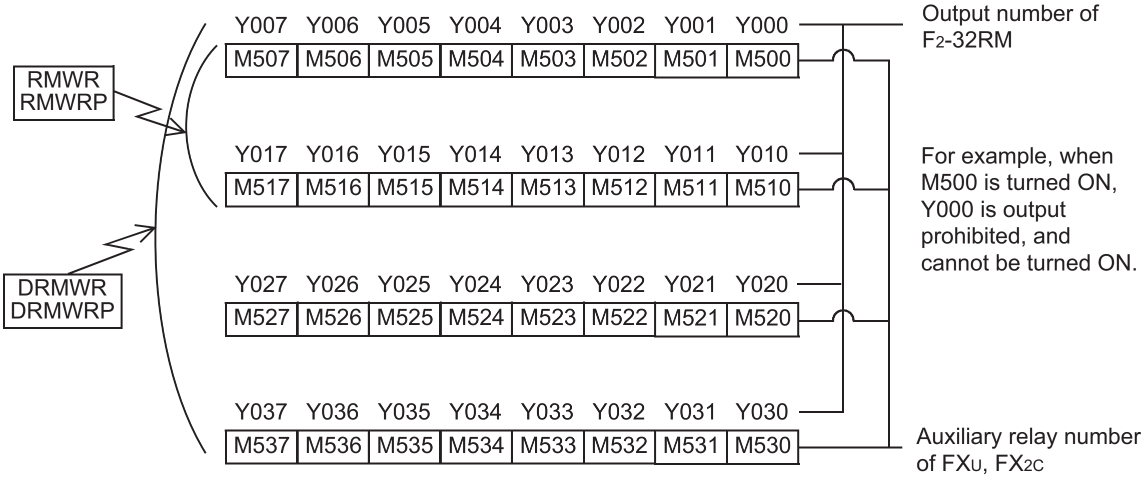
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1	●	●	●			●													●					
s2	●																							
d		●																						

Function and operation explanation



Writing of output prohibit information

- This instruction sends output prohibit information from the PLC to the F2-32RM type programmable cam switch.
- The content of the device specified by (s2), (d) is determined by the connection position of FX2-24EI type special adapter.
- The device specified by (s1) is an output prohibit table, and is handled as octagonal number as shown in the example below.



7.10.6 RMRD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	○	×	×

Outline

This instruction reads out the ON/OFF state of output of the F2-32RM type programmable cam switch to the PLC.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RMRD	16 bits	Continuous	RMRD: EN, s → ENO, d1, d2	RMRD(EN, s, d1, d2);
RMRDP	16 bits	Pulse	RMRDP: EN, s → ENO, d1, d2	RMRDP(EN, s, d1, d2);
DRMRD	32 bits	Continuous	DRMRD: EN, s → ENO, d1, d2	DRMRD(EN, s, d1, d2);
DRMRDP	32 bits	Pulse	DRMRDP: EN, s → ENO, d1, d2	DRMRDP(EN, s, d1, d2);

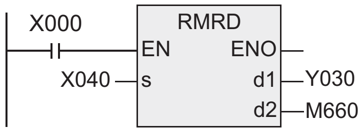
2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head input number of FX2-24EI connected to F2-32RM (16 points occupied)	Bit
Output variable	ENO	Execution state	Bit
	d1	Head output number of FX2-24EI connected to F2-32RM (8 points occupied)	Bit
	d2	Bit device for storing output status (ON/OFF). • 16-bit operation: 16 points occupied • 32-bit operation: 32 points occupied	Bit

3. Applicable devices

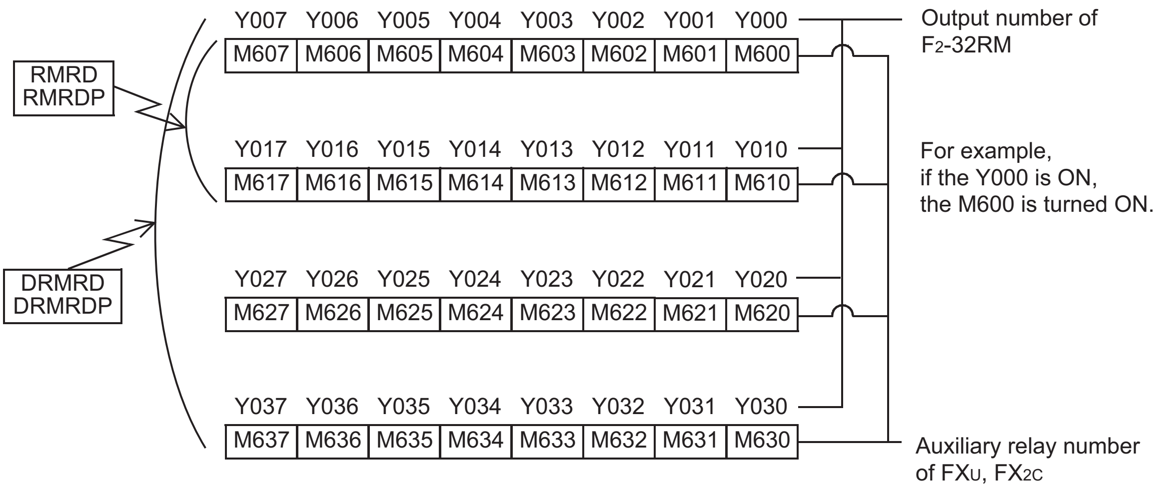
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●																							
d1		●																						
d2		●	●			●													●					

Function and operation explanation



Reading of ON/OFF information

- This instruction reads out the ON/OFF state of output of the F2-32RM type programmable cam switch to the PLC.
- The content of the device specified by (s), (d1) is determined by the connection position of FX2-24EI type special adapter.
- If X000 is turned OFF, the content of the device specified by (d2) is not changed.
- The ON/OFF information being read out is stored in the device specified by (d2), and is handled as octagonal number as shown in the example below.



7.10.7 RMMN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	○	×	×

Outline

This instruction reads out the rotating speed (rpm) or present angle of the resolver connected to the F2-32RM type programmable cam switch to the PLC.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RMMN	16 bits	Continuous	RMMN EN ENO s d1 d2	RMMN(EN, s, d1, d2);
RMMNP	16 bits	Pulse	RMMNP EN ENO s d1 d2	RMMNP(EN, s, d1, d2);

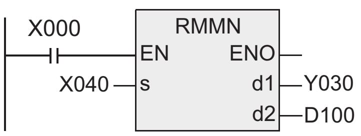
2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head input number of FX2-24EI connected to F2-32RM.	Bit
Output variable	ENO	Execution state	Bit
	(d1)	Head output number of FX2-24EI connected to F2-32RM.	Bit
	(d2)	Device for storing data of rotating speed and present angle (2 points occupied)	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)	●																							
(d1)		●																						
(d2)									●	●	●	●	●	●			●	●	●					

Function and operation explanation



Monitor of rotating speed or present angle

- The rotating speed (rpm) or present angle of the resolver connected to the F2-32RM type programmable cam switch is read out to the PLC. Whether the rotating speed or the present angle is determined by the setting switch #4 of F2-32RM, whether OFF or ON.
- The content of the device specified by (s), (d1) is determined by the connection position of FX2-24EI type special adapter.
- In the device specified by (d2), the data of the rotating speed or the present angle being read out is stored.

D100	8	3	0	← In the case of rotating speed (rpm) (actually binary value)
D100	3	5	0	← In the case of present angle (deg) (binary value by rounding down 0.5°)

7.10.8 BLK

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	△	×	×

Outline

This instruction specifies the block number for the F2-30GM type pulse output unit from the PLC.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BLK	16 bits	Continuous	BLK: EN, s1, s2 → ENO, d	BLK(EN, s1, s2, d);
BLKP	16 bits	Pulse	BLKP: EN, s1, s2 → ENO, d	BLKP(EN, s1, s2, d);

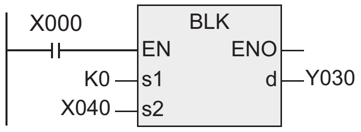
2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Block number (K0 to K31)	ANY16
	s2	Head input number of FX2-24EI connected to F2-30GM	Bit
Output variable	ENO	Execution state	Bit
	d	Head output number of FX2-24EI connected to F2-30GM	Bit

3. Applicable devices

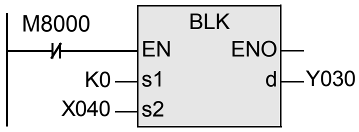
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●			●	●	●	●	●			
s2	●																		●					
d		●																						

Function and operation explanation



FXU, FX2C→FX2-30GM
Block numbers 0 to 31 (decimal)

- The block number is specified from the PLC to the F2-30GM type pulse output unit.
  The block number is the content of the device specified by (s1), and the value is BIN, but is valid in a range of 0 to 31 as converted to BCD.
- When using the BCD digital switch as (s1), it is converted to BIN value, and the result must be specified.
  (Constant K is automatically converted into BIN value, and 0 to 31 can be directly entered.)
- The content of the device specified by (s1) and (d) is determined by the connection position of the FX2-24EI type special block.
- You must always use this instruction when using the F2-30GM.
  If block number specification is not needed from FXU, FX2C PLCs to F2-30GM, please program as follows.



- In the FX-1GM type pulse output unit, TO instruction is used instead of BLK instruction, and the FX2-24EI type interface block is not needed.

Caution

FXU, FX2C PLCs do not support the instruction at V3.30 or later.
This instruction is disused from V3.30 and later.

7.10.9 MCDE

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	×	×	×	△	×	×

Outline

This instruction sends the M code numbers M0 to M77 to the PLC from the F2-30GM type pulse output unit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
MCDE	16 bits	Continuous	MCDE: EN, s → ENO, d1, d2	MCDE(EN, s, d1, d2);
MCDEP	16 bits	Pulse	MCDEP: EN, s → ENO, d1, d2	MCDEP(EN, s, d1, d2);2

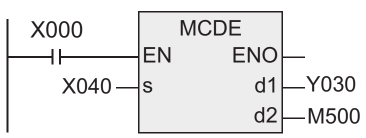
2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head input number of FX2-24EI connected to F2-30GM	Bit
Output variable	ENO	Execution state	Bit
	d1	Head output number of FX2-24EI connected to F2-30GM	Bit
	d2	Bit device for issuing M code number (78 points occupied)	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s	●																							
d1		●																						
d2		●	●			●																		

Function and operation explanation



FX2-30GM→FXU,FX2C
M code number M0 to 77 (octal)

- From F2-30GM type pulse output unit, M code number M0 to M77 is sent out to the PLC.
- The content of the device specified by s, d1 is determined by the connection position of FX2-24EI type special block.
- When M code output instruction is executed at the F2-30GM side, input X is operated according to the value 0 to 77 (octal), and the result is stored in M500 to M577 in octal notation.
  For example, in the case of M code 23, M523 is turned ON.
- For the ease of understanding of correspondence of M code number between the PLC side and the F2-30GM side, it is recommended to set 00 in the lower two digits of device M, S specified by d2.
- In the FX-1GM type pulse output unit, FROM instruction is used instead of MCDE instruction.

Cautions

FXU, FX2C PLCs do not support the instruction at V3.30 or later.
This instruction is disused from V3.30 and later.

7.11 Data Transfer 2

7.11.1 ZPUSH

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction temporarily retracts the present values of index registers V0 to V7, Z0 to Z7.
To return the retracted present values to the original values, use the ZPOP instruction.

→ **As for ZPOP instruction, refer to section 7.11.2.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ZPUSH	16 bits	Continuous	ZPUSH EN ENO d	ZPUSH(EN, d);
ZPUSHP	16 bits	Pulse	ZPUSHP EN ENO d	ZPUSHP(EN, d);

2. Set data

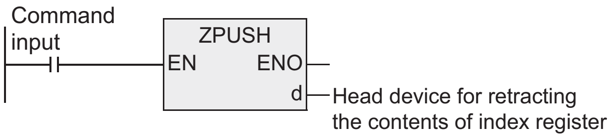
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	d	Head device for retracting contents of index registers V0 to V7, Z0 to Z7 temporarily [(1+16 × times of retraction) occupied]	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
d														●	●									

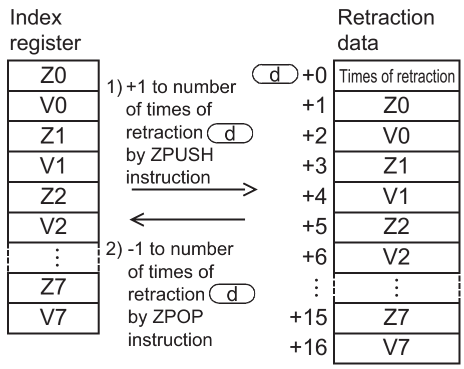
Function and operation explanation

1. 16-bit operation (ZPUSH/ZPUSHP)

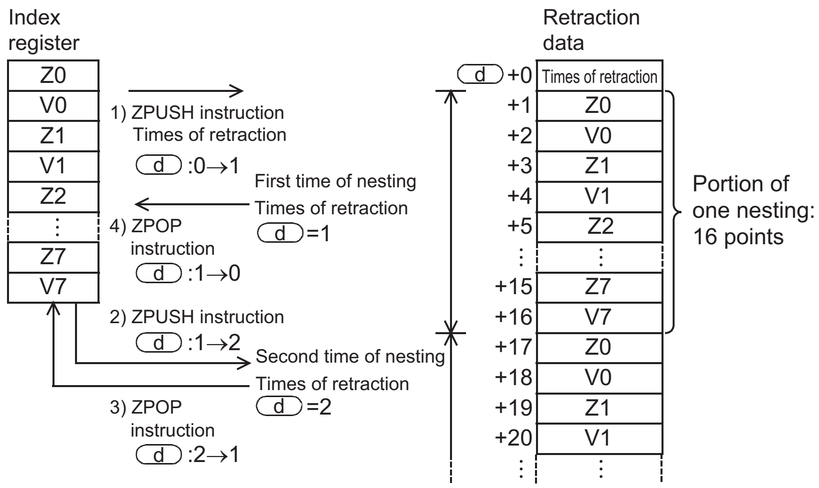


1) Contents of index registers V0 to V7, Z0 to Z7 are temporarily retracted in and after the device specified by d. When the contents of index registers are retracted, the number of retracting of the device specified by d is incremented by +1.
2) To restore the data, use ZPOP instruction.
   Use ZPUSH, ZPOP instructions in pair.
3) By specifying d by the same device, ZPUSH and ZPOP instructions can be used as nesting. In this case, every time the ZPUSH instruction is executed, the region to be used after the device specified by d is added by 16 points each. Hence, you must preliminarily reserve the region for the number of times to be used by nesting.
4) The composition of the data to be retracted after the device specified by d is as shown below.

- **No action of nesting**



- **With action of nesting**



Related instructions

Instruction	Content
ZPOP	This instruction restores the index registers V0 to V7, Z0 to Z7 once retracted by ZPUSH instruction.

Cautions

- Without action of nesting, please clear the number of retraction of the device specified by d before execution of ZPUSH instruction.
- With action of nesting, please clear the number of retraction of the device specified by d before first execution.

Error

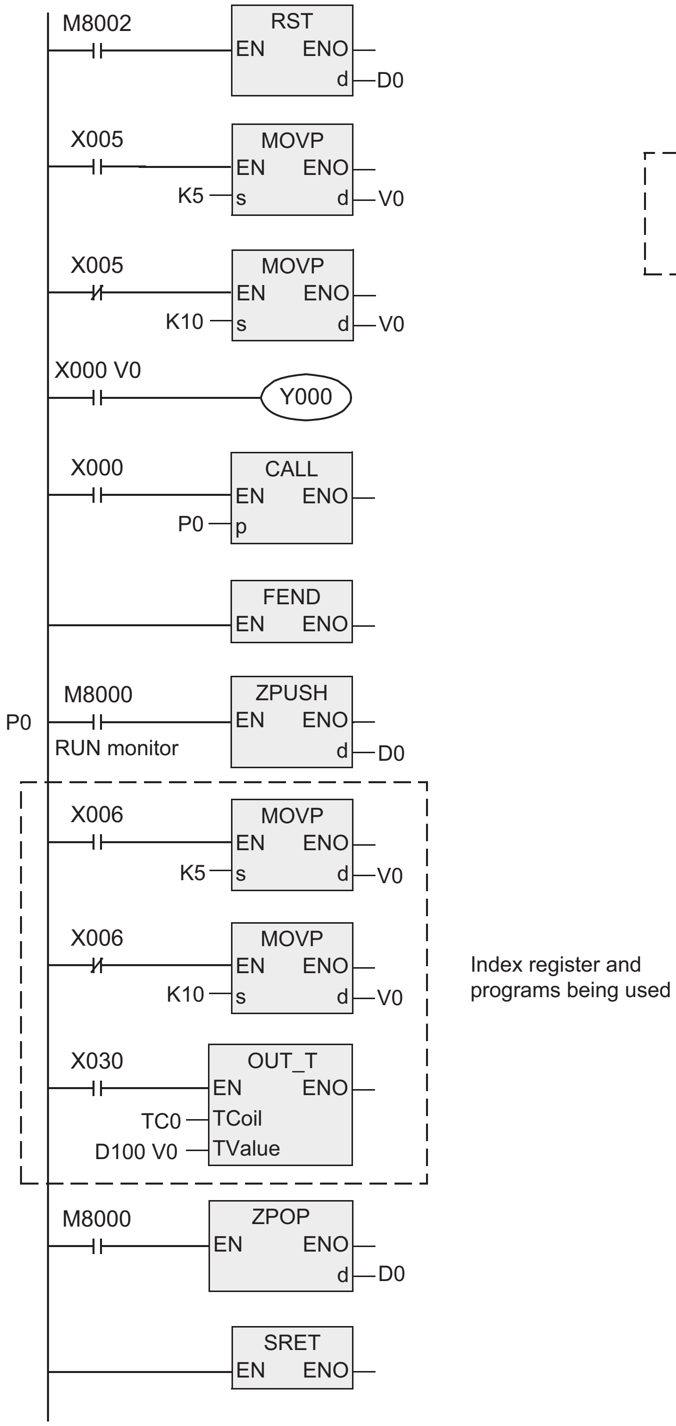
It is an operation error in the following case, and error flag M8067 is turned ON, and error code is stored in D8067.

- When the range of the number of points used after the device specified by d by ZPUSH exceeds the range of the corresponding device.
  (Error code: K6706)
- In ZPUSH instruction execution, when the number of retraction of the device specified by d is negative.
  (Error code: K6707)

Program example

This is a program for retracting the contents of index registers Z0 to Z7, V0 to V7 before execution of subroutine program after D0, when using the index register in the subroutine after pointer P0.

[Structured ladder]



[ST]

```
RST(M8002, D0);
MOVP(X005, K5, V0);
MOVP(NOT X005, K10, V0);
OUT(X000V0, Y000);
ZPUSH(M8000, D0);

FBDD(····················);
Function block of the function
corresponding to the subroutine
program.

ZPOP(M8000, D0);
```

7.11.2 ZPOP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction returns the contents of index registers V0 to V7, Z0 to Z7 once retracted by the ZPUSH instruction to the original state.

→ **As for ZPUSH (FNC102) instruction, refer to section 17.1.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
ZPOP	16 bits	Continuous	ZPOP (EN, ENO, d)	ZPOP(EN, d);
ZPOPP	16 bits	Pulse	ZPOPP (EN, ENO, d)	ZPOPP(EN, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	(d)	Head device once retracting the contents of index registers V0 to V7, Z0 to Z7 [(1+16 × times of retraction) occupied]	ANY16

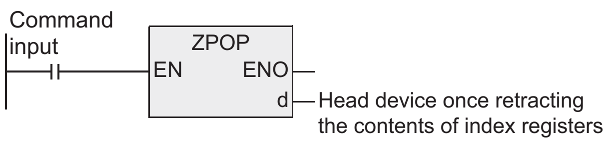
3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s3)														●	●									

Function and operation explanation

1. 16-bit operation (ZPOP/ZPOPP)

→ **As for the function and operation, refer also to section 7.11.1.**



1) The contents of index registers Z0 to Z7, V0 to V7 once retracted after the device specified by (d) by ZPUSH instruction are restored in the original index register. When the contents of the index register are restored, the number of retraction of the device specified by (d) is processed by -1.
2) Use ZPUSH instruction for temporary retraction of data.
   Use ZPUSH, ZPOP instructions in pair.

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Related instructions

Instruction	Content
ZPUSH	This instruction temporarily retracts the present values of index registers V0 to V7, Z0 to Z7.

Error

It is an operation error in the following case, and error flag M8067 is turned ON, and error code is stored in D8067.

- When the content of the number of retraction of the device specified by (d) during execution of ZPOP instruction is 0 or negative. (Error code: K6706)

Program example

→ **As for program examples, refer to section 7.11.1.**

7.12 Floating Point

7.12.1 DECMP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	○	×	×	×	×	×

Outline

This instruction compares two data (binary floating decimal point), and issues the result of greater, smaller, or equal to the bit device (3 points).

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DECMP	32 bits	Continuous	DECMP EN ENO s1 d s2	DECMP(EN, s1, s2, d);
DECMPP	32 bits	Pulse	DECMPP EN ENO s1 d s2	DECMPP(EN, s1, s2, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Device for storing binary floating point data to be compared	FLOAT(Single Precision)
	s2	Device for storing binary floating point data to be compared	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Head bit device for output of result (3 points occupied)	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1	▲2			●	●	●	▲1		
s2														●	▲1	▲2			●	●	●	▲1		
d		●	●			●	▲3												●					

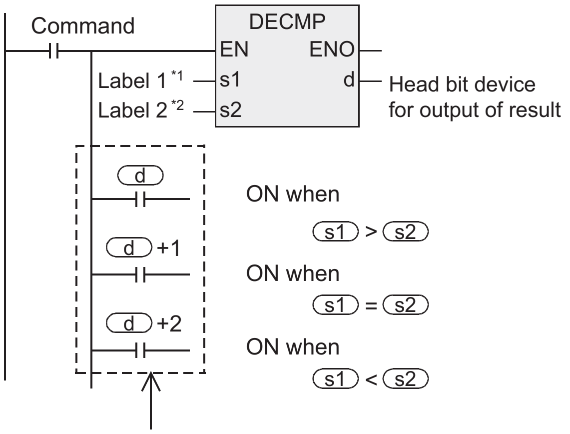
▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DECMP, DECMPP)

The compared value specified by (s1) and the comparison source specified by (s2) are compared as floating decimal point data, and depending on the result of greater, smaller, or equal, any bit of devices ((d), (d)+1, and (d)+2) specified by (d) is turned ON.

- When constants (K, H) are specified in the devices specified by (s1), (s2), the values are automatically converted from BIN and handled as binary floating decimal point data.



*1. To define the device for storing binary floating decimal point data (data 1) to be compared.

*2. To define the device for storing binary floating decimal point data (data 2) to be compared.

If the command input is OFF and DECMP instruction cannot be executed, the device specified by (d) holds the state before the command input is turned OFF.

Cautions

1. Number of devices occupied

(d) occupies 3 points.
Be careful not to overlap with the devices used in other applications.

2. Specification of input and output variables

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label. However, the 32-bit counter is a 32-bit long device, and can be specified directly. When specifying the device, use the global label.

3. The instruction is provided in the FX3G PLC Ver. 1.10 or later.

4. Applicable devices are limited.

▲1: FX3U, FX3UC and FX3G PLCs only are applicable.
▲2: FX3U, FX3UC PLCs only are applicable.
▲3: FX3U, FX3UC PLCs only are applicable.
However, index decoration is not applicable.

7.12.2 DEZCP

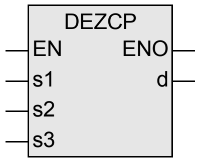
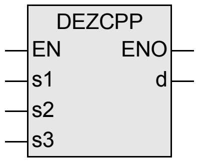
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	×	×	×

Outline

This instruction compares the comparison range of upper and lower two points and the data (binary floating decimal point), and issues the result to the bit device (3 points) depending on the greater, smaller or the band.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DEZCP	32 bits	Continuous		DEZCP(EN, s1, s2, s3, d);
DEZCPP	32 bits	Pulse		DEZCPP(EN, s1, s2, s3, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Device for storing binary floating point data to be compared	FLOAT(Single Precision)
	s2	Device for storing binary floating point data to be compared	FLOAT(Single Precision)
	s3	Device for storing binary floating point data to be compared	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Head bit device for output of result (3 points occupied)	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1	▲1			●	●	●	▲1		
s2														●	▲1	▲1			●	●	●	▲1		
s3														●	▲1	▲1			●	●	●	▲1		
d		●	●			●	▲2												●					

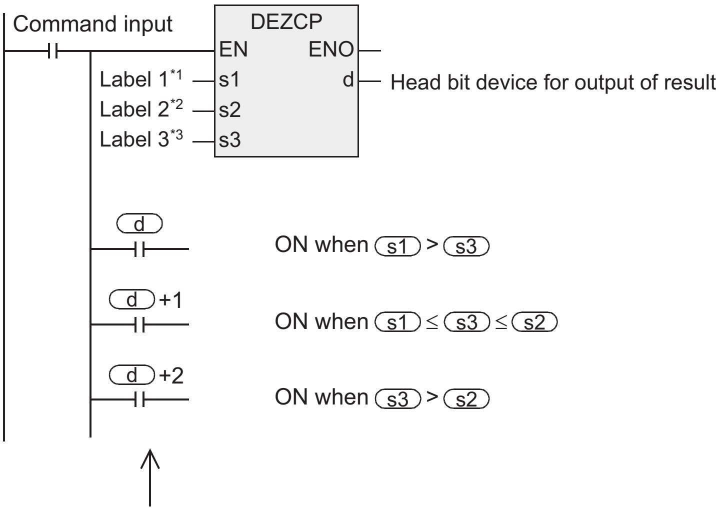
▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DEZCP, DEZCPP)

The compared value specified by s1, s2 and the comparison source specified by s3 are compared as floating decimal point data, and depending on the result of smaller, within range, or greater, any bit of devices (d, d+1, and d+2) specified by d is turned ON.

- When constants (K, H) are specified in the devices specified by s1, s2, s3, the values are automatically converted and handled as binary floating decimal point data.



If the command input is OFF and DEZCP instruction cannot be executed, the device specified by d holds the state before the command input is turned OFF.

*1. To define the device for storing binary floating decimal point data (data 1) to be compared.
*2. To define the device for storing binary floating decimal point data (data 2) to be compared.
*3. To define the device for storing binary floating decimal point data (data 3) to be compared.

Cautions

1. Number of devices occupied

d occupies 3 points.
Be careful not to overlap with the devices used in other applications.

2. Comparison data of s1 and s2

The magnitude relation of comparison data is s1 ≤ s2.
In the case of s1 > s2, the value of s2 is regarded to be same as s1, and is compared.

3. Specification of input and output variables

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

4. Applicable devices are limited.

▲1: FX3U, FX3UC PLCs only are applicable.
▲2: FX3U, FX3UC PLCs only are applicable.
However, index decoration is not applicable.

7.12.3 DEMOV

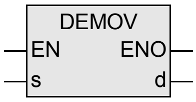
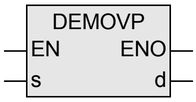
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	×	×	×	×	×	×

Outline

This instruction transfers binary floating point data.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	ST
DEMOV	32 bits	Continuous		DEMOV(EN, s, d);
DEMOVP	32 bits	Pulse		DEMOVP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Binary floating decimal point data of transfer source, or device storing the data	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	(d)	Transfer destination device of binary floating decimal point data	FLOAT(Single Precision)

3. Applicable devices

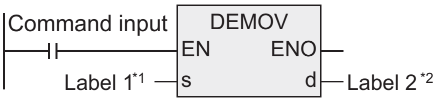
Operand type	Bit Devices: System user							Word Devices: Digit designation				System user				Special unit	Index			Others: Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	●	▲1			●			●		
(d)														●	●	▲1			●					

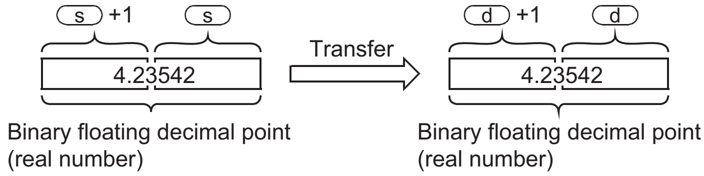
▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DEMOV/DEMOVP)

The content (binary floating decimal point data) of transfer source of device specified by (s) is transferred to the device specified by (d). Real number (E) can be directly specified in the device specified by (s).



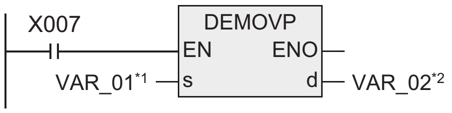


*1. To define the device storing the transfer source data.
*2. To define the transfer destination device.

Program example

1. This is a program for storing the real number of D11, D10 in D1, D0 when X007 is turned ON.

[Structured ladder]



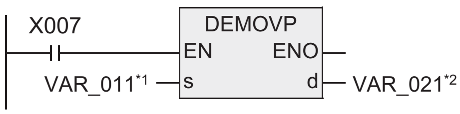
*1. VAR_01 is global label, and D10 is defined.
*2. VAR_02 is global label, and D0 is defined.

[ST]

```
DEMOVP(X007,VAR_01,VAR_02);
```

2. This is a program for storing the real number -1.23 in D11, D10 when X007 is turned ON.

[Structured ladder]



*1. VAR_011 is global label, and E-1.23 is defined.
*2. VAR_021 is global label, and D10 is defined.

[ST]

```
DEMOVP(X007,VAR_011,VAR_021);
```

Cautions

1) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
2) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
3) Applicable devices are limited.
   ▲1: FX3U, FX3UC PLCs only are applicable.

7.12.4 DESTR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction converts the binary floating decimal point data into character string (ASCII code) in a specified number of digits. You can also use the STR instruction for converting BIN data into character string (ASCII code).

→ As for character string, refer to FX Structured Programming Manual (Device & Common).
→ As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).
→ As for STR instruction, see section 7.20.1.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DESTR	32 bits	Continuous	DESTR EN ENO s1 d1 s2	DESTR(EN, s1, s2, d1);
DESTRP	32 bits	Pulse	DESTRP EN ENO s1 d1 s2	DESTRP(EN, s1, s2, d1);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s1)	Binary floating decimal point data to be converted, or device storing the data	FLOAT(Single Precision)
	(s2)	Head device storing the display specification of the numeric value to be converted (3 points occupied).	ANY16
Output variable	ENO	Execution state	Bit
	(d1)	Head of the storing destination device of the converted character string (all digits + 1) ÷2 points occupied (fractions are rounded up).	String

3. Applicable devices

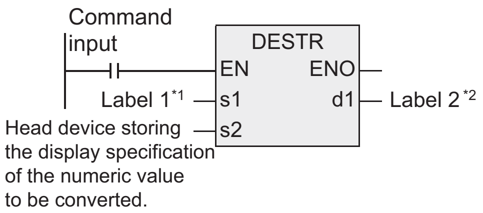
Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Special unit	Index			Others: Con stant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)														●	●	●			●			●		
(s2)								●	●	●	●	●	●	●	●	●			●					
(d1)									●	●	●	●	●	●	●	●			●					

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

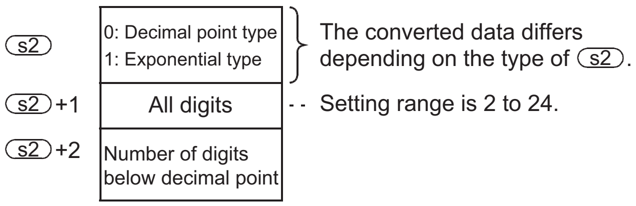
1. 32-bit operation (DESTR/DESTRP)

The content (binary floating decimal point data) of the device specified by (s1) is converted into character string depending on the content of the device specified by (s2), and stored in or after the device specified by (d). Real number can be directly specified in the device specified by (s1).

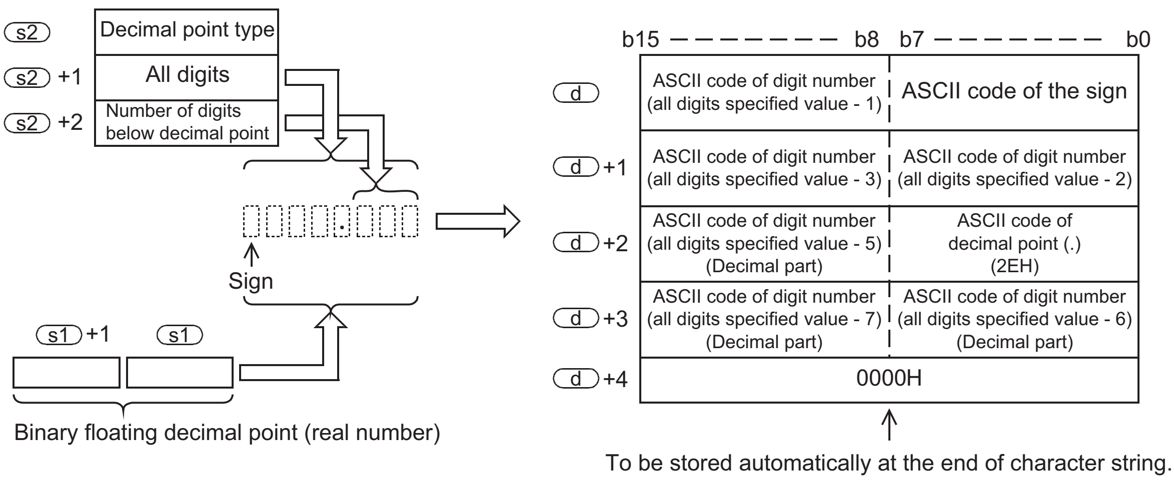


*1. To define the binary floating decimal point data to be converted, or the device storing the data.
*2. To define the head of the storing destination device of the converted character string.

- The converted data differs depending on the display specification specified by (s2).

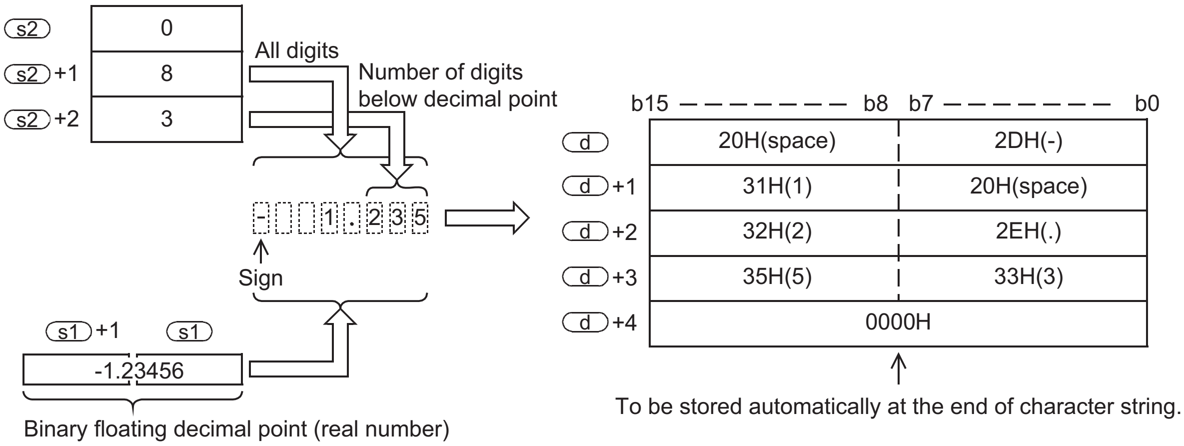


2. In the case of decimal point type

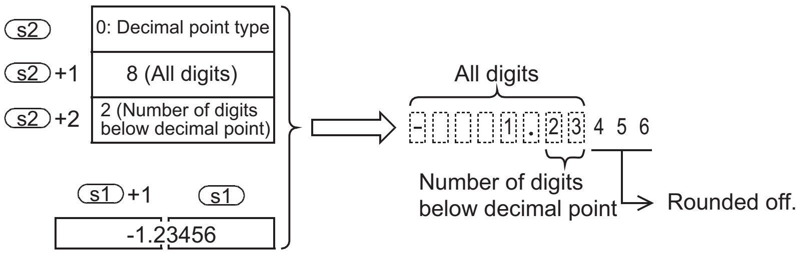


- The number of all digits that can be specified by (s2)+1 is as follows. (Maximum: 24 digits)
  When the number of digits below decimal point is "0" . . . . .All digits ≥ 2
  When the number of digits below decimal point is other than "0". . . .All digits ≥ (decimal digits + 3)
- Decimal digits that can be specified by (s2)+2 is 0 to 7 digits.
  However, please set in the range of decimal digits ≤ (all digits - 3).

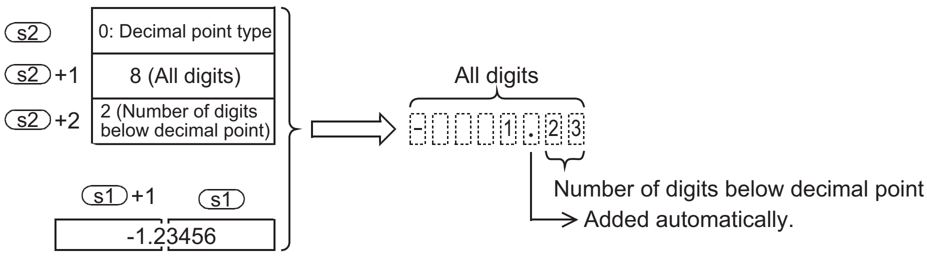
For example, in the case of all digits of 8 and decimal digits of 3, when -1.23456 is specified, the data after (d) is stored as follows.



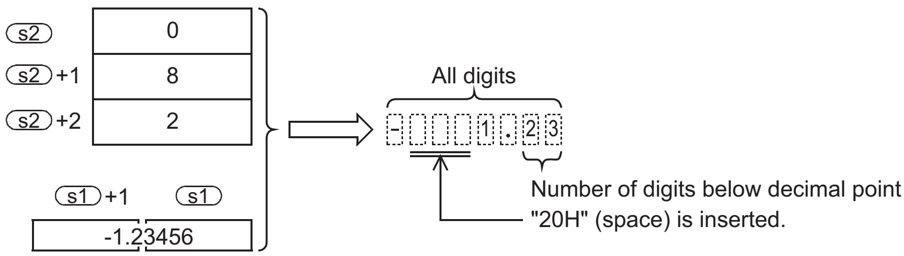
- The converted character string data is stored in the device after (d) as follows.
  - For the sign, "20H" (space) is stored when the binary floating decimal point data is positive, and "2DH" (-) is stored when negative.
  - When the decimal part of binary floating decimal point data does not settle within the decimal digits, the lower decimal digits are rounded off.



  - When the number of decimal digits is set in other than "0", automatically "2EH" (.) is stored at the specified decimal digits + 1 digit.
    However, when the number of decimal digits is "0", "2EH" (.) is not stored.

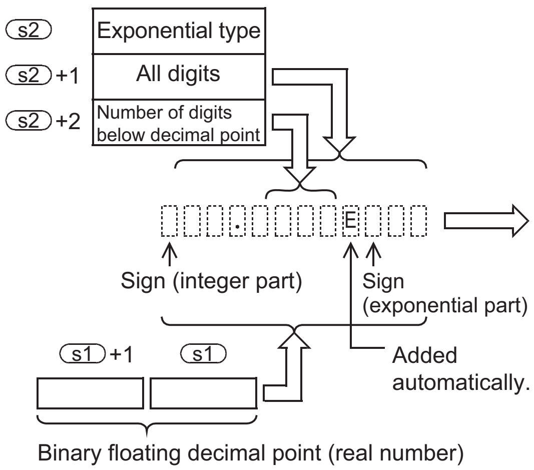


  - When the number of digits subtracting the sign, decimal point, and decimal part from the total number of digits is greater than the integer part of the binary floating decimal point data, "20H" (space) is inserted between the sign and the integer part.



  - "00H" or "0000H" is automatically stored at the end of the converted character string.

3. In the case of exponential type

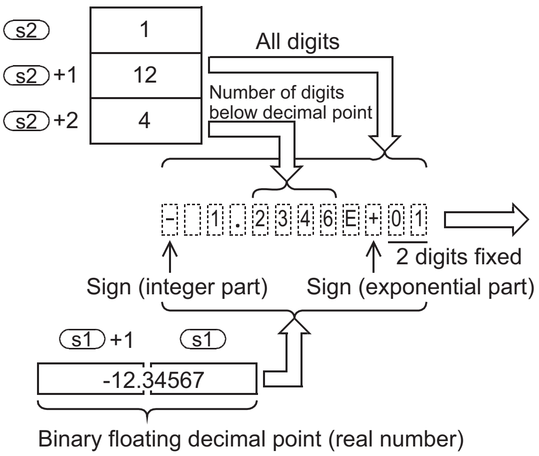


	b15 ― b8	b7 ― b0
d	ASCII code of digit number (all digits specified value - 1)	ASCII code of the sign
d +1	ASCII code of decimal point (.) (2EH)	ASCII code of digit number (all digits specified value - 2)
d +2	ASCII code of digit number (all digits specified value - 5) (Decimal part)	ASCII code of digit number (all digits specified value - 4) (Decimal part)
d +3	ASCII code of digit number (all digits specified value - 7) (Decimal part)	ASCII code of digit number (all digits specified value - 6) (Decimal part)
d +4	ASCII code of the sign (exponential part)	45H(E)
d +5	ASCII code of digit number (all digits specified value - 11) (exponential part)	ASCII code of digit number (all digits specified value - 10) (exponential part)
d +6	0000H	

To be stored automatically at the end of character string.

- The number of all digits that can be specified by (s2)+1 is as follows. (Maximum: 24 digits)
  When the number of digits below decimal point is "0" . . . . .No. of digits ≥ 6
  When the number of digits below decimal point is other than "0". . . .No. of digits ≥ (decimal digits + 7)
- Decimal digits that can be specified by (s2)+2 is 0 to 7 digits. Please set within the range of decimal digits ≤ (total digits - 7).

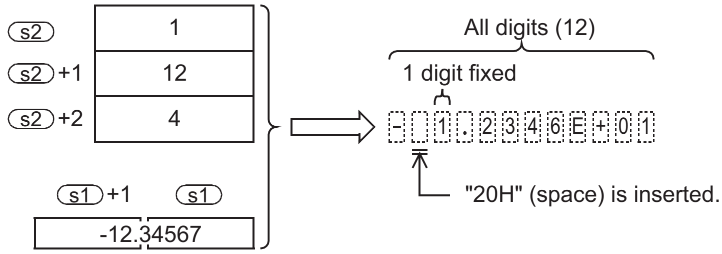
For example, in the case of all digits of 12 and decimal digits of 4, when -12.34567 is specified, the data after (d) is stored as follows.



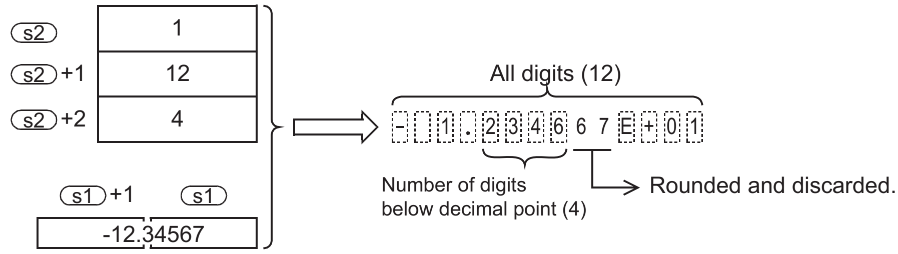
	b15 ― b8	b7 ― b0
d	20H (space)	2DH(-)
d +1	2EH(.)	31H(1)
d +2	33H(3)	32H(2)
d +3	36H(6)	34H(4)
d +4	2BH(+)	45H(E)
d +5	31H(1)	30H(0)
d +6	0000H	

To be stored automatically at the end of character string.

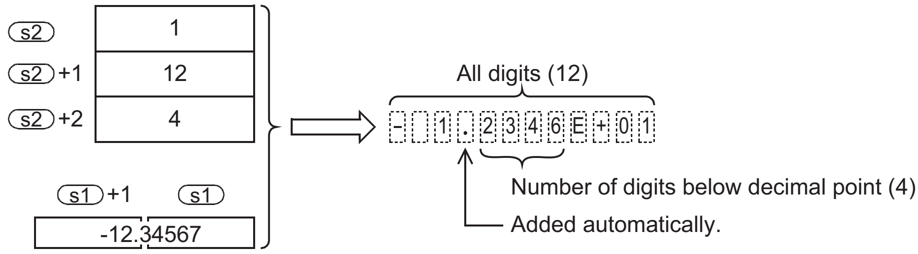
- The converted character string data is stored in the device after (d) as follows.
  - For the sign of integer part, "20H" (space) is stored when the binary floating decimal point data is positive, and "2DH" (-) is stored when negative.
  - The integer part is fixed in one digit.
    Between the integer part and the sign, "20H" (space) is inserted.



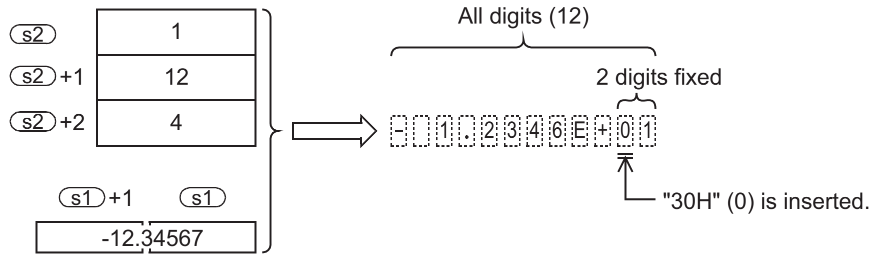
  - When the decimal part of binary floating decimal point data does not settle within the decimal digits, the lower decimal digits are rounded off.



  - When the number of decimal digits is set in other than "0", automatically "2EH" (.) is stored at the specified decimal digits + 1 digit.
    However, when the number of decimal digits is "0", "2EH" (.) is not stored.



  - For the sign of exponential part, "2BH" (+) is stored when the index is positive, and "2DH" (-) is stored when negative.
  - The exponential part is fixed in two digits.
    When the exponential part is one digit, between the exponential part and the sign, "30H" (0) is inserted.



  - "00H" or "0000H" is automatically stored at the end of the converted character string.

Cautions

When handling character string data or 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling character string data or 32-bit data, please use the label. However, the 32-bit counter is a 32-bit long device, and can be specified directly. When specifying the device, use the global label.

Related instructions

Instruction	Content
EVAL	This instruction converts the character string (ASCII code) data into binary floating decimal point data.
STR	This instruction converts BIN data into character string (ASCII code).
VAL	This instruction converts the character string (ASCII code) data into BIN data.

Error

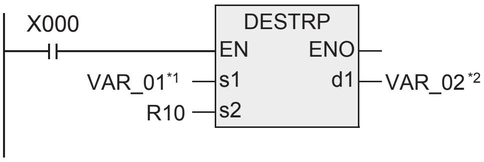
In the following cases, it is an operation error, and error flag (M8067) is turned ON, and error code is stored in D8067.

- When (s1) is not within the following range. (Error code: K6706)
  $0, \pm 2^{-126} \leq$ (s1) $< \pm 2^{128}$
- When the type specification specified by (s2) is other than 0, 1. (Error code: K6706)
- When the all digits specification specified by (s2)+1 is out of the following range. (Error code: K6706)
  In the case of decimal point type
  When the number of digits below decimal point is "0" . . . . All digits ≥ 2
  When the number of digits below decimal point is other than "0" . . All digits ≥ (decimal digits + 3)
  In the case of exponential type
  When the number of digits below decimal point is "0" . . . . All digits ≥ 6
  When the number of digits below decimal point is other than "0" . . All digits ≥ (decimal digits + 7)
- When the decimal digits specification specified by (s2)+2 is out of the following range. (Error code: K6706)
  In the case of decimal point type: Decimal digits ≤ (all digits - 3)
  In the case of index type: Decimal digits ≤ (all digits - 7)
- When the device range for storing the character string specified by (d) is over the range of the corresponding device.
  (Error code: K6706)
- When the result of conversion exceeds the specified all digits. (Error code: K6706)

Program example

1) This is a program for converting the content (binary floating decimal point data) of R0, R1 depending on the content specified by R10 to R12 when the X000 is turned ON, and storing after the D0.

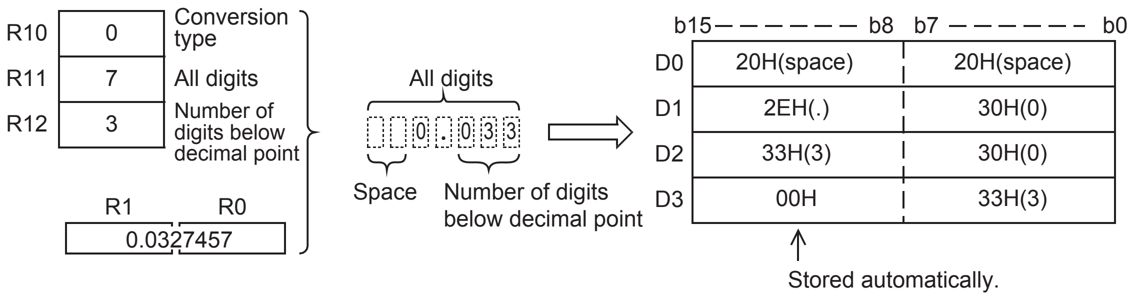
[Structured ladder]



[ST]

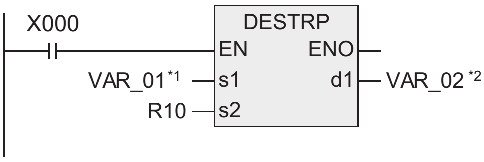
```
DESTRP(X000,VAR_01,R10,VAR_02);
```

*1. VAR_01 is global label, and R10 is defined.
*2. VAR_02 is global label, and D0 is defined.



2) This is a program for converting the content (binary floating decimal point data) of R0, R1 depending on the content specified by R10 to R12 when the X000 is turned ON, and storing after the D10.

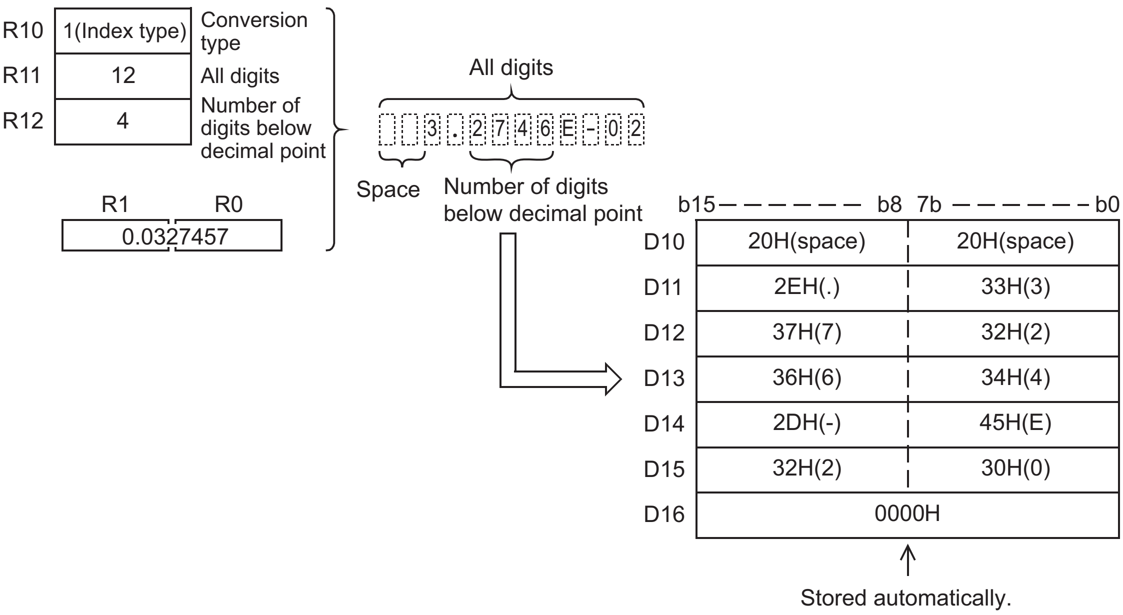
[Structured ladder]



[ST]

```
DESTRP(X000,VAR_01,R10,VAR_02);
```

*1. VAR_01 is global label, and R0 is defined.
*2. VAR_02 is global label, and D10 is defined.



7.12.5 DEVAL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction converts the character string (ASCII code) into binary floating decimal point data.
You can also use the VAL instruction for converting the character string (ASCII code) into BIN data.

→ **As for character string, refer to FX Structured Programming Manual (Device & Common).**
→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**
→ **As for VAL instruction, refer to section 7.20.2.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DEVAL	32 bits	Continuous	DEVAL (EN, ENO, s, d)	DEVAL(EN, s, d);
DEVALP	32 bits	Pulse	DEVALP (EN, ENO, s, d)	DEVALP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device storing the character string to be converted into binary floating point decimal data.	String
Output variable	ENO	Execution state	Bit
	(d)	Device for storing the converted binary floating point decimal data.	FLOAT(Single Precision)

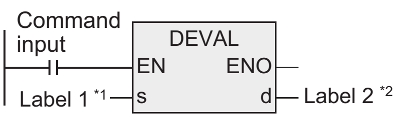
3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit designation				System user				Special unit	Index			Others: Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	●	●			●					
(d)														●	●	●			●					

Function and operation explanation

1. 32-bit operation (EVAL/EVALP)

The character string stored after the device specified by (s) is converted into binary decimal floating point data, and stored in the device specified by (d).



*1. To define the head device storing the character string data to be converted into binary floating point decimal data.

*2. To define the device for storing the converted binary floating point decimal data.

The character string specified can be converted into binary floating point decimal data whether in decimal point type or in specified type.

	b15 ––– b8	b7 ––– b0
s	ASCII code of first character	ASCII code of the sign
s +1	ASCII code of third character	ASCII code of second character
s +2	ASCII code of fifth character	ASCII code of fourth character
s +3	ASCII code of seventh character	ASCII code of sixth character
s +4		00H

Showing the end of character string.

⇒ d +1, d : Binary floating decimal point (real number)

a) In the case of decimal point type

	b15 ––– b8	b7 ––– b0
s	31H(1)	2DH(-)
s +1	30H(0)	2EH(.)
s +2	38H(8)	37H(7)
s +3	32H(2)	31H(1)
s +4		00H

- 1 . 0 7 8 1 2

⇒ d +1, d : -1.07812

Binary floating decimal point (real number)

b) In the case of exponential type

	b15 ––– b8	b7 ––– b0
s	20H(space)	2DH(-)
s +1	2EH(.)	31H(1)
s +2	32H(2)	33H(3)
s +3	31H(1)	30H(0)
s +4	2BH(+)	45H(E)
s +5	30H(0)	31H(1)
s +6		00H

-   1 . 3 2 0 1 E + 1 0

⇒ d +1, d : -1.3201E+10

Binary floating decimal point (real number)

- When the character string to be converted into the binary floating decimal point specified by s is more than 7 digits excluding the sign, decimal point, and the exponential part, the data after the seventh digit is cut off.

a) In the case of decimal point type

	b15 ––– b8	b7 ––– b0
s	20H(space)	2DH(-)
s +1	31H(1)	20H(space)
s +2	33H(3)	2EH(.)
s +3	31H(1)	30H(0)
s +4	36H(6)	35H(5)
s +5	31H(1)	38H(8)
s +6	00H	32H(2)

-     1 . 3 0 1 5 6 8 1 2

To be cut off.

⇒ d +1, d : -1.30156

Matched exactly up to the sixth digit.

Binary floating decimal point (real number)

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

b) In the case of exponential type

	b15 — b8	b7 — b0
s	20H(space)	2DH(-)
s+1	2EH(.)	31H(1)
s+2	35H(5)	33H(3)
s+3	33H(3)	30H(0)
s+4	31H(1)	34H(4)
s+5	45H(E)	32H(2)
s+6	30H(0)	2DH(-)
	00H	32H(2)

d+1 d
-1.35034E-2
Binary floating decimal point (real number)

- 1 . 3 5 0 3 4 1 2 E - 0 2
To be cut off.

- To be converted as positive value when "2BH" (+) is specified by the sign or the sign is omitted in the decimal point type. To be converted as negative value when "2DH" (-) is specified by the sign.
- To be converted as positive value when "2BH" (+) is specified in the exponential part sign or the sign is omitted in the exponential type. To be converted as negative value when "2DH" (-) is specified in the exponential part sign.
- When "20H" (space) or "30H" (0) is present among the numeric values except for the first "0" in the character string specified by s, "20H" or "30H" is ignored in the converting operation.

	b15 — b8	b7 — b0
s	20H(space)	2DH(-)
s+1	31H(1)	30H(0)
s+2	32H(2)	2EH(.)
s+3	31H(1)	33H(3)
s+4		00H

d+1 d
-1.231
Binary floating decimal point (real number)

- 0 1 . 2 3 1
To be ignored.

- When "30H" (0) is present between "E" and the numeric value in the character string in the exponential type, "30H" is ignored in the converting operation.

b15 — b8	b7 — b0
20H(space)	2DH(-)
2EH(.)	31H(1)
34H(4)	30H(0)
33H(3)	35H(5)
2BH(+)	45H(E)
33H(3)	30H(0)
	00H

d+1 d
-1.0453E+3

- 1 . 0 4 5 3 E + 0 3
To be ignored.

- The character string can be set in a maximum of 24 characters.
In the character string, "20H" (space) or "30H" (0) is counted as one character.

Caution

When handling character string data or 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling character string data or 32-bit data, please use the label. However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

Related devices

→ **As for the manner of using the zero, borrow, or carry flag, refer to the FX Structured Programming Manual (Device & Common).**

Device	Name	Content	
		Condition	Operation
M8020	Zero	The conversion result is really zero (when the mantissa part is "0")	Zero flag (M8020) is ON.
M8021	Borrow	Absolute value of conversion result $< 2^{-126}$	The value of (d) is the smallest value (2^{-126}) of 32-bit real number, and borrow flag (M8021) is ON.
M8022	Carry	Absolute value of conversion result $\geq 2^{128}$	The value of (d) is the largest value (2^{128}) of 32-bit real number, and carry flag (M8022) is ON.

Related instructions

Instruction	Content
ESTR	This instruction converts the binary floating decimal point data into character string (ASCII code).
STR	This instruction converts BIN data into character string (ASCII code).
VAL	This instruction converts the character string (ASCII code) data into BIN data.

Error

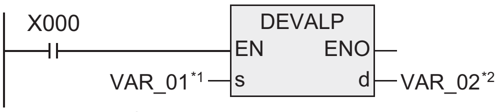
In the following cases, it is an operation error, error flag (M8067) is turned ON, and error code is stored in D8067.

- When other characters than "30H" (0) to "39H" (9) are present in the integer part or the decimal part. (Error code : K6706)
- When two or more "2EH" (.) are present in the character string specified by (s). (Error code : K6706)
- When other characters than "45H" (E), "2BH" (+), or "2DH" (-) are present in the exponential part, or when there are plural exponential parts. (Error code : K6706)
- When "00H" is not present in the corresponding device range from (s). (Error code : K6706)
- When the number of characters after (s) is 0 or exceeds 24 characters. (Error code : K6706)

Program example

1) This is a program for converting the character string stored after R0 when the X000 is turned ON, into binary floating decimal point, and storing in D0, D1.

[Structured ladder]

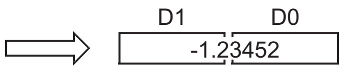


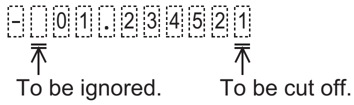
*1. VAR_01 is global label, and R0 is defined.
*2. VAR_02 is global label, and D0 is defined.

[ST]

```
DEVAR(X000,VAR_01,VAR_02);
```

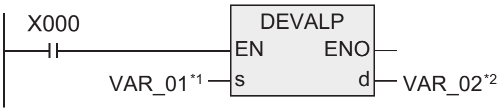
	b15 ― b8	b7 ― b0
R0	20H(space)	2DH(-)
R1	31H(1)	30H(0)
R2	32H(2)	2EH(.)
R3	34H(4)	33H(3)
R4	32H(2)	35H(5)
R5	00H	31H(1)





2) This is a program for converting the character string stored after D10 when the X000 is turned ON, into binary floating decimal point, and storing in D100, D101.

[Structured ladder]

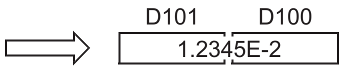


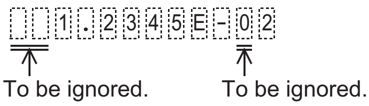
*1. VAR_01 is global label, and D10 is defined.
*2. VAR_02 is global label, and D100 is defined.

[ST]

```
DEVALP(X000,VAR_01,VAR_02);
```

	b15 ------ b8	b7 ------ b0
D10	20H(space)	20H(space)
D11	2EH(.)	31H(1)
D12	33H(3)	32H(2)
D13	35H(5)	34H(4)
D14	2DH(-)	45H(E)
D15	32H(2)	30H(0)
D16		00H





Operation in overflow, underflow, zero mode

Condition	Operation
Absolute value of conversion result $< 2^{-126}$	The value of (d) is the smallest value (2^{-126}) of 32-bit real number, and borrow flag (M8021) is ON.
Absolute value of conversion result $\geq 2^{128}$	The value of (d) is the largest value (2^{128}) of 32-bit real number, and carry flag (M8022) is ON.
The conversion result is really zero (when the mantissa part is "0")	Zero flag (M8020) is ON.

7.12.6 DEBCD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	×	×	×

Outline

This instruction converts the binary floating decimal point in the device into decimal floating decimal point.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DEBCD	32 bits	Continuous	DEBCD EN ENO s d	DEBCD(EN, s, d);
DEBCDP	32 bits	Pulse	DEBCDP EN ENO s d	DEBCDP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Device for storing binary floating decimal point data.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Device for storing the converted decimal floating decimal point data.	ANY32

3. Applicable devices

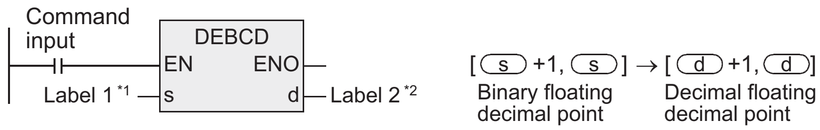
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	▲1	▲1			●					
d														●	▲1	▲1			●					

▲: Refer to "Cautions".

Function and operation explanation

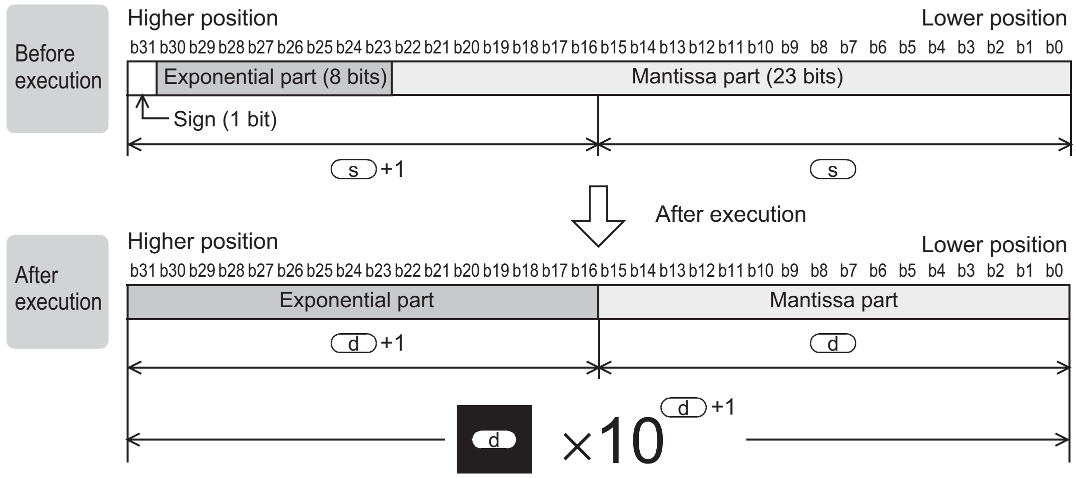
1. 32-bit operation (DEBCD, DEBCDP)

The binary floating decimal point of the device specified by (s) is converted into decimal floating decimal point, and is transferred to the device specified by (d).



*1. To define the device for storing the binary floating decimal point data.

*2. To define the device for storing the converted decimal floating decimal point data.



Cautions

1) In the floating decimal point operation, all operations are executed at the binary floating decimal point. However, since the binary floating decimal point is a difficult numeric value (exclusive monitor method), by converting it into decimal floating decimal point, it is easier for monitoring by peripheral devices or the like.
Meanwhile, GX Works 2 or GOT is provided with a function for monitoring or displaying the binary floating decimal point directly.

2) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly. When specifying the device, use the global label.

3) Applicable devices are limited.
▲: FX3U, FX3UC PLCs only are applicable.

7.12.7 DEBIN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	×	×	×

Outline

This instruction converts the decimal floating decimal point in the device into binary floating decimal point.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DEBIN	32 bits	Continuous	DEBIN EN ENO s d	DEBIN(EN, s, d);
DEBINP	32 bits	Pulse	DEBINP EN ENO s d	DEBINP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	ENO	Execution condition	Bit
	s	Device for storing decimal floating decimal point data.	ANY32
Output variable	ENO	Execution state	Bit
	d	Device for storing the converted binary floating decimal point data.	FLOAT(Single Precision)

3. Applicable devices

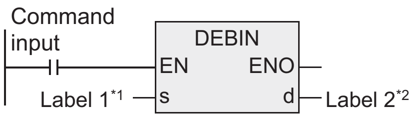
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	▲1	▲1			●					
d														●	▲1	▲1			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DEBIN, DEBINP)

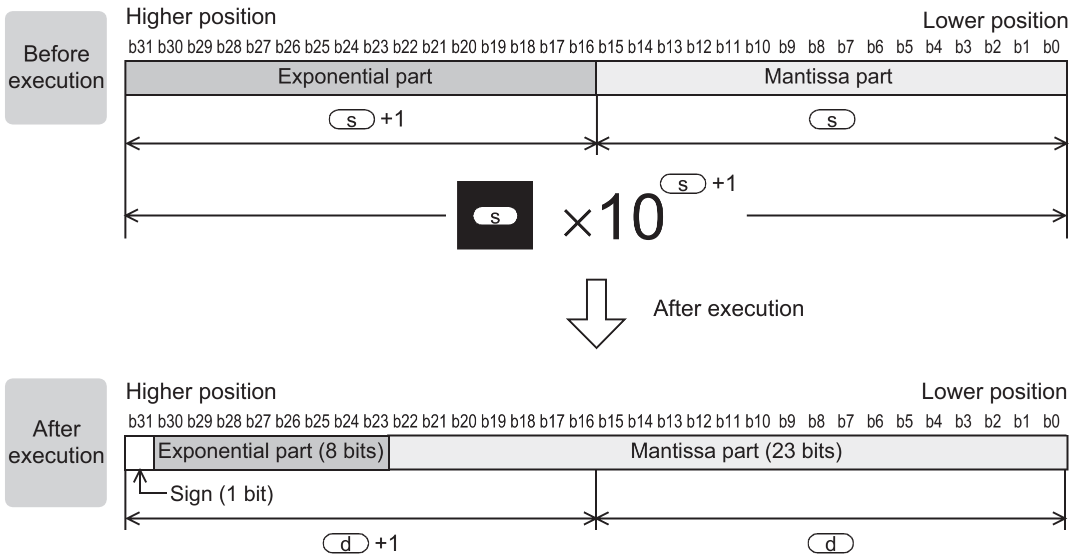
The decimal floating decimal point of the device specified by (s) is converted into binary floating decimal point, and is transferred to the device specified by (d).



[(s)+1, (s)] → [(d)+1, (d)]
Decimal floating decimal point → Binary floating decimal point

*1. To define the device for storing the decimal floating decimal point data.

*2. To define the device for storing the converted binary floating decimal point data.



Cautions

1) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

2) Applicable devices are limited.
▲: FX3U, FX3UC PLCs only are applicable.

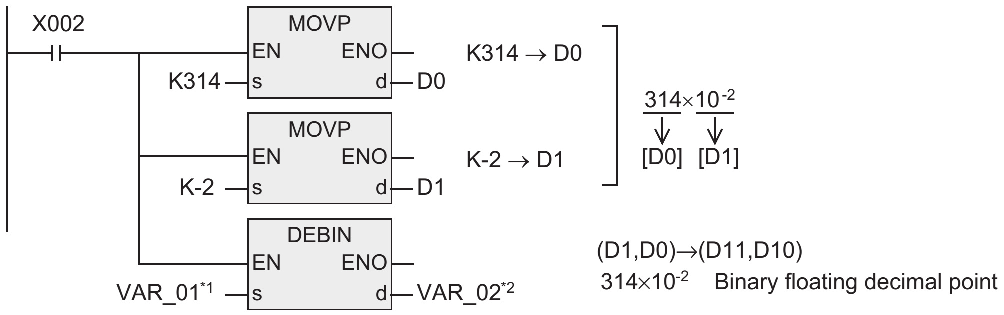
Program example

By using the DEBIN instruction, the numeric value including the decimal point can be directly converted into the binary floating decimal point.

Example: Binary floating decimal point conversion of 3.14

$3.14=314\times10^{-2}$ (Decimal floating decimal point)

[Structured ladder]



*1. VAR_01 is global label, and D0 is defined.
*2. VAR_02 is global label, and D10 is defined.

[ST]

```
MOVP(X002,K314,D0);
MOVP(X002,K-2,D1);
DEBIN(Y002,VAR_01,VAR_02);
```

→ **As for program example of floating decimal point operation, refer to section 7.5.10.**

7.12.8 DEADD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	○	×	×	×	×	×

Outline

This instruction adds two binary floating decimal points.

→ As for program example of floating decimal point operation, refer to section 7.5.10.
→ As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).
→ As for the operation of the flag, refer to the FX Structured Programming Manual (Device & Common).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DEADD	32 bits	Continuous	DEADD EN ENO s1 d s2	DEADD(EN, s1, s2, d);
DEADDP	32 bits	Pulse	DEADDP EN ENO s1 d s2	DEADDP(EN, s1, s2, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Device for storing binary floating decimal point data to be added.	FLOAT(Single Precision)
	s2	Device for storing binary floating decimal point data to be added.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Device for storing the added binary floating decimal point data.	FLOAT(Single Precision)

3. Applicable devices

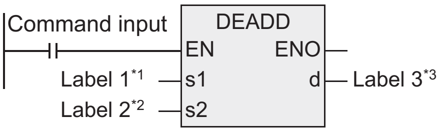
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1	▲2			●	●	●	▲1		
s2														●	▲1	▲2			●	●	●	▲1		
d														●	▲1	▲2			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DEADD, DEADDP)

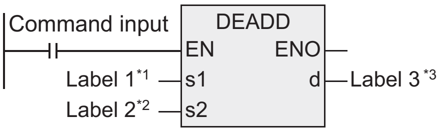
The binary floating decimal point data in the device specified by (s1) and in the device specified by (s2) are added, and the result is transferred to the device specified by (d) in binary floating decimal point.



[(s1)+1, (s1)] + [(s2)+1, (s2)] → [(d)+1, (d)]
Binary floating decimal point | Binary floating decimal point | Binary floating decimal point

*1.To define the device for storing binary floating decimal point data to be added.
*2.To define the device for storing binary floating decimal point data to be added.
*3.To define the device for storing the added binary floating decimal point data.

When constants (K, H) are specified in the device specified by (s1) or in the device specified by (s2), the values are automatically converted and handled as binary floating decimal point.



[(s1)+1, (s1)] + [K2346] → [(d)+1, (d)]
Binary floating decimal point | Automatic conversion into binary floating decimal point | Binary floating decimal point

*1. To define the device for storing binary floating decimal point data to be added.
*2. To define the device for storing binary floating decimal point data to be added.
*3. To define the device for storing the added binary floating decimal point data.

Cautions

1) When the same devices are specified, the same device numbers can be specified in (s1) and (s2) and (d). In this case, when the continuous execution type instruction (DEADD) is used, it must be noted that the addition result changes in every operation cycle.
2) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
3) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
4) Applicable devices are limited.
   ▲1: FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

7.12.9 DESUB

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	○	×	×	×	×	×

Outline

This instruction subtracts two binary floating decimal points.

→ As for program example of floating decimal point operation, refer to section 7.5.10.

→ As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).

→ As for the operation of the flag, refer to the FX Structured Programming Manual (Device & Common).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DESUB	32 bits	Continuous	DESUB (EN, ENO; s1, d; s2)	DESUB(EN, s1, s2, d);
DESUBP	32 bits	Pulse	DESUBP (EN, ENO; s1, d; s2)	DESUBP(EN, s1, s2, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Device for storing binary floating decimal point data to be subtracted.	FLOAT(Single Precision)
	s2	Device for storing binary floating decimal point data to be subtracted.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Device for storing the subtracted binary floating decimal point data.	FLOAT(Single Precision)

3. Applicable devices

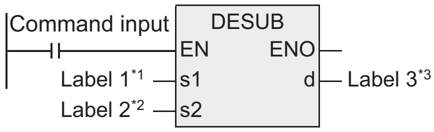
Operand type	Bit Devices							Word Devices													Others					
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1	▲2			●	●	●	▲1		
s2														●	▲1	▲2			●	●	●	▲1		
d														●	▲1	▲2			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DESUB, DESUBP)

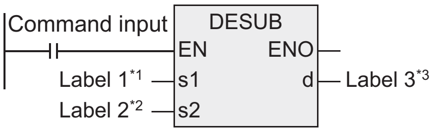
The binary floating decimal point data in the device specified by (s2) are subtracted from the device specified by (s1), and the result is transferred to the device specified by (d) in binary floating decimal point.



[(s1)+1, (s1)] - [(s2)+1, (s2)] → [(d)+1, (d)]
Binary floating decimal point | Binary floating decimal point | Binary floating decimal point

*1. To define the device for storing binary floating decimal point data to be subtracted.
*2. To define the device for storing binary floating decimal point data to be subtracted.
*3. To define the device for storing the subtracted binary floating decimal point data.

When constants (K, H) are specified in the device specified by (s1) or in the device specified by (s2), the values are automatically converted and handled as binary floating decimal point.



[(s1)+1, (s1)] - [K2346] → [(d)+1, (d)]
Binary floating decimal point | Automatic conversion into binary floating decimal point | Binary floating decimal point

*1. To define the device for storing binary floating decimal point data to be subtracted.
*2. To define the device for storing binary floating decimal point data to be subtracted.
*3. To define the device for storing the subtracted binary floating decimal point data.

Cautions

1) When the same devices are specified, the same device numbers can be specified in (s1) and (s2) and (d). In this case, when the continuous execution type instruction (DESUB) is used, it must be noted that the subtraction result changes in every operation cycle.
2) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
3) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
4) Applicable devices are limited.
   ▲1: FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

7.12.10 DEMUL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	○	×	×	×	×	×

Outline

This instruction multiplies two binary floating decimal points.

→ As for program example of floating decimal point operation, refer to section 7.5.10.
→ As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DEMUL	32 bits	Continuous	DEMUL EN ENO s1 d s2	DEMUL(EN, s1, s2, d);
DEMULP	32 bits	Pulse	DEMULP EN ENO s1 d s2	DEMULP(EN, s1, s2, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Device for storing binary floating decimal point data to be multiplied.	FLOAT(Single Precision)
	s2	Device for storing binary floating decimal point data to be multiplied.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Device for storing the multiplied binary floating decimal point data.	FLOAT(Single Precision)

3. Applicable devices

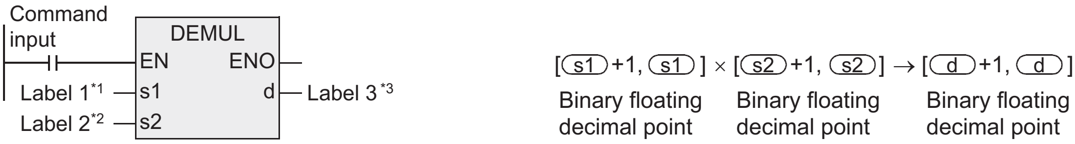
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1	▲2			●	●	●	▲1		
s2														●	▲1	▲2			●	●	●	▲1		
d														●	▲1	▲2			●					

▲: Refer to "Cautions".

Function and operation explanation

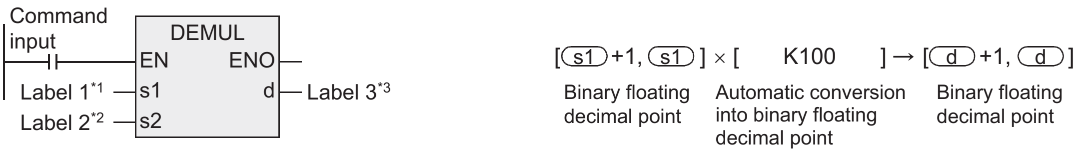
1. 32-bit operation (DEMUL, DEMULP)

The binary floating decimal point data in the device specified by (s1) and in the device specified by (s2) are multiplied, and the result is transferred to the device specified by (d) in binary floating decimal point.



*1. To define the device for storing binary floating decimal point data to be multiplied.
*2. To define the device for storing binary floating decimal point data to be multiplied.
*3. To define the device for storing the multiplied binary floating decimal point data.

When constants (K, H) are specified in the device specified by (s1) or in the device specified by (s2), the values are automatically converted and handled as binary floating decimal point.



*1. To define the device for storing binary floating decimal point data to be multiplied.
*2. To define the device for storing binary floating decimal point data to be multiplied.
*3. To define the device for storing the multiplied binary floating decimal point data.

Cautions

1) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
2) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
3) Applicable devices are limited.
   ▲1: FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

7.12.11 DEDIV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	○	×	×	×	×	×

Outline

This instruction divides two binary floating decimal points.

→ As for program example of floating decimal point operation, refer to section 7.5.10.
→ As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).
→ As for the operation of the flag, refer to the FX Structured Programming Manual (Device & Common).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DEDIV	32 bits	Continuous	DEDIV (EN, s1, s2 → ENO, d)	DEDIV(EN, s1, s2, d);
DEDIV	32 bits	Pulse	DEDIVP (EN, s1, s2 → ENO, d)	DEDIVP(EN, s1, s2, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Device for storing binary floating decimal point data to be divided.	FLOAT(Single Precision)
	s2	Device for storing binary floating decimal point data to be divided.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Device for storing the divided binary floating decimal point data.	FLOAT(Single Precision)

3. Applicable devices

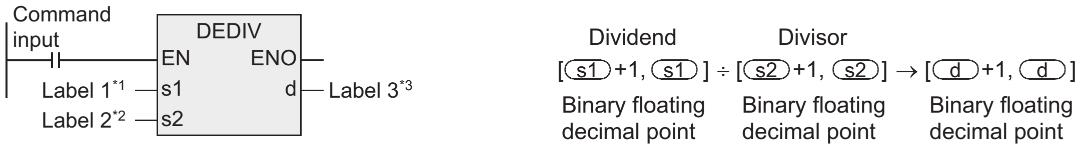
Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1	▲2			●	●	●	▲1		
s2														●	▲1	▲2			●	●	●	▲1		
d														●	▲1	▲2			●					

▲: Refer to "Cautions".

Function and operation explanation

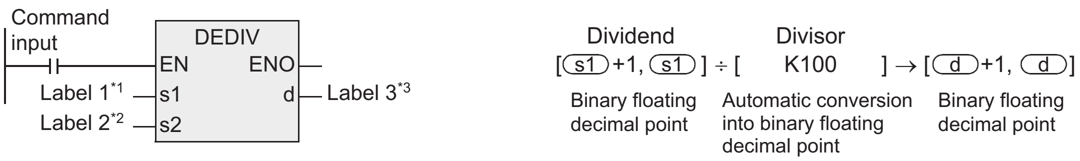
1. 32-bit operation (DEDIV, DEDIVP)

The binary floating decimal point data in the device specified by (s1) and in the device specified by (s2) are divided, and the result is transferred to the device specified by (d) in binary floating decimal point.



*1. To define the device for storing binary floating decimal point data to be divided.
*2. To define the device for storing binary floating decimal point data to be divided.
*3. To define the device for storing the divided binary floating decimal point data.

When constants (K, H) are specified in the device specified by (s1) or in the device specified by (s2), the values are automatically converted and handled as binary floating decimal point.



*1. To define the device for storing binary floating decimal point data to be divided.
*2. To define the device for storing binary floating decimal point data to be divided.
*3. To define the device for storing the divided binary floating decimal point data.

Cautions

1) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
2) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
3) Applicable devices are limited.
   ▲1: FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

7.12.12 DEXP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction executes exponential operation whose base is "e (2.71828)".

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DEXP	32 bits	Continuous	DEXP EN ENO s d	DEXP(EN, s, d);
DEXPP	32 bits	Pulse	DEXPP EN ENO s d	DEXPP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device for storing binary floating decimal point data for exponential operation.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	(d)	Head device for storing the operation result.	FLOAT(Single Precision)

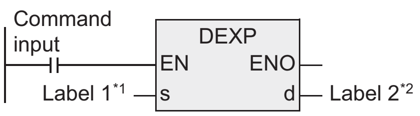
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	●	●			●			●		
(d)														●	●	●			●					

Function and operation explanation

1. 32-bit operation (DEXP/DEXPP)

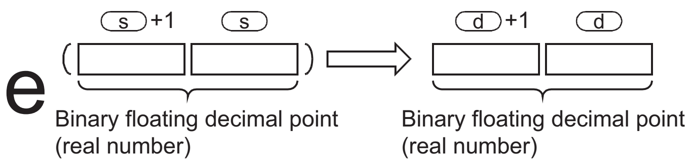
By exponential operation of the device specified by (s), the operation result is stored in the device specified by (d). Real number can be directly specified in the device specified by (s).



*1. To define the head device for storing the binary floating decimal point data for exponential operation.

*2. To define the head device for storing the operation result.

- In exponential operation, the bottom (e) is supposed to be 2.71828.



Error

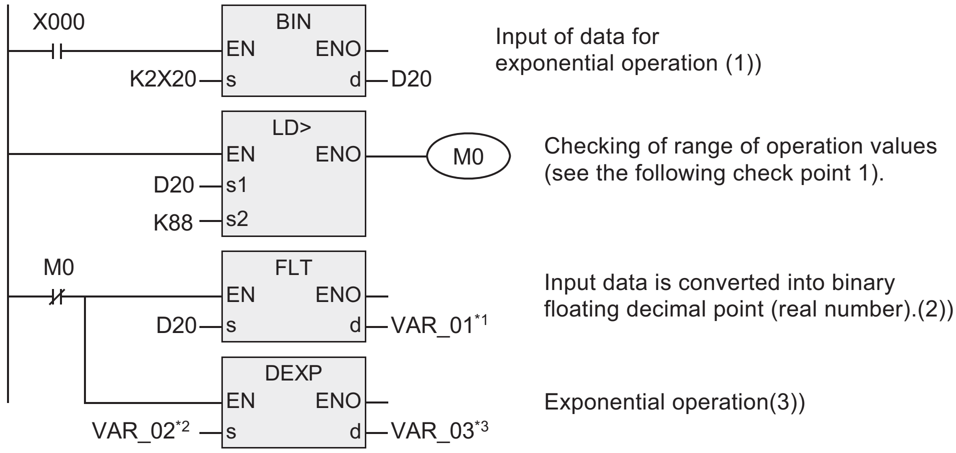
In the following cases, it is an operation error, error flag (M8067) is turned ON, and error code is stored in D8067.

- When the operation result is out of the following range. (Error code: K6706)
  $2^{-126} \leq |\text{Operation result}| < 2^{128}$

Program example

This is a program for executing exponential operation of the values set in BCD two digits in X020 to X027 when the X000 is turned ON, and storing in binary floating decimal points D0, D1.

[Structured ladder]

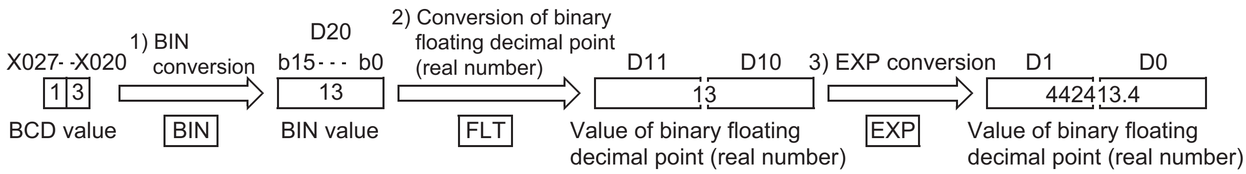


*1. VAR_01 is global label, and D10 is defined.
*2. VAR_02 is global label, and D10 is defined.
*3. VAR_03 is global label, and D0 is defined.

[ST]

```
BIN(X060,K2X20,D20);
M0:=LD>(TRUE,D20,K88);
FLT(NOT M0,D20,VAR_01);
DEXP(NOT M0,VAR_02,VAR_03);
```

Operation when 13 is specified in X020 to X027.



Points

1) The operation result is less than 2^{128} only when the BCD value of X020 to X027 is smaller than 88, because log e 2^{128} = 88.7. When a value greater than 89 is specified, it is an operation error, and therefore when a value greater than 89 is specified, M0 is turned ON, so that the operation is not carried out.
2) Conversion from natural logarithm into common logarithm
   The CPU operates in natural logarithm.
   To determine value in common logarithm, please specify the value of common logarithm divided by 0.4342945 in (s)+1, (s).

$$10^{X} = e^{\frac{X}{0.4342945}}$$

Caution

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

7.12.13 DLOGE

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction executes natural logarithm operation.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DLOGE	32 bits	Continuous	DLOGE EN ENO s d	DLOGE(EN, s, d);
DLOGEP	32 bits	Pulse	DLOGEP EN ENO s d	DLOGEP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device for storing the binary floating decimal point data for natural logarithm operation.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	(d)	Head device for storing the operation result.	FLOAT(Single Precision)

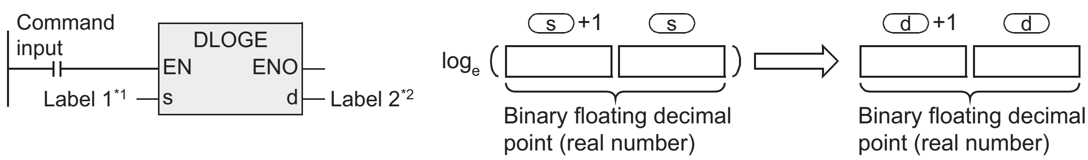
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	●	●			●			●		
(d)														●	●	●			●					

Function and operation explanation

1. 32-bit operation (DLOGE/DLOGEP)

By natural logarithm operation of the device specified by (s) (logarithm supposing e (2.71828) to be the bottom), the operation result is stored in the device specified by (d). Real number can be directly specified in the device specified by (s).



*1. To define the head device for storing the binary floating decimal point data for natural logarithm operation.

*2. To define the head device for storing the operation result.

- The value to be specified by (s) can be set in positive number only. (Negative number cannot be operated.)

Caution

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

Error

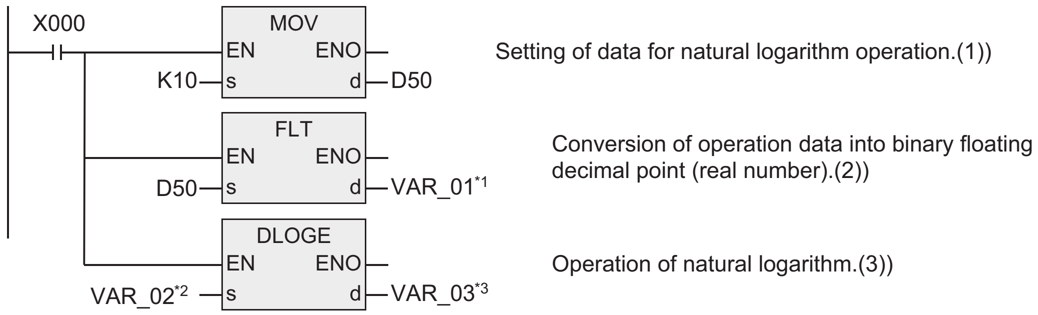
In the following cases, it is an operation error, error flag (M8067) is turned ON, and error code is stored in D8067.

- When the value specified by (s) is negative. (Error code: K6706)
- When the value specified by (s) is 0. (Error code: K6706)

Program example

This is a program for determining the natural logarithm of "10" set in D50 when the X000 is turned ON, and storing in D30, D31.

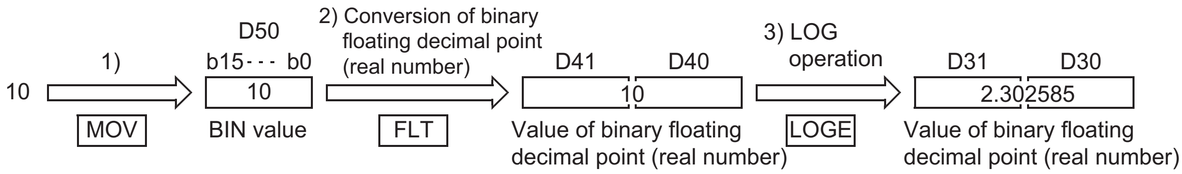
[Structured ladder]



*1. VAR_01 is global label, and D40 is defined.
*2. VAR_02 is global label, and D40 is defined.
*3. VAR_03 is global label, and D30 is defined.

[ST]

```
MOV(X000,K10,D50);
FLT:=(X000,D50,VAR_01);
DLOGE(X000,VAR_02,VAR_03);
```



7.12.14 DLOG10

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction executes common logarithm operation.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DLOG10	32 bits	Continuous	DLOG10 (EN, ENO, s, d)	DLOG10(EN, s ,d);
DLOG10P	32 bits	Pulse	DLOG10P (EN, ENO, s, d)	DLOG10P(EN ,s ,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device for storing the binary floating decimal point data for common logarithm operation.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	(d)	Head device for storing the operation result.	FLOAT(Single Precision)

3. Applicable devices

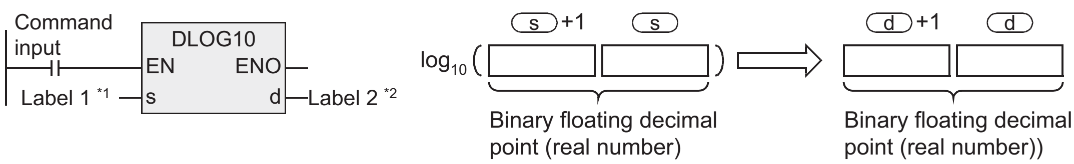
Operand type	Bit Devices							Word Devices													Others					
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	●	●			●			●		
(d)														●	●	●			●					

Function and operation explanation

1. 32-bit operation (DLOG10/DLOG10P)

By common logarithm operation of the device specified by (s) (logarithm supposing 10 to be the bottom), the operation result is stored in the device specified by (d).

Real number can be directly specified in the device specified by (s).



*1. To define the head device for storing the binary floating decimal point data for common logarithm operation.

*2. To define the head device for storing the operation result.

- The value to be specified by (s) can be set in positive number only. (Negative number cannot be operated.)

Caution

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

Error

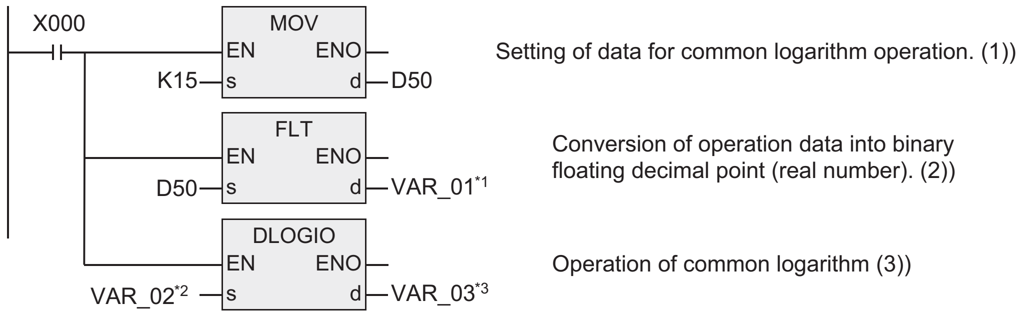
In the following cases, it is an operation error, error flag (M8067) is turned ON, and error code is stored in D8067.

- When the value specified by (s) is negative. (Error code: K6706)
- When the value specified by (s) is 0. (Error code: K6706)

Program example

This is a program for determining the common logarithm of "15" set in D50 when the X000 is ON, and storing in D30, D31.

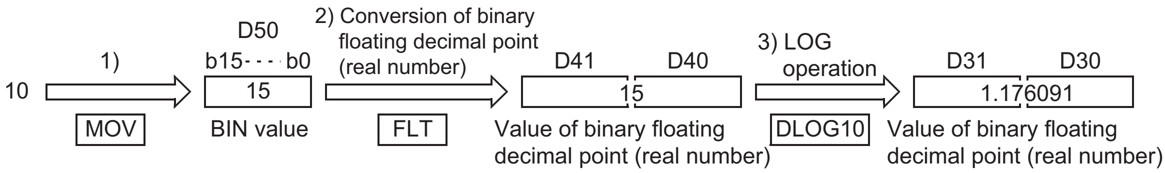
[Structured ladder]



*1. VAR_01 is global label, and D40 is defined.
*2. VAR_02 is global label, and D40 is defined.
*3. VAR_03 is global label, and D30 is defined.

[ST]

```
MOV(X000,K15,D50);
FLT:=(X000,D50,VAR_01);
DLOGIO(X000,VAR_02,VAR_03);
```



7.12.15 DESQR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	○	×	×	×	×	×

Outline

This instruction executes the square root operation of binary floating decimal point.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DESQR	32 bits	Continuous	DESQR (EN, ENO, s, d)	DESQR(EN, s, d);
DESQRP	32 bits	Pulse	DESQRP (EN, ENO, s, d)	DESQRP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Device for storing binary floating decimal point data for square root operation	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	(d)	Device for storing binary floating decimal point data after square root operation	FLOAT(Single Precision)

3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	▲1	▲2			●	●	●	▲1		
(d)														●	▲1	▲2			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DESQR, DESQRP)

The data in the device specified by (s) is operated by square root (binary floating decimal point), and the result is transferred to the device specified by (d).

$\sqrt{[(s)+1, (s)]}$ → [(d)+1, (d)]
Binary floating decimal point → Binary floating decimal point

*1. To define the device for storing binary floating decimal point data for square root operation

*2. To define the device for storing binary floating decimal point data after operation of square root

Related devices

→ As for the manner of using the zero flag, refer to the FX Structured Programming Manual (Device & Common).

Device	Name	Content
M8020	Zero	To be turned ON when the operation result is true 0.

Cautions

1) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
2) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
3) Applicable devices are limited.
   ▲1: FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

Error

The content of the device specified by (s1) is valid only in positive number, and negative number leads to operation error (M8067), and the instruction is not executed.

7.12.16 DENEG

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction inverts the sign of binary floating decimal point (real number) data.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DENEG	32 bits	Continuous	DENEG: EN, ENO, d	DENEG(EN, d);
DENEGP	32 bits	Pulse	DENEGP: EN, ENO, d	DENEGP(EN, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	d	Head device for storing the binary floating decimal point data of which sign is to be inverted.	FLOAT(Single Precision)

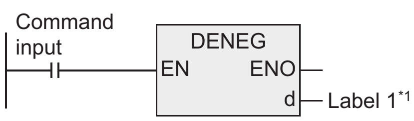
3. Applicable devices

Operand type	Bit Devices							Word Devices													Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
d														●	●	●			●					

Function and operation explanation

1. 32-bit operation (DENEG/DENEGP)

The sign of the binary floating decimal point data of the device specified by d is inverted, and stored in the device specified by d.



*1. To define the head device for storing the binary floating decimal point data of which sign is to be inverted.

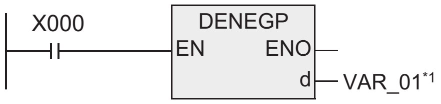
Caution

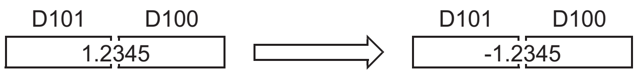
When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

Program example

This is a program for inverting the sign of the binary floating decimal point data of D100, D101 when the X000 is turned ON, and storing in D100, D101.

[Structured ladder]





*1. VAR_01 is global label, and D100 is defined.

[ST]

```
DENEGP(X000,VAR_01);
```

7.12.17 INT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	○	×	×	×	×	×

Outline

This instruction converts the binary floating decimal point into BIN integer in normal data type in the PLC.
(From binary floating decimal point data to BIN integer)

→ **As for program example of floating decimal point operation, refer to section 7.5.10.**
→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
INT	16 bits	Continuous	INT EN ENO s d	INT(EN, s, d);
INTP	16 bits	Pulse	INTP EN ENO s d	INTP(EN, s, d);
DINT	32 bits	Continuous	DINT EN ENO s d	DINT(EN, s, d);
DINTP	32 bits	Pulse	DINTP EN ENO s d	DINTP(EN, s, d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Device for storing the binary floating decimal point data to be converted into BIN integer.	FLOAT(Single Precision)	
Output variable	ENO	Execution state	Bit	
	d	Device for storing the converted BIN integer.	ANY16	ANY32

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	▲1	▲2			●					
d														●	▲1	▲2			●					

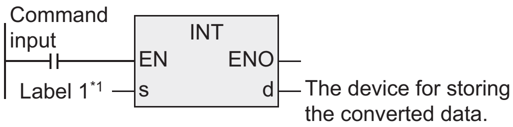
▲: Refer to "Cautions".

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

1. 16-bit operation (INT, INTP)

The binary floating decimal point of the device specified by (s) is converted into BIN integer, and is transferred to the device specified by (d).



(s)+1, (s) → (d)

Binary floating decimal point → To discard below the decimal point of 16-bit BIN integer.

*1. To define the device for storing the data to be converted into BIN integer.

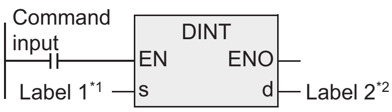
Instruction of reverse converting operation

Reverse converting operation of the operation of this instruction is FLT.

→ **As for FLT instruction, refer to 7.5.10.**

2. 32-bit operation (DINT, DINTP)

The binary floating decimal point of the device specified by (s) is converted into BIN integer, and is transferred to the device specified by (d).



(s)+1, (s) → (d)+1, (d)

Binary floating decimal point → To discard below the decimal point of 32-bit BIN integer.

*1. To define the device for storing the data to be converted into BIN integer.
*2. To define the device for storing the converted data.

Instruction of reverse converting operation

Reverse converting operation of the operation of this instruction is DFLT(FNC49) instruction.

→ **As for FLT instruction, refer to 7.5.10.**

Related devices

→ **As for the manner of using the zero, borrow, and carry flag, refer to the FX Structured Programming Manual (Device & Common).**

Device	Name	Content
M8020	Zero	To be turned ON when the operation result is 0.
M8021	Borrow	To be turned ON when less than 1 is discarded in convertion.
M8022	Carry	To be turned ON when the operation result overflows by exceeding the range of -32, 768 to 32, 767 (in 16-bit operation), or -2, 147, 483, 648 to 2, 147, 483, 647 (in 32-bit operation). (Operation result is not reflected.)

Cautions

1) Fractions below the decimal point are discarded in operation.
2) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
3) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
4) Applicable devices are limited.
   ▲1: FX3U, FX3UC and FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

7.12.18 DSIN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	×	×	×

Outline

This instruction determines the SIN value of angle (RAD).

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DSIN	32 bits	Continuous	DSIN (EN, ENO, s, d)	DSIN(EN, s, d);
DSINP	32 bits	Pulse	DSINP (EN, ENO, s, d)	DSINP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Device for storing the RAD (angle) of binary floating decimal point.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Device for storing the SIN value of binary floating decimal point.	FLOAT(Single Precision)

3. Applicable devices

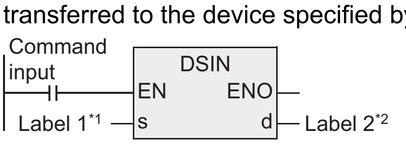
Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	▲1	▲1			●			▲1		
d														●	▲1	▲1			●					

▲: Refer to "Cautions".

Function and operation explanation

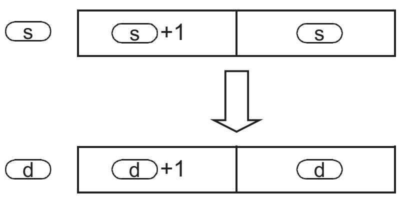
1. 32-bit operation (DINT, DINTP)

The value of angle (binary floating decimal point) specified by (s) is converted into SIN value, and is transferred to the device specified by (d).



[(s)+1, (s)]RAD → [(d)+1, (d)]SIN

Binary floating decimal point value → Binary floating decimal point value



RAD value (angle × π/180)
Binary floating decimal point value is specified.

SIN value
Binary floating decimal point value

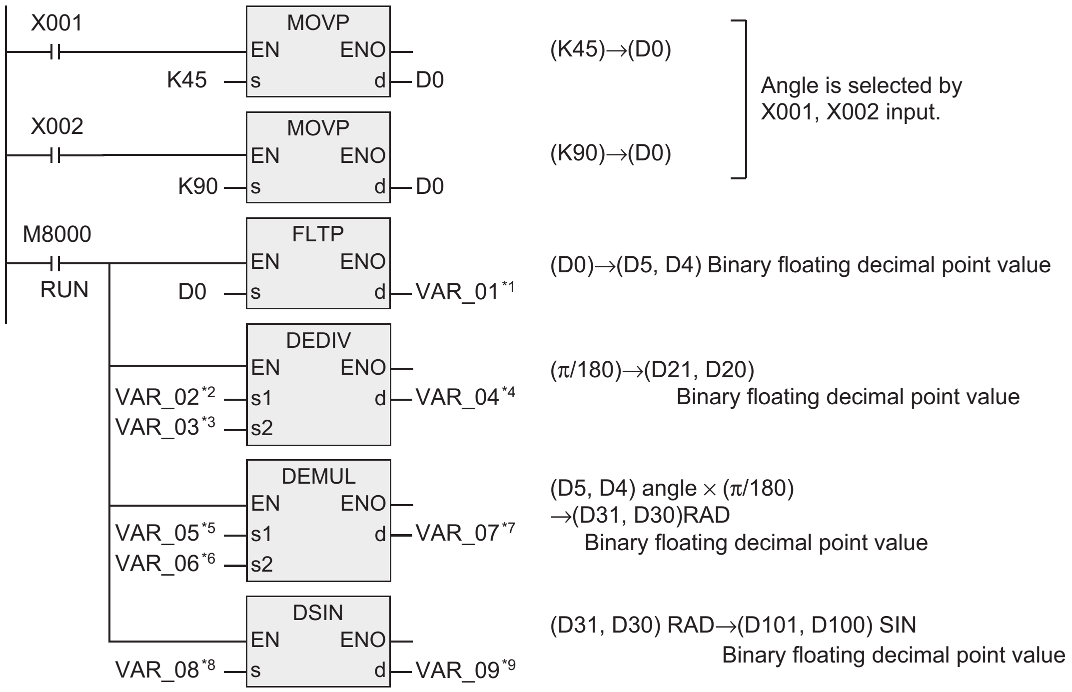
*1. To define the device for storing RAD (angle).
*2. To define the device for storing SIN value.

Cautions

1) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
2) Applicable devices are limited.
   ▲: FX3U, FX3UC PLCs only are applicable.

Program example

[Structured ladder]



*1. VAR_01 is global label, and D4 is defined.
*2. VAR_02 is global label, and K31415926 is defined.
*3. VAR_03 is global label, and K1800000000 is defined.
*4. VAR_04 is global label, and D20 is defined.
*5. VAR_05 is global label, and D4 is defined.
*6. VAR_06 is global label, and D20 is defined.
*7. VAR_07 is global label, and D30 is defined.
*8. VAR_08 is global label, and D30 is defined.
*9. VAR_09 is global label, and D100 is defined.

[ST]

```
MOVP(X001,K45,D0);
MOVP(X002,K90,D0);
FLTP(M8000,D0,VAR_01);
DEDIV(M8000,VAR_02,VAR_03,VAR_04);
DEMUL(M8000,VAR_05,VAR_06,VAR_07);
DSIN(M8000,VAR_08,VAR_09);
```

7.12.19 DCOS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	×	×	×

Outline

This instruction determines the COS value of angle (RAD).

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DCOS	32 bits	Continuous	DCOS (EN, ENO, s, d)	DCOS(EN, s, d);
DCOSP	32 bits	Pulse	DCOSP (EN, ENO, s, d)	DCOSP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Device for storing the RAD (angle) of binary floating decimal point.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Device for storing the COS value of binary floating decimal point.	FLOAT(Single Precision)

3. Applicable devices

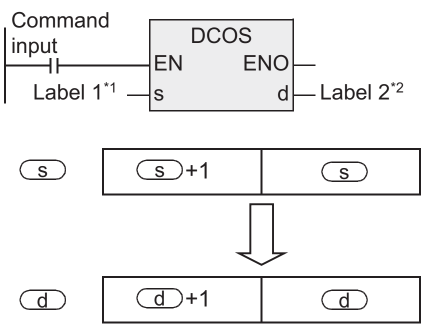
Operand type	Bit Devices: System user							Word Devices: Digit designation				System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	▲1	▲1			●			▲1		
d														●	▲1	▲1			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DCOS, DCOSP)

The value of angle (binary floating decimal point) specified by s is converted into COS value, and is transferred to the device specified by d.



[s +1, s]RAD → [d +1, d]COS
Binary floating decimal point value → Binary floating decimal point value

s: RAD value (angle × π/180)
Binary floating decimal point value is specified.

d: COS value
Binary floating decimal point value

*1. To define the device for storing RAD (angle).
*2. To define the device for storing the COS value.

Cautions

1) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
2) Applicable devices are limited.
   ▲: FX3U, FX3UC PLCs only are applicable.

7.12.20 DTAN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	×	×	×

Outline

This instruction determines the TAN value of angle (RAD).

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DTAN	32 bits	Continuous	DTAN (EN, ENO, s, d)	DTAN(EN, s, d);
DTANP	32 bits	Pulse	DTANP (EN, ENO, s, d)	DTANP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Device for storing the RAD (angle) of binary floating decimal point.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Device for storing the TAN value of binary floating decimal point.	FLOAT(Single Precision)

3. Applicable devices

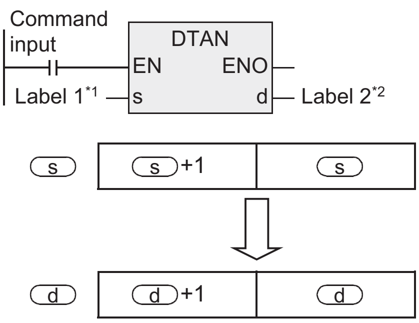
Operand type	Bit Devices: System user							Word Devices: Digit designation				System user			Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	▲1	▲1			●			▲1		
d														●	▲1	▲1			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DTAN, DTANP)

The value of angle (binary floating decimal point) specified by s is converted into TAN value, and is transferred to the device specified by d.



[s+1, s]RAD → [d+1, d]TAN
Binary floating decimal point value → Binary floating decimal point value

RAD value (angle × π/180)
Binary floating decimal point value is specified.

TAN value
Binary floating decimal point value

*1. To define the device for storing RAD (angle).
*2. To define the device for storing the TAN value.

Cautions

1) When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
2) Applicable devices are limited.
   ▲1: FX3U, FX3UC PLCs only are applicable.

7.12.21 DASIN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction executes SIN^{-1} operation.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DASIN	32 bits	Continuous	DASIN EN ENO s d	DASIN(EN, s, d);
DASINP	32 bits	Pulse	DASINP EN ENO s d	DASINP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device for storing the SIN value for SIN^{-1} (reverse sine) operation.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	(d)	Head device for storing the operation result.	FLOAT(Single Precision)

3. Applicable devices

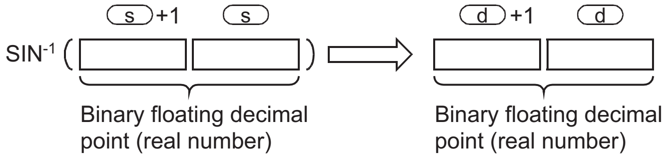
Operand type	Bit Devices: System user							Word Devices: Digit designation				System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	●	●			●			●		
(d)														●	●	●			●					

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

1. 32-bit operation (DASIN/DASINP)

The angle is determined from the SIN value specified by (s), and the operation result is stored in the device specified by (d). In the device specified by (s), real number can be directly specified.



*1. To define the head device for storing the SIN value for SIN^{-1} operation.
*2. To define the device for storing the operation result.

- The SIN value specified by (s) can be set in a range of -1.0 to 1.0.
- The angle (operation result) to be stored in the device specified by (d) stores the value of radian ($-\pi/2$ to $\pi/2$). As for conversion from radian to angle or vice versa, refer to DRAD instruction or DDEG instruction.

→ **As for DRAD instruction, refer to section 7.12.24.**
→ **As for DDEG instruction, refer to section 7.12.25.**

Caution

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

Error

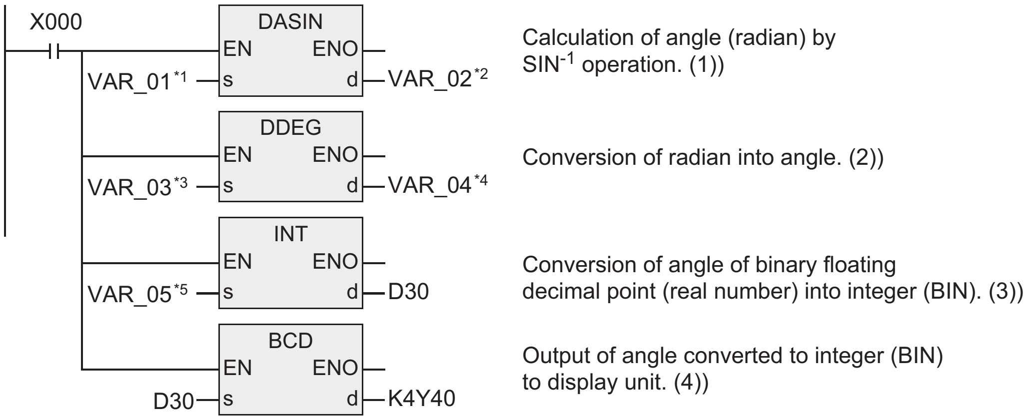
In the following cases, it is an operation error, error flag (M8067) is turned ON, and error code is stored in D8067.

- When the value specified by (s) is out of the range of -1.0 to 1.0. (Error code: K6706)

Program example

This is a program for determining SIN-1 of D0, D1 (binary floating decimal point) when the X000 is ON, and sending the angle to Y040 to Y057 in BCD four digits.

[Structured ladder]

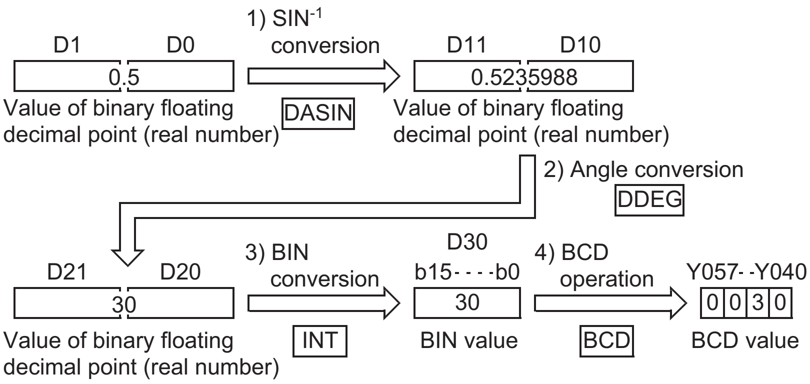


*1. VAR_01 is global label, and D0 is defined.
*2. VAR_02 is global label, and D10 is defined.
*3. VAR_03 is global label, and D10 is defined.
*4. VAR_04 is global label, and D20 is defined.
*5. VAR_05 is global label, and D20 is defined.

[ST]

```
DASIN(X000,VAR_01,VAR_02);
DDEG:=(X000,VAR_03,VAR_04);
INT(X000,VAR_05,D30);
BCD(X000,D30,K4Y10);
```

Operation when the value of D0, D1 is 0.5.



7.12.22 DACOS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction executes COS^{-1} operation.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DACOS	32 bits	Continuous	DACOS EN ENO s d	DACOS(EN, s, d);
DACOSP	32 bits	Pulse	DACOSP EN ENO s d	DACOSP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device for storing the COS value for COS^{-1} (reverse cosine) operation.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Head device for storing the operation result.	FLOAT(Single Precision)

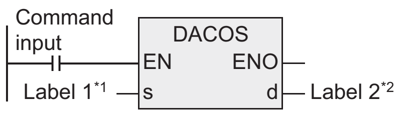
3. Applicable devices

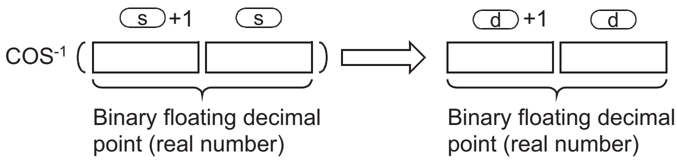
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	●	●			●			●		
d														●	●	●			●					

Function and operation explanation

1. 32-bit operation (DACOS/DACOSP)

The angle is calculated from the COS value specified by (s), and the operation result is stored in the device specified by (d). Real number can be directly specified by the device specified by (s).





*1. To define the head device for storing the COS value for COS^{-1} operation.
*2. To define the device for storing the operation result.

- The COS value specified by (s) can be set in a range of -1.0 to 1.0.
- The angle (operation result) to be stored in the device specified by (d) stores the value 0 to π in the radian unit.
  As for conversion from radian to angle or vice versa, refer to DRAD instruction or DDEG instruction.

→ **As for DRAD instruction, refer to section 7.12.24.**
→ **As for DDEG instruction, refer to section 7.12.25.**

Caution

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

Error

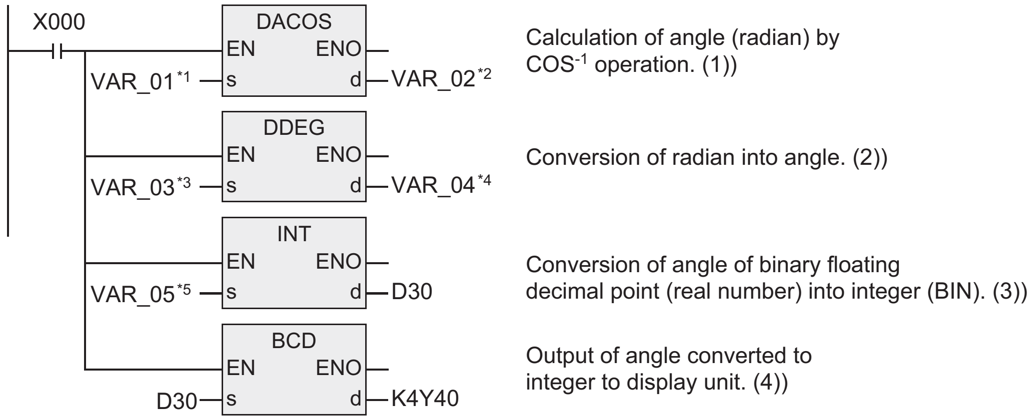
In the following cases, it is an operation error, error flag (M8067) is turned ON, and error code is stored in D8067.

- When the value specified by (s) is out of the range of -1.0 to 1.0. (Error code: K6706)

Program example

This is a program for determining COS^{-1} of D0, D1 (binary floating decimal point) when the X000 is ON, and sending the angle to Y040 to Y057 in BCD four digits.

[Structured ladder]

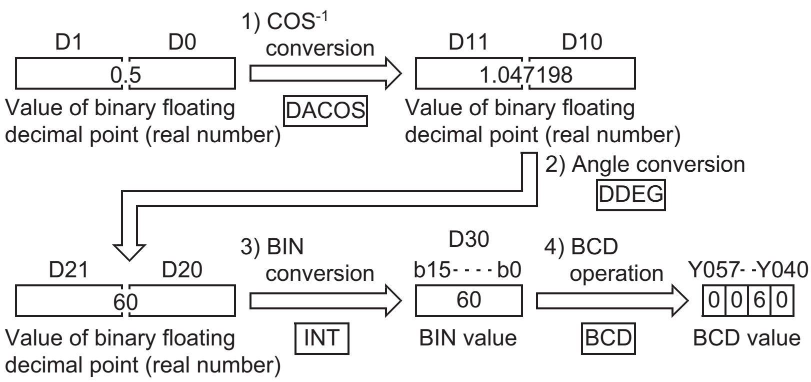


*1. VAR_01 is global label, and D0 is defined.
*2. VAR_02 is global label, and D10 is defined.
*3. VAR_03 is global label, and D10 is defined.
*4. VAR_04 is global label, and D20 is defined.
*5. VAR_05 is global label, and D20 is defined.

[ST]

```
DACOS(X000,VAR_01,VAR_02);
DDEG:=(X000,VAR_03,VAR_04);
INT(X000,VAR_05,D30);
BCD(X000,D30,K4Y10);
```

Operation when the value of D0, D1 is 0.5.



7.12.23 DATAN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction executes TAN^{-1} operation.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DATAN	32 bits	Continuous	DATAN EN ENO s d	DATAN(EN, s, d);
DATANP	32 bits	Pulse	DATANP EN ENO s d	DATANP(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device for storing the TAN value for TAN^{-1} operation (reverse tangent).	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Head device for storing the operation result.	FLOAT(Single Precision)

3. Applicable devices

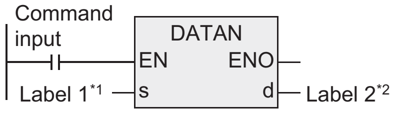
Operand type	Bit Devices							Word Devices													Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	●	●			●			●		
d														●	●	●			●					

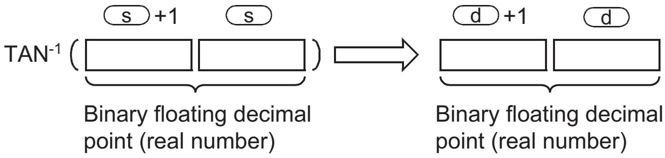
1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

1. 32-bit operation (DATAN/DATANP)

The angle is calculated from the TAN value specified by (s), and the operation result is stored in the device specified by (d). Real number can be directly specified by the device specified by (s).





*1. To define the head device for storing the TAN value for TAN^{-1} operation.

*2. To define the device for storing the operation result.

- The angle (operation result) to be stored in the device specified by (d) stores the value larger than ($-\pi/2$) and smaller than ($\pi/2$) in the radian unit.
  As for conversion from radian to angle or vice versa, refer to DRAD instruction or DDEG instruction.

→ **As for DRAD instruction, refer to section 7.12.24.**
→ **As for DDEG instruction, refer to section 7.12.25.**

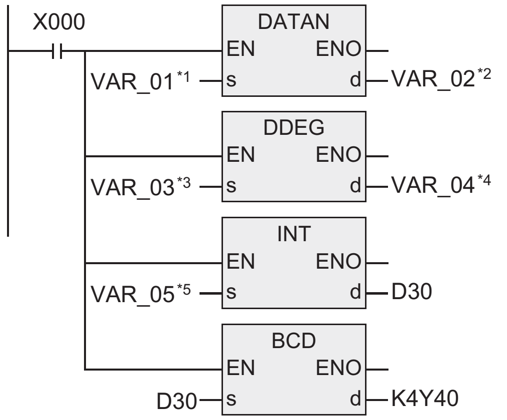
Caution

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

Program example

This is a program for determining TAN^{-1} of D0, D1 (binary floating decimal point) when the X000 is ON, and sending the angle to Y040 to Y057 in BCD four digits.

[Structured ladder]



Calculation of angle (radian) by TAN^{-1} operation. (1))

Conversion of radian into angle. (2))

Conversion of angle of binary floating decimal point (real number) into integer (BIN). (3))

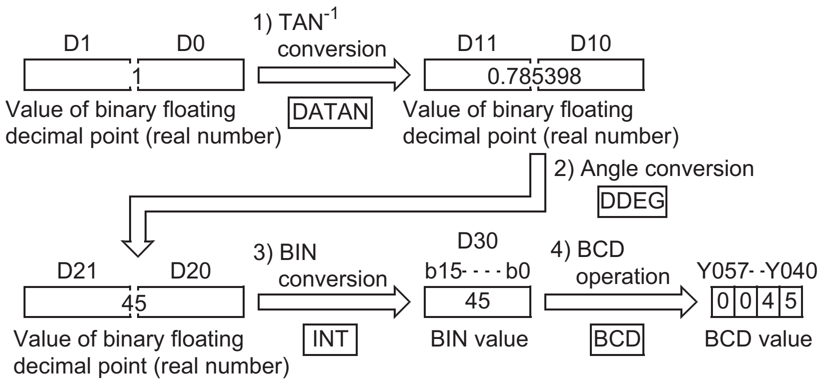
Output of angle converted to integer (BIN) to display unit. (4))

*1. VAR_01 is global label, and D0 is defined.
*2. VAR_02 is global label, and D10 is defined.
*3. VAR_03 is global label, and D10 is defined.
*4. VAR_04 is global label, and D20 is defined.
*5. VAR_05 is global label, and D20 is defined.

[ST]

```
DATAN(X000,VAR_01,VAR_02);
DDEG:=(X000,VAR_03,VAR_04);
INT(X000,VAR_05,D30);
BCD(X000,D30,K4Y10);
```

Operation when the value of D0, D1 is 1.



7.12.24 DRAD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction converts the value of angle unit to the radian unit.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DRAD	32 bits	Continuous	DRAD EN ENO s d	DRAD(EN, s, d);
DRADP	32 bits	Pulse	DRAD_P EN ENO s d	DRAD_P(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device for storing the angle to be converted to radian unit.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	d	Head device for storing the value having been converted to radian unit	FLOAT(Single Precision)

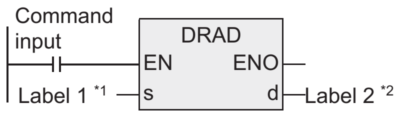
3. Applicable devices

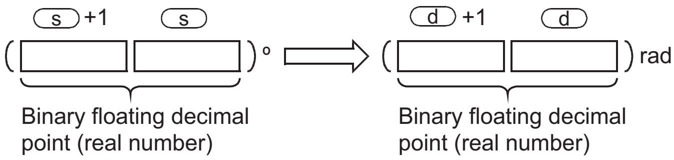
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	●	●			●			●		
d														●	●	●			●					

Function and operation explanation

1. 32-bit operation (DRAD/DRADP)

The angle specified by (s) is converted from the degree unit to the radian unit, and is stored in the device specified by (d). Real number can be directly specified in the device specified by (s).





*1. To define the head device for storing the angle to be converted to the radian unit.

*2. To define the head device for storing the value having been converted to the radian unit.

- Conversion from degree unit to radian unit is as follows.

$$\text{Radian unit} = \text{degree unit} \times \frac{\pi}{180}$$

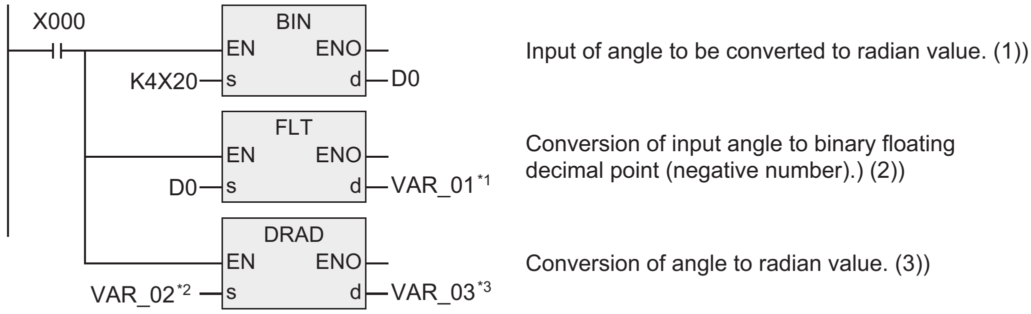
Caution

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

Program example

This is a program for converting the angle set in BCD four digits in X020 to X037 when the X000 is ON, and storing in binary floating decimal point in D20, D21.

[Structured ladder]

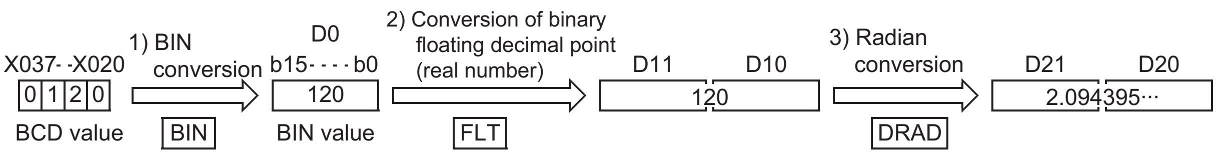


*1. VAR_01 is global label, and D10 is defined.
*2. VAR_02 is global label, and D10 is defined.
*3. VAR_03 is global label, and D20 is defined.

[ST]

```
BIN(X000,K4X20,D0);
FLT:=(X000,D0,VAR_01);
DRAD(X000,VAR_02,VAR_03);
```

Operation when 120 is specified in X020 to X037.



7.12.25 DDEG

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction converts the radian unit value into the angle (DEG) unit.

→ **As for handling of floating decimal point, refer to FX Structured Programming Manual (Device & Common).**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DDEG	32 bits	Continuous	DDEG (EN / ENO, s / d)	DDEG(EN, s, d);
DDEGP	32 bits	Pulse	DDEG_P (EN / ENO, s / d)	DDEG_P(EN, s, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device for storing the radian angle to be converted to degree unit.	FLOAT(Single Precision)
Output variable	ENO	Execution state	Bit
	(d)	Head device for storing the value having been converted to degree unit.	FLOAT(Single Precision)

3. Applicable devices

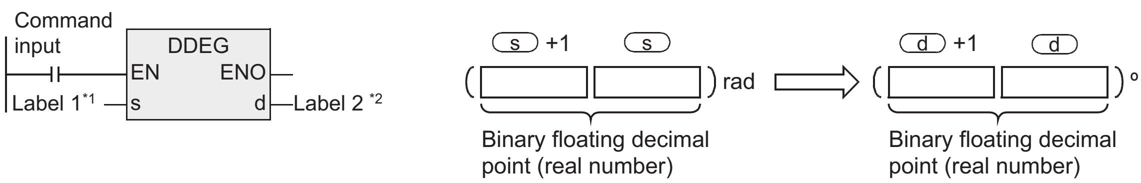
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	●	●			●			●		
(d)														●	●	●			●					

Function and operation explanation

1. 32-bit operation (DDEG/DDEGP)

(DDEG/DDEGP)

The unit of the angle specified by (s) is converted from the radian unit to the degree unit, and is stored in the device specified by (d).



*1. To define the head device for storing the radian angle to be converted to degree unit.

*2. To define the head device for storing the value having been converted to degree unit.

- Conversion from radian unit to degree unit is as follows.

$$\text{Degree unit} = \text{radian unit} \times \frac{180}{\pi}$$

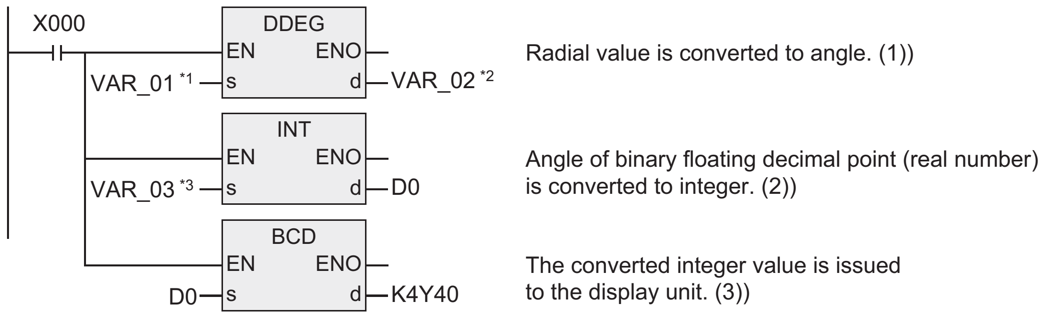
Caution

When handling 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and can be specified directly.
When specifying the device, use the global label.

Program example

This is a program for converting the radian value set in binary floating decimal point in D20, D21 to the angle when the X000 is ON, and issuing to the Y040 to Y057 in BCD value.

[Structured ladder]

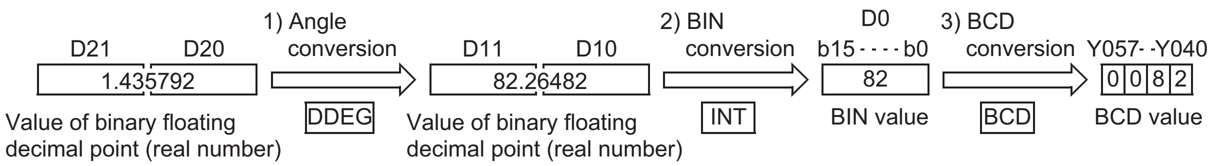


*1. VAR_01 is global label, and D20 is defined.
*2. VAR_02 is global label, and D10 is defined.
*3. VAR_03 is global label, and D10 is defined.

[ST]

```
DDEG(X000,VAR_01,VAR_02);
INT:=(X000,VAR_03,D0);
BCD(X000,D0,K4Y40);
```

Operation when the value of D20, D21 is 1.435792.



7.13 Data Operation 2

7.13.1 WSUM

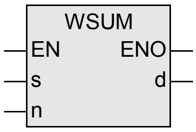
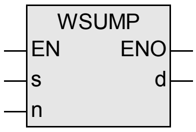
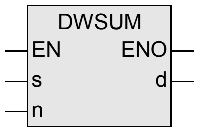
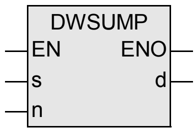
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction calculates the total value of continuous 16-bit data or 32-bit data.
Please use the CCD when calculating the sum data (total value) in byte (8-bit) unit.

→ **As for the CCD, refer to section 7.9.5.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
WSUM	16 bits	Continuous		WSUM(EN, s, n, d);
WSUMP	16 bits	Pulse		WSUMP(EN, s, n, d);
DWSUM	32 bits	Continuous		DWSUM(EN, s, n, d);
DWSUMP	32 bits	Pulse		DWSUMP(EN, s, n, d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Head device for storing the data for calculating the total value (n points occupied).	ANY16	ANY32
	n	Number of data (0<n)	ANY16	
Output variable	ENO	Execution state	Bit	
	d	Head device for storing the total value.	ANY32	ARRAY [1..4] OF ANY16

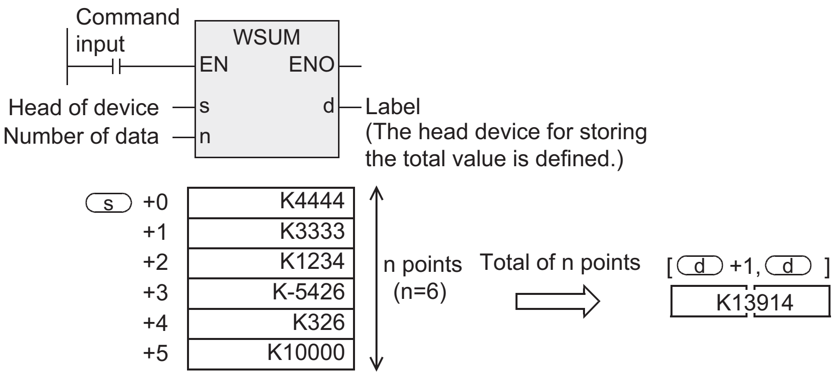
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)												●	●	●	●	●			●					
(d)												●	●	●	●	●			●					
(n)														●	●					●	●			

Function and operation explanation

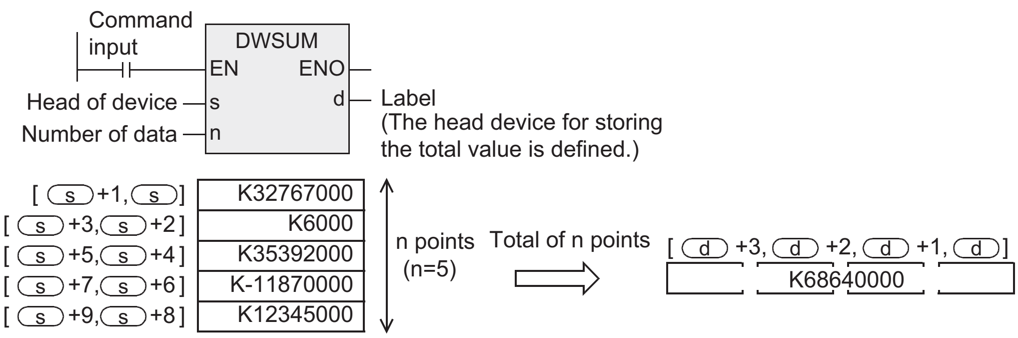
1. 16-bit operation (WSUM/WSUMP)

The total value of n points of 16-bit data from the device specified by (s) is stored in the device specified by (d) as 32-bit data.



2. 32-bit operation (DWSUM/DWSUMP)

The total value of n points of 32-bit data from the device specified by (s) is stored in the device specified by (d) as 64-bit data.



Related instructions

Instruction	Content
CCD	Check code This instruction calculates the total value and the horizontal parity of 16-bit data in byte (8-bit) unit.

Cautions

1) In 32-bit operation, the total value becomes 64-bit data. FX3U, FX3UC PLCs cannot handle 64-bit data. However, when the total value is in the numeric value range of 32-bit data (K-2,147,483,648 to K2,147,483,647), higher 32-bit data is ignored, and the lower 32-bit data can be handled as the total value.
2) When handling array data or 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project. When handling array data or 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and can be specified directly.
   When specifying the device, use the global label.
3) FX3U, FX3UC PLCs support the instruction at V2.20 or later.

Error

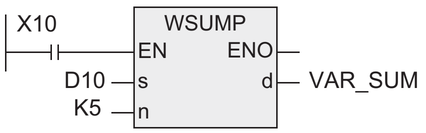
In the following case, it is an operation error, error flag M8067 is turned ON, and error code is stored in D8067.

1) When n devices from the device specified by (s) exceed the device range. (Error code: K6706)
2) When n ≤ 0. (Error code: K6706)
3) When the device specified by (d) exceeds the range. (Error code: K6706)

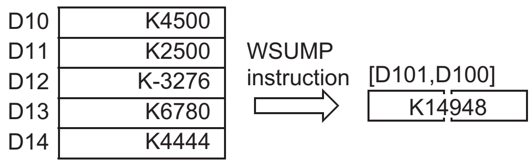
Program example

This is a program for storing the total value of 16-bit data of D10 to D14 when the X010 is turned ON to [D101, D100].

[Structured ladder]



VAR_SUM is a global label, and D100 is defined.



[ST]

```
WSUMP(X10,D10,K5,VAR_SUM);
```

7.13.2 WTOB

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction separates continuous 16-bit data in byte (8-bit) unit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
WTOB	16 bits	Continuous	WTOB EN ENO s d n	WTOB(EN, s, n, d);
WTOBP	16 bits	Pulse	WTOBP EN ENO s d n	WTOBP(EN, s, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device for storing the data to be separated in byte unit.	ANY16
	n	Number of byte data to be separated. (0≤n)	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device for storing the result separated in byte unit.	ANY16

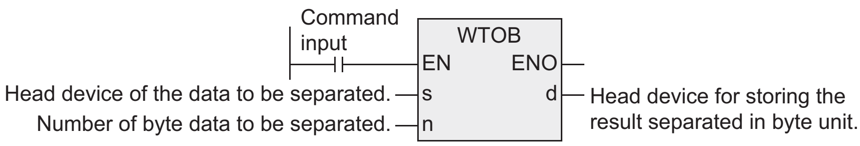
3. Applicable devices

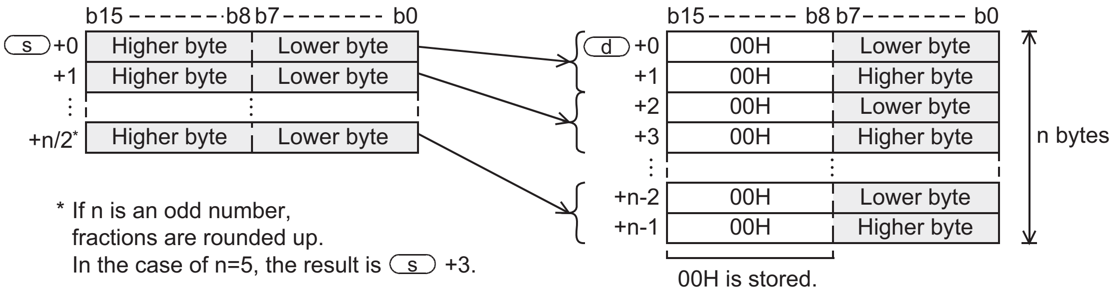
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s												●	●	●	●				●					
d												●	●	●	●				●					
n														●	●					●	●			

Function and operation explanation

1. 16-bit operation (WTOB/WTOBP)

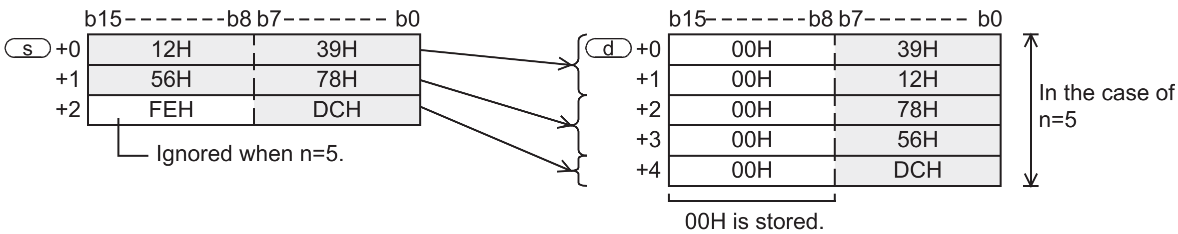
1) This instruction separates n/2 points of 16-bit data stored after the device specified by (s) into n bytes, and stores into n devices starting from the device specified by (d) as explained below.





2) 00H is stored in the higher byte (8 bits) of the device for storing the separated byte data (after the device specified by (d)).

3) In the case of n = odd number, the final data of separation source is applicable only to the data in the lower byte (8 bits) as shown below.
For example, in the case of n=5, data of lower byte (8 bits) of (s) to (s)+2 is stored in (d) to (d)+4.



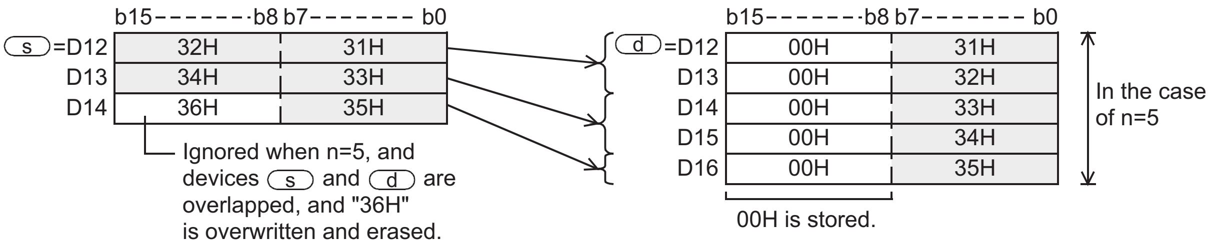
4) Instruction is not processed in the case of n=0.

Related instructions

Instruction	Content
BTOW	This instruction couples the lower 8 bits (lower byte) of continuous 16-bit data.

Cautions

1) The device storing the separation source data and the device storing the separated data can be used in overlap. However, in the case of n = odd number, it must be noted that the data of the higher byte (8 bits) of the final data of the separation source may be overwritten and erased as shown below.



2) FX3UC PLC supports at V2.20 or later.

Error

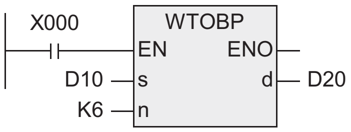
In the following case, it is an operation error, error flag M8067 is turned ON, and error code is stored in D8067.

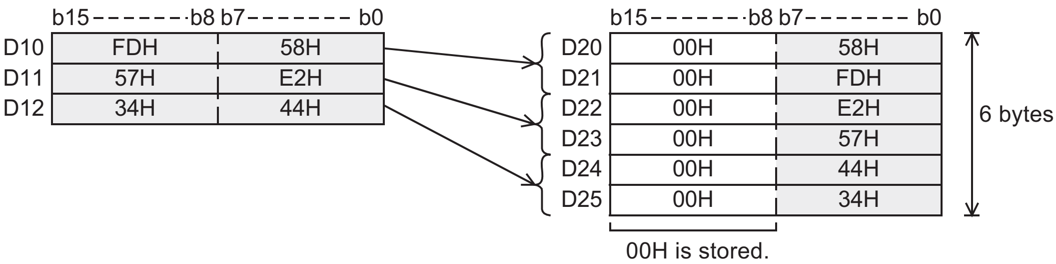
1) When devices (s) to (s) +n/2 of separation source exceed the device range of specified devices. When n is an odd number, the devices are required in the number by rounding up. (Error code: K6706)
2) When devices (d) to (d) +n-1 for storing the separated data exceed the device range of specified devices. (Error code: K6706)

Program example

This is a program for separating the data of D10 to D12 into byte unit when the X000 is turned ON, and storing in D20 to D25.

[Structured ladder]





[ST]

```
WTOBP(X000,D10,K6,D20);
```

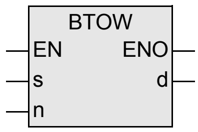
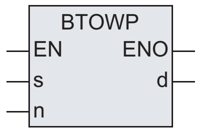
7.13.3 BTOW

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction couples the lower 8 bits (lower byte) of continuous 16-bit data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BTOW	16 bits	Continuous		BTOW(EN, s, n, d);
BTOWP	16 bits	Pulse		BTOWP(EN, s, n, d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device for storing the data to be coupled in byte unit.	ANY16
	n	Number of byte data to be coupled (0≤n)	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device for storing the coupled result in byte unit.	ANY16

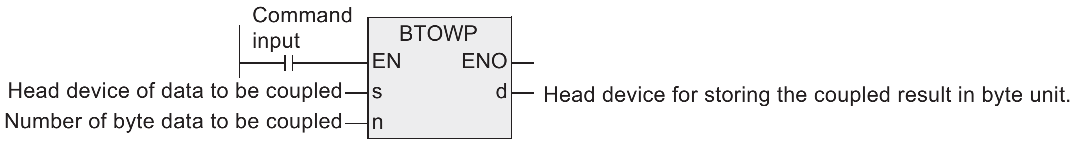
3. Applicable devices

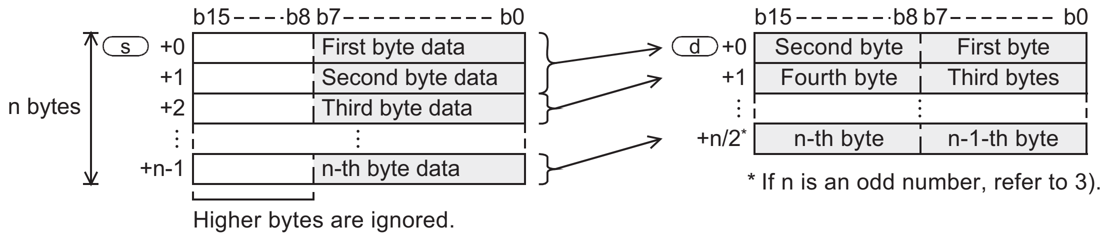
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s												●	●	●	●				●					
d												●	●	●	●				●					
n														●	●					●	●			

Function and operation explanation

1. 16-bit operation (BTOW/BTOWP)

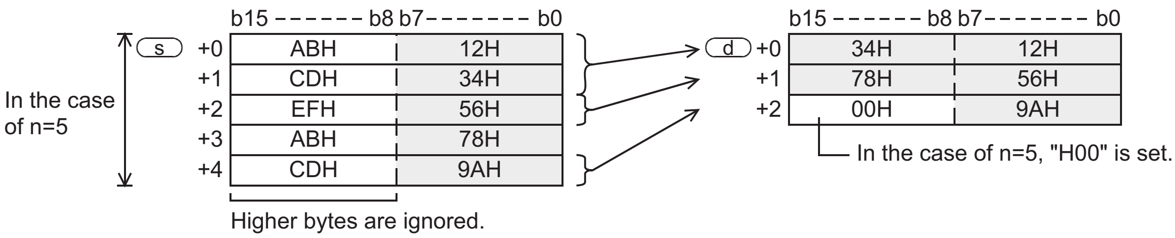
1) This instruction stores the 16-bit data coupling lower byte (8 bit) of 16-bit data of n points from the device specified by (s), into n/2 devices starting from (d) as follows.





2) Higher byte (8 bits) of 16-bit data (after (s)) in the coupling source is ignored.

3) When n is an odd number, as shown below, higher byte (8 bits) of the data coupled in the last place is set to 00H. For example, in the case of n=5, data of lower byte (8 bits) of (s) to (s) +4 is stored in (d) to (d) +2. Higher byte (8 bits) of (d) +2 is 00H.



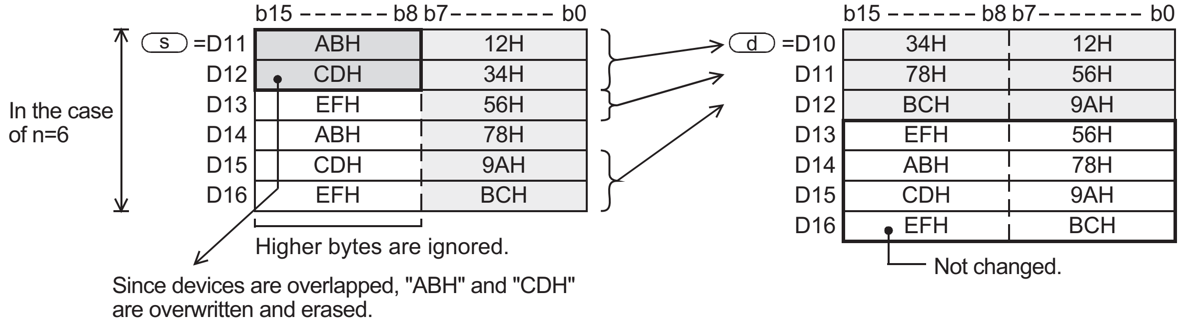
4) Instruction is not processed in the case of n=0.

Related instructions

Instruction	Content
WTOB	This instruction separates continuous 16-bit data in byte (8-bit) unit.

Cautions

1) The device storing the coupling source data and the device storing the coupled data can be used in overlap. However, it must be noted that the higher byte (8 bits) of the coupling source data stored in the device used in overlap is erased because the data of the higher byte (8 bits) is overwritten by the coupled data.



2) FX3UC PLC supports at V2.20 or later.

Error

In the following case, it is an operation error, error flag M8067 is turned ON, and error code is stored in D8067.

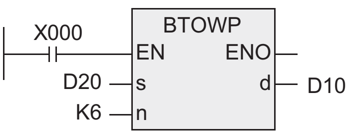
1) When the devices (s) to ((s) +n-1) at the coupling source exceeds the device range of the specified devices. (Error code: K6706)

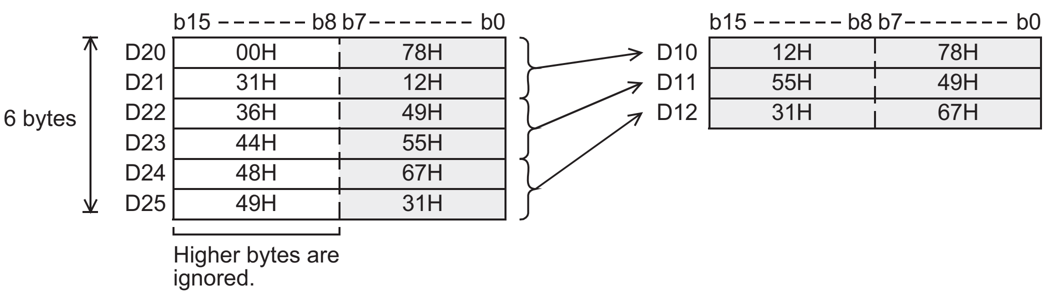
2) When the devices (d) to ((d) +n/2) for storing the coupled data exceeds the device range of the specified devices. When n is an odd number, the devices are required in the number by rounding up. (Error code: K6706)

Program example

This is a program for coupling the data of lower byte (8 bits) of D20 to D25 when the X000 is turned ON, and storing into D10 to D12.

[Structured ladder]





[ST]

```
BTOWP(X000,D20,K6,D10);
```

7.13.4 UNI

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction couples lower 4 bits of continuous 16-bit data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
UNI	16 bits	Continuous	UNI EN ENO s d n	UNI(EN,s,n,d);
UNIP	16 bits	Pulse	UNIP EN ENO s d n	UNIP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device for storing the data to be coupled.	ANY16
	n	Number of couples (no processing in the case of 0 to 4, n=0)	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device for storing the coupled data.	ANY16

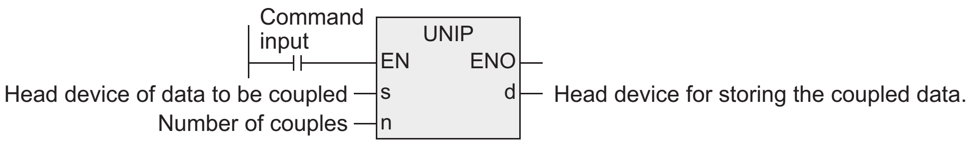
3. Applicable devices

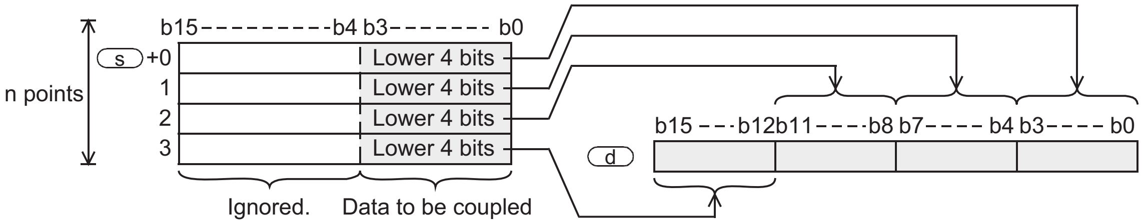
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s												●	●	●	●				●					
d												●	●	●	●				●					
n														●	●					●	●			

Function and operation explanation

1. 16-bit operation (UNI/UNIP)

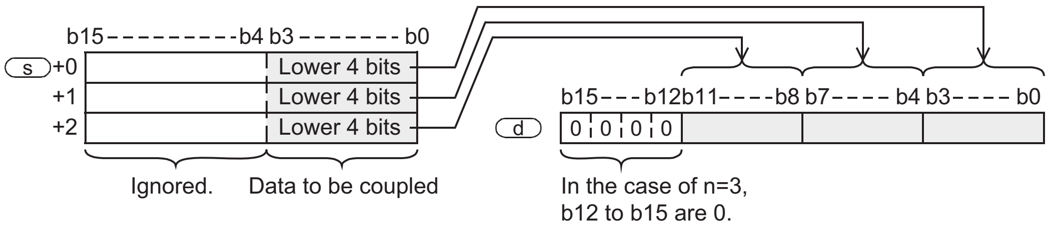
1) 16-bit data coupling lower 4 bits of 16-bit data of n points from the device specified by (s) is stored in the device specified by (d) as shown below.





2) Any one of 1 to 4 is specified by n.
In the case of n=0, instruction is not processed.

3) In the case of 1 ≤ n ≤ 3, higher (4×(4-n)) bits of the device specified by (d) are 0.
For example, in the case of n=3, lower 4 bits of (s) to ((s)+2) are stored in b0 to b11 of (d), and higher 4 bits of (d) are 0.



Related instructions

Instruction	Content
DIS	This instruction separates 16-bit data in 4-bit unit.

Cautions

FX3UC PLC supports at V2.20 or later.

Error

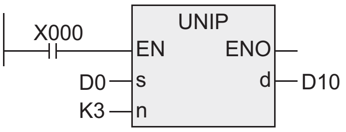
In the following case, it is an operation error, error flag M8067 is turned ON, and error code is stored in D8067.

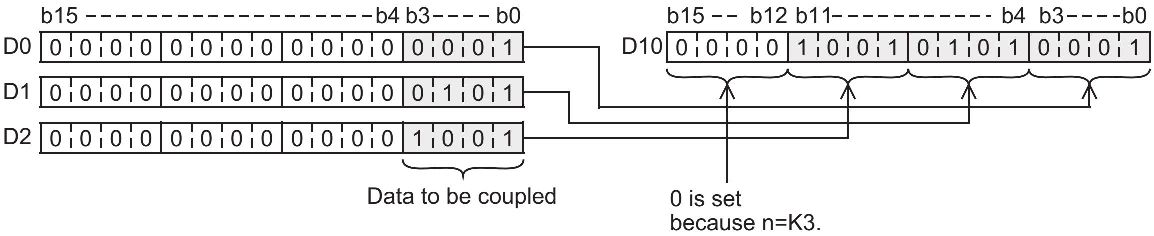
1) When (s) to ((s)+n) exceeds the device range of the specified devices. (Error code: K6706)
2) When n is specified other than 0 to 4. (Error code: K6706)

Program example

This is a program for coupling lower 4 bits of D0 to D2 when the X000 is turned ON, and storing in D10.

[Structured ladder]





[ST]

```
UNIP(X000,D0,K3,D10);
```

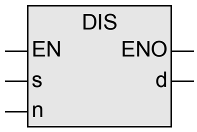
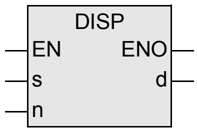
7.13.5 DIS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction separates 16-bit data in 4-bit unit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DIS	16 bits	Continuous		DIS(EN,s,n,d);
DISP	16 bits	Pulse		DISP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Device for storing the data to be separated.	ANY16
	n	Number of separates (no processing in the case of 0 to 4, n=0)	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device for storing the separated data.	ANY16

3. Applicable devices

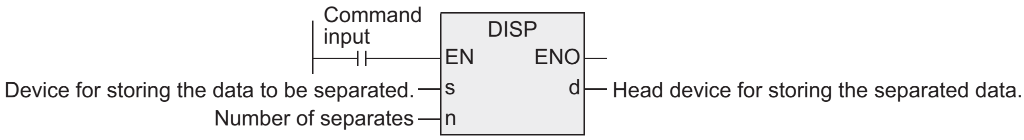
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s												●	●	●	●				●					
d												●	●	●	●				●					
n														●	●					●	●			

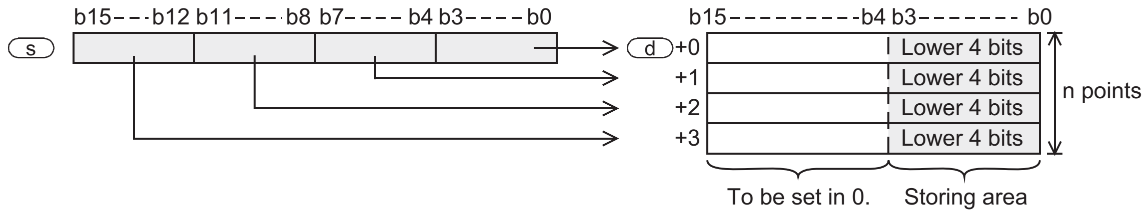
1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

1. 16-bit operation (DIS/DISP)

1) This instruction separates 16-bit data of the device specified by (s) in 4-bit unit, and storing in the device specified by (d) as follows.





2) Any one of 1 to 4 is specified by n.
In the case of n=0, instruction is not processed.

3) 0 is set in higher 12 bits in the device of n points from the device specified by (d).

Related instructions

Instruction	Content
UNI	This instruction couples lower 4 bits of 16-bit data.

Cautions

FX3UC PLC supports at V2.20 or later.

Error

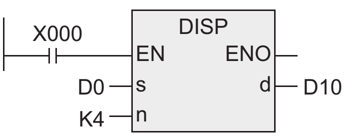
In the following case, it is an operation error, error flag M8067 is turned ON, and error code is stored in D8067.

1) When the range of n points from the device specified by (d) exceeds the device range of specified device. (Error code: K6706)
2) When n is specified other than 0 to 4. (Error code: K6706)

Program example

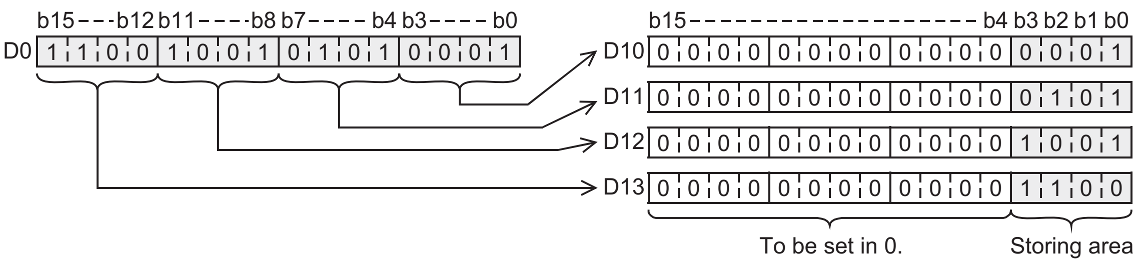
This is a program for separating D0 in 4 bits each when the X000 is turned ON, and storing in D10 to D13.

[Structured ladder]



[ST]

```
DISP(X000,D0,K4,D10);
```



7.13.6 SWAP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	○	×	×	×	×	×

Outline

This instruction swaps higher 8 bits and lower 8 bits of word data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
SWAP	16 bits	Continuous	SWAP [EN, s → ENO]	SWAP(EN,s);
SWAPP	16 bits	Pulse	SWAPP [EN, s → ENO]	SWAPP(EN,s);
DSWAP	32 bits	Continuous	DSWAP [EN, s → ENO]	DSWAP(EN,s);
DSWAPP	32 bits	Pulse	DSWAPP [EN, s → ENO]	DSWAPP(EN,s);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Device for swapping higher and lower bytes.	ANY16	ANY32
Output variable	ENO	Execution state	Bit	

3. Applicable devices

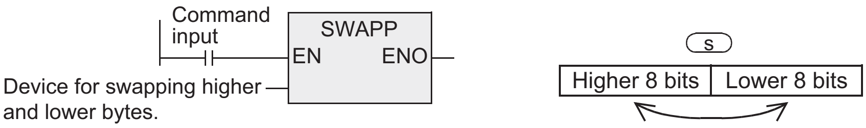
Operand type	Bit Devices: System user							Word Devices: Digit designation				Special unit				Index				Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)									●	●	●	●	●	●	▲1	▲1	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation

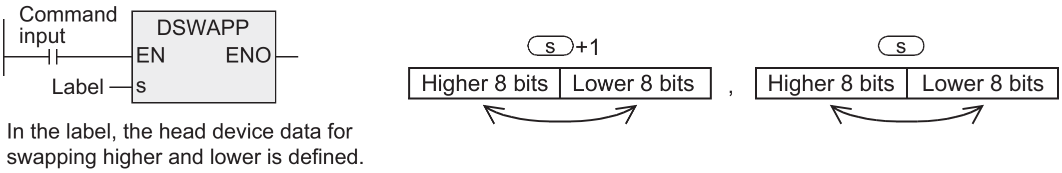
1. 16-bit operation (SWAP, SWAPP, DSWAP, DSWAPP)

This instruction swaps lower 8 bits and higher 8 bits.



2. 32-bit operation (DSWAP, DSWAPP)

In the case of 32-bit instruction, too, lower 8 bits and higher 8 bits are swapped individually.



Cautions

1) When the continuous execution instruction is used, it must be noted that the swap takes place in every operation cycle.
2) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
   When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
   When specifying the device, use the global label.
3) Applicable devices are limited.
   ▲1: FX3U, FX3UC PLCs only are applicable.

7.13.7 SORT2

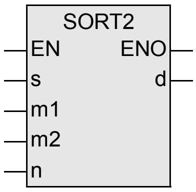
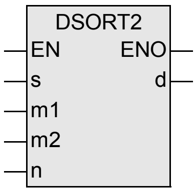
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction sorts the data table composed of data (rows) and group data (columns) in the ascending order/descending order in row unit on the basis of the specified group data (rows). In this instruction, data (row direction) is stored in continuous devices, and it is easy to add the data (row).
In SORT, sorting is in ascending order only, and data composition is different (data is composed in devices continuous in row direction).

→ As for SORT, refer to section 7.7.10.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
SORT2	16 bits	Continuous		SORT2(EN,s,m1,m2,n,d);
DSORT2	32 bits	Continuous		DSORT2(EN,s,m1,m2,n,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Head device for storing data table [m1 × m2 points occupied]	ANY16	ANY32
	m1	Number of data (rows) [1 to 32]	ANY16	ANY32
	m2	Number of group data (columns) [1 to 6]	ANY16	ANY32
	n	Columns of group data (columns) as reference for sorting [1 to m2]	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head device for storing operation result [m1 × m2 points occupied]	ANY16	ANY32

3. Applicable devices

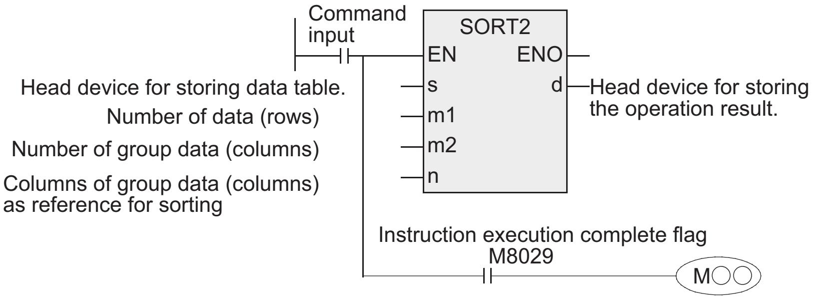
Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	●									
m1														●	●					●	●			
m2																				●	●			
d														●	●									
n														●	●					●	●			

Function and operation explanation

1. 16-bit operation (SORT2)

This instruction sorts the data rows of data table (sorting source) of (m1×m2) points from the device specified by (s) in the ascending order/descending order on the basis of group data of n rows, and stores in data table (after sorting) of (m1×m2) points from the device specified by (d).

→ **As for operation example, refer to 3. Operation examples.**



The data table is composed of m1=K3, m2=K4 in the sorting source as explained in this case. In the data table after sorting, exchange (s) and (d).

Column number / Row number		Number of groups m2 (in the case of m2=K4)			
		1	2	3	4
		Management number	Height	Body weight	Age
In the case of number of data m1=3	1	(s)	(s)+1	(s)+2	(s)+3
	2	(s)+4	(s)+5	(s)+6	(s)+7
	3	(s)+8	(s)+9	(s)+10	(s)+11

1) Sorting is set depending on the ON/OFF state of M8165.

	Setting of sorting order
M8165=ON	Descending
M8165=OFF	Ascending

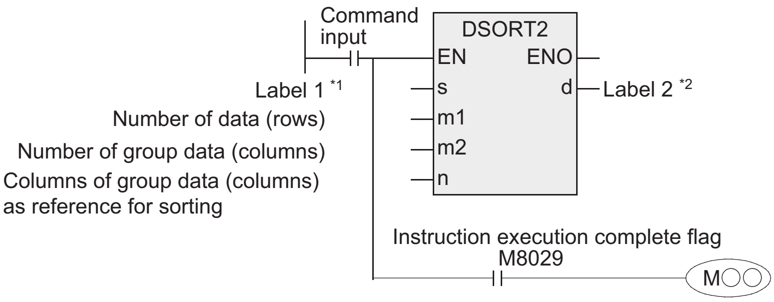
2) Data sorting is started when the instruction input is ON, and is completed after m1 scan, and instruction execution complete flag M8029 is turned ON.

→ **As for the manner of using the instruction execution complete flag, refer to section 1.3.4.**

2. 32-bit operation (DSORT2)

This instruction sorts the data rows of data table (sorting source) of (m1×m2) points from the device specified by (s) in the ascending order/descending order on the basis of group data of n rows, and stores in data table (after sorting) of (m1×m2) points from the device specified by (d).

→ **As for operation example, refer to 3. Operation examples.**



*1. Head device for storing the data table is defined in label 1.

*2. Head device for storing the operation result is defined in label 1.

The data table is composed of m1=K3, m2=K4 in the sorting source as explained in this case. In the data table after sorting, exchange (s) and (d).

Row number \ Column number		Number of groups m2 (in the case of m2=K4)			
		1	**2**	**3**	**4**
		Management number	**Height**	**Body weight**	**Age**
In the case of number of data m1=3	**1**	[(s)+1, (s)]	[(s)+3, (s)+2]	[(s)+5, (s)+4]	[(s)+7, (s)+6]
	2	[(s)+9, (s)+8]	[(s)+11, (s)+10]	[(s)+13, (s)+12]	[(s)+15, (s)+14]
	3	[(s)+17, (s)+16]	[(s)+19, (s)+18]	[(s)+21, (s)+20]	[(s)+23, (s)+22]

1) Sorting is set depending on the ON/OFF state of M8165.

	Sorting order
M8165=ON	Descending
M8165=OFF	Ascending

2) When using data register D or extension register R for m1, the data length is 32 bits. For example, when m1 is specified by D0, m1 is 32-bit data of [D1, D0].

3) Data sorting is started when the instruction input is ON, and is completed after m1 scan, and instruction execution complete flag M8029 is turned ON.

→ **As for the manner of using the instruction execution complete flag, refer to section 1.3.4.**

3. Operation example

The operation is as follows when the following sorting source data is executed in "n=K2 (column number 2)", "n=K3 (column number 3)".
The operation example is a case of 16-bit operation. In the case of 32-bit operation, the data table should be composed of BIN 32 bits.
When a serial number such as management number is entered in the first column, it is convenient because the original row number can be judged from its content.

Sorting source data

Row number \ Column number		Number of groups m2 (in the case of m2=K4)			
		1	2	3	4
		Management number	Height	Body weight	Age
In the case of number of data m1=5	1	(s)	(s)+1	(s)+2	(s)+3
		1	150	45	20
	2	(s)+4	(s)+5	(s)+6	(s)+7
		2	180	50	40
	3	(s)+8	(s)+9	(s)+10	(s)+11
		3	160	70	30
	4	(s)+12	(s)+13	(s)+14	(s)+15
		4	100	20	8
	5	(s)+16	(s)+17	(s)+18	(s)+19
		5	150	50	45

1) Sorting result when instruction is executed in n=K2 (column number 2) (ascending order)

Row number \ Column number	1	2	3	4
	Management number	Height	Body weight	Age
1	(d)	(d)+1	(d)+2	(d)+3
	4	100	20	8
2	(d)+4	(d)+5	(d)+6	(d)+7
	1	150	45	20
3	(d)+8	(d)+9	(d)+10	(d)+11
	5	150	50	45
4	(d)+12	(d)+13	(d)+14	(d)+15
	3	160	70	30
5	(d)+16	(d)+17	(d)+18	(d)+19
	2	180	50	40

2) Sorting result when instruction is executed in n=K3 (column number 3) (descending order)

Row number \ Column number	1	2	3	4
	Management number	Height	Body weight	Age
1	(d)	(d)+1	(d)+2	(d)+3
	3	160	70	30
2	(d)+4	(d)+5	(d)+6	(d)+7
	2	180	50	40
3	(d)+8	(d)+9	(d)+10	(d)+11
	5	150	50	45
4	(d)+12	(d)+13	(d)+14	(d)+15
	1	150	45	20
5	(d)+16	(d)+17	(d)+18	(d)+19
	4	100	20	8

Related devices

→ **As for the manner of using the instruction execution complete flag, refer to section 1.3.4.**

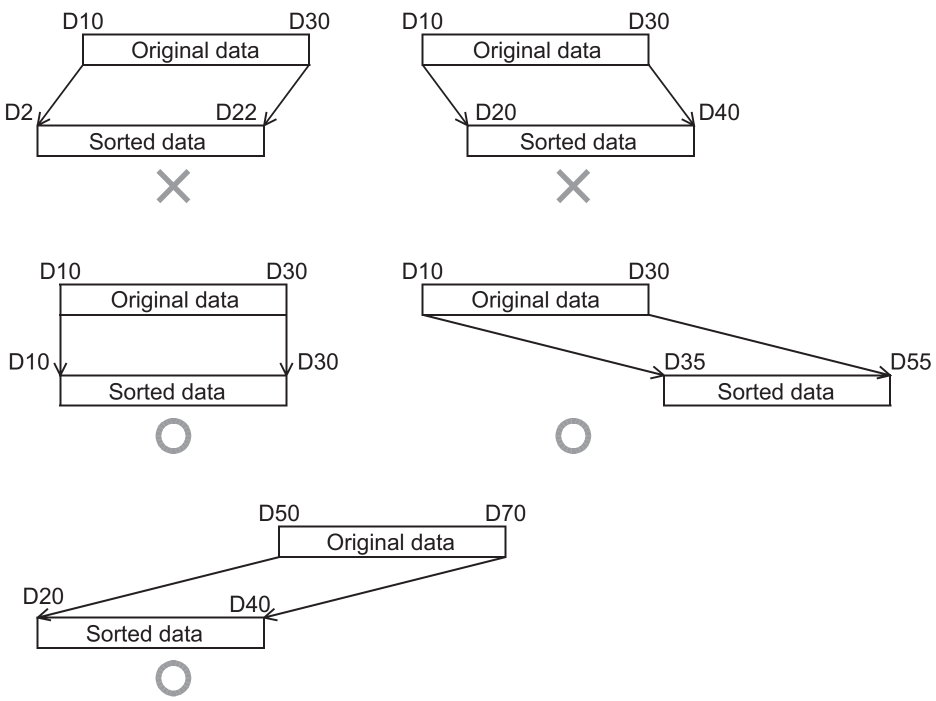
Device	Name	Content
M8029	Instruction execution complete	Turned ON when data sorting instruction is complete.
M8165	Descending order	Sorted in descending order when M8165=ON. Sorted in ascending order when M8165=OFF.

Related instructions

Instruction	Content
SORT	Data sorting This instruction sorts the data table composed of data (rows) and group data (columns) in ascending order in row unit on the basis of the specified group data (columns). In this instruction, the group data (column direction) is stored in continuous devices.

Cautions

1) During operation, do not change the content of the operand or data.
2) When executing again, once turn OFF the command input.
3) Limitation of times of instruction
   Up to two instructions can be driven simultaneously in the program.
4) The circuit block including this instruction cannot be written during RUN.
5) When specifying same device in s and d
   The original data is sorted in the data order after sorting.
   In particular, be careful not to change the content of d until instruction execution is complete.
6) Be careful not to allow overlap of original data and sorted data due to deviation.



7) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
   When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
   When specifying the device, use the global label.
8) FX3UC PLC supports at V2.20 or later.

7.14 Positioning Control

7.14.1 DSZR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	×	×	×	×	×	×

Outline

This instruction matches the mechanical position and the current value register in the PLC by zero return.
This instruction can perform following operation which is not supported by ZRN.

- Corresponding action of DOG search function
- Zero return is possible by using near-point DOG and zero-point signal.
  Zero point cannot be determined by counting the zero-point signals.

→ As for explanation of the instruction, see the positioning control manual.
→ As for cautions of use of high speed output special adapter, see the positioning control manual.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
DSZR	16 bits	Continuous	DSZR block: EN / ENO, s1 / d1, s2 / d2	DSZR(EN,s1,s2,d1,d2);

2. Set data

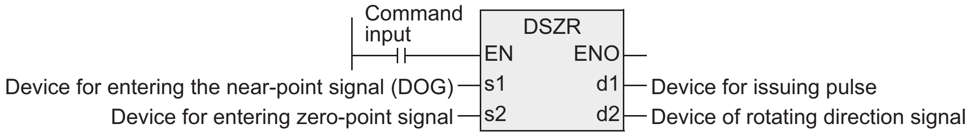
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Device for entering the near-point signal (DOG)	Bit
	s2	Device for entering zero-point signal	Bit
Output variable	ENO	Execution state	Bit
	d1	Device for issuing pulse (Y)	Bit
	d2	Device of rotating direction signal	Bit

3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit designation				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1	●	●	●			●	▲1												●					
s2	▲2																		●					
d1		▲3																	●					
d2		▲4	●			●	▲1												●					

▲: Refer to "Cautions".

Function and operation explanation



Cautions

1) Applicable devices are limited.

▲1: FX3U, FX3UC PLCs only are applicable, index (V, Z) decoration is disabled.

▲2: Please specify X000 to X007.

▲3: Please specify Y000, Y001, Y002[*1] of transistor output of basic unit, or Y000, Y001, Y002[*3], Y003[*3] of high speed output special adapter[*2].

*1. Y002 cannot be used in the 14-point type or 24-point type of FX3G PLC.
*2. High speed output special adapter can be connected only in FX3U PLC.
*3. When using Y002, Y003 in high speed output special adapter, a second high speed output special adapter is needed.

Points

- When using FX3U PLC of relay output type, high speed output special adapter is needed.
- The output of high speed output special adapter is a differential line driver.

▲4: When high speed output special adapter is used at the pulse output destination in the FX3U PLC, as the rotating direction signal, use the output shown in the table below.

Connection position of high speed output special adapter	Pulse output	Rotating direction signal
First unit	d1 =Y000	d2 =Y004
	d1 =Y001	d2 =Y005
Second unit	d1 =Y002	d2 =Y006
	d1 =Y003	d2 =Y007

When built-in transistor output is used at the pulse output destination in the FX3U, FX3UC, FX3G PLCs, as the rotating direction signal, use the transistor output.

2) Output number of rotating direction signal d2
Operation changes as follows depending on polarity of output pulse frequency.
[+ (Positive)] → d2 : ON
[- (Negative)] → d2 : OFF

Cautions about writing during RUN

Avoid writing during RUN while DSZR is being executed (during pulse output).
If, during pulse output, writing during RUN is attempted in the circuit block including this instruction, it must be noted that the pulse output slows down and stops.

Function changes by version

This instruction includes the function changes as shown in the table below depending on the version.
→ **As for the explanation of the instruction and contents of function changes, refer to the positioning control manual.**

Corresponding version			Item	Outline of function
FX3G	FX3U	FX3UC		
Ver. 1.00 or later	Ver. 2.20 or later	Ver. 2.20 or later	Clear signal output destination designating function	When special auxiliary relay corresponding to d1 is turned ON, the clear signal output specification is changed to the output number specified by special data register corresponding to d

7.14.2 DVIT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction executes one-speed interrupt inching.

→ **As for explanation of the instruction, see the positioning control manual.**

→ **As for cautions of use of high speed output special adapter, see the positioning control manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DVIT	16 bits	Continuous	DVIT EN ENO s1 d1 s2 d2	DVIT(EN,s1,s2,d1,d2);
DDVIT	32 bits	Continuous	DDVIT EN ENO s1 d1 s2 d2	DDVIT(EN,s1,s2,d1,d2);

2. Set data

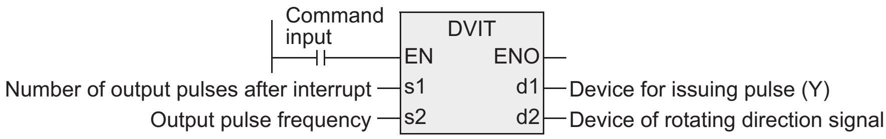
Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Number of output pulses after interrupt (relative address)	ANY16	ANY32
	s2	Output pulse frequency	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d1	Device for issuing pulse (Y)	Bit	
	d2	Device of rotating direction signal	Bit	

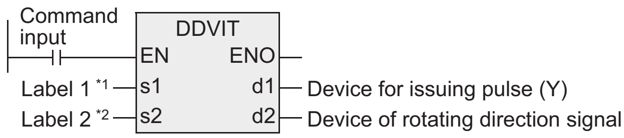
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	●	●			●	●	●			
s2								●	●	●	●	●	●	●	●	●			●	●	●			
d1		▲1																	●					
d2		▲2	●			●	▲3												●					

▲: Refer to "Cautions".

Function and operation explanation





*1. Number of output pulses after interrupt is defined.
*2. Number of output pulses is defined.

Cautions about writing during RUN

Avoid writing during RUN while DVIT is being executed (during pulse output).
If, during pulse output, writing during RUN is attempted in the circuit block including this instruction, it must be noted that the pulse output slows down and stops.

Function changes by version

This instruction includes the function changes as shown in the table below depending on the version.
→ **As for the explanation of the instruction and contents of function changes, refer to the positioning control manual.**

Corresponding version		Item	Outline of function
FX3U	FX3UC		
Ver. 2.20 or later	Ver. 1.30 or later	Interrupt input signal designating function	When M8336 is turned ON, the interrupt input signal corresponding to Y000 to Y003 is changed to input number (X000 to X007) specified by D8336. However, Y003 cannot be specified when transistor output of basic unit is used.
Ver. 2.20 or later	Ver. 2.20 or later	User interrupt mode	When 8 is specified by D8336 in interrupt input signal corresponding to Y000 to Y003 and M8336 is turned ON, the interrupt input signal is changed to special auxiliary relay. When the changed special auxiliary relay is changed from OFF to ON by input interrupt program, the interrupt operation is started. However, when this function is used, the logic of interrupt input cannot be inverted. However, Y003 cannot be specified when transistor output of basic unit is used.

Cautions

1) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
   When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
   When specifying the device, use the global label.
2) Applicable devices are limited.
   ▲1: Please specify Y000, Y001, Y002 of transistor output of basic unit, or Y000, Y001, Y002*2, Y003*2 of high speed output special adapter*1.
   *1. High speed output special adapter can be connected only in FX3U PLC.
   *2. When using Y002, Y003 in high speed output special adapter, a second high speed output special adapter is needed.

   Points
   - When using FX3U PLC of relay output type, high speed output special adapter is needed.
   - The output of high speed output special adapter is a differential line driver.

   ▲2: When high speed output special adapter is used at the pulse output destination in the FX3U PLC, as the rotating direction signal, use the output shown in the table below.

Connection position of high speed output special adapter	Pulse output	Rotating direction signal
First unit	(d1)=Y000	(d2)=Y004
	(d1)=Y001	(d2)=Y005
Second unit	(d1)=Y002	(d2)=Y006
	(d1)=Y003	(d2)=Y007

   When built-in transistor output is ussed at the pulse output destination in the FX3U and FX3UC PLCs, as the rotating direction signal, use the transistor output.
   ▲3: In D□.b, index (V, Z) decoration is disabled.
3) Number of output pulses after interrupt is specified (specified by (s1)).
   The setting range is as follows.
   a) In the case of 16-bit operation
      -32,768 to +32,767 (excluding 0)
   b) In the case of 32-bit operation
      -999,999 to +999,999 (excluding 0)
4) Output pulse frequency is specified (specified by (s2)).
   The setting range is as follows.
   a) In the case of 16-bit operation
      10 to 32,767 (Hz)
   b) In the case of 32-bit operation

Pulse output destination		Setting range
FX3U PLC	High speed output special adapter	10 to 200,000 (HZ)
FX3U, FX3UC PLCs	Basic unit (transistor output)	10 to 100,000 (HZ)

5) Output number of rotating direction signal (d2)
   Operation changes as follows depending on polarity of output pulse frequency.
   [+ (Positive)] → (d2): ON
   [- (Negative)] → (d2): OFF

7.14.3 DTBL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	○	×	×	×	×	×	×

Outline

This instruction operates the instruction predetermined in the data table by GX Works2 by one specified table.

→ **As for explanation of the instruction, see the positioning control manual.**
→ **As for cautions of use of high speed output special adapter, see the positioning control manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DTBL	32 bits	Continuous	DTBL EN ENO n d	DTBL(EN,n,d);

2. Set data

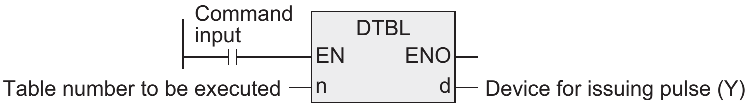
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	n	Table number to be executed (n=1 to 100)	ANY32
Output variable	ENO	Execution state	Bit
	d	Device for issuing pulse (Y)	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
d		▲1																						
n																				●	●			

▲: Refer to "Cautions".

Function and operation explanation



Cautions about writing during RUN

The circuit block including DTBL cannot be written during RUN.

Cautions

1) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
   When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
   When specifying the device, use the global label.
2) FX3UC PLC supports at V2.20 or later.
3) Applicable devices are limited.
   ▲1: Please specify Y000, Y001, Y002[*1] of transistor output of basic unit, or Y000, Y001, Y002[*3], Y003[*3] of high speed output special adapter[*2].
   *1. Y002 cannot be used in the 14-point type or 24-point type of FX3G PLC.
   *2. High speed output special adapter can be connected only in FX3U PLC.
   *3. When using Y002, Y003 in high speed output special adapter, a second high speed output special adapter is needed.

   Points

   - When using FX3U PLC of relay output type, high speed output special adapter is needed.
   - The output of high speed output special adapter is a differential line driver.

7.14.4 DABS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	△	○	○	×	×	×

Outline

This instruction connects with our company's MR-H, MR-J2(S) or MR-J3 type servo amplifier (with absolute position detecting function), and reads out the absolute position (ABS) data. The data is read out in pulse converted value.

→ **As for explanation of the instruction, see the positioning control manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DABS	32 bits	Continuous	DABS block: EN, s inputs; ENO, d1, d2 outputs	DABS(EN,s,d1,d2);

2. Set data

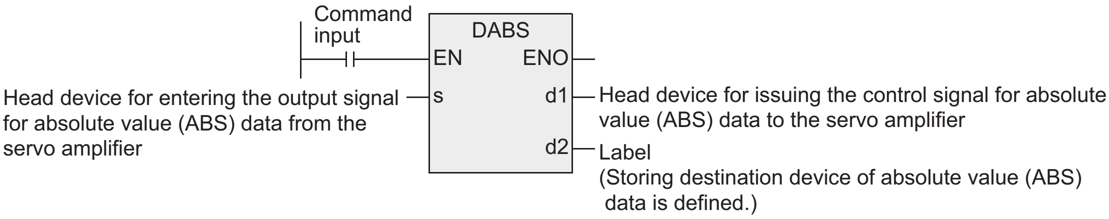
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device for entering the output signal for absolute value (ABS) data from the servo amplifier (3 points occupied).	Bit
Output variable	ENO	Execution state	Bit
	(d1)	Head device for issuing the control signal for absolute value (ABS) data to the servo amplifier (3 points occupied).	Bit
	(d2)	Storing destination device of absolute value (ABS) data (32-bit value)	ANY32

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)	●	●	●			●	▲2												●					
(d1)		▲1	●			●	▲2												●					
(d2)									●	●	●	●	●	●	▲3	▲4	●	●	●					

▲: Refer to "Cautions".

Function and operation explanation



Cautions

1) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
   When handling 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
   When specifying the device, use the global label.
2) FX2N, FX2NC PLCs support the instruction at V3.00 or later.
3) Since ABS data is read out in pulse converted value, please specify "Motor system" for parameter setting (BFM#3) of FX2N-1PG. (FX1S, FX1N, FX2N, FX1NC, FX2NC PLCs.)
4) Writing of ABS data into FX2N-10PG should be addressed to the current value registers (BFM#40, #39) in which the pulse converted values are stored. (FX1S, FX1N, FX2N, FX1NC, FX2NC PLCs.)
5) Applicable devices are limited.
   ▲1: Please designate the transistor output.
   ▲2: FX3U, FX3UC PLCs only are applicable, index (V, Z) decoration is disabled.
   ▲3: FX3U, FX3UC, FX3G PLCs only are applicable.
   ▲4: FX3U, FX3UC PLCs only are applicable.

7.14.5 ZRN

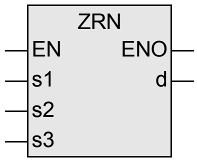
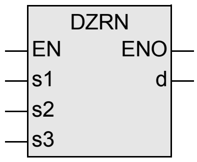
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	×	○	○	×	×	×

Outline

This instruction matches the mechanical position and the current value register in the PLC by zero return.
Please use DSZR when DOG search function is necessary.

→ **As for explanation of the instruction, see the positioning control manual.**
→ **As for cautions of use of high speed output special adapter, see the positioning control manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
ZRN	16 bits	Continuous		ZRN(EN,s1,s2,s3,d);
DZRN	32 bits	Continuous		DZRN(EN,s1,s2,s3,d);

2. Set data

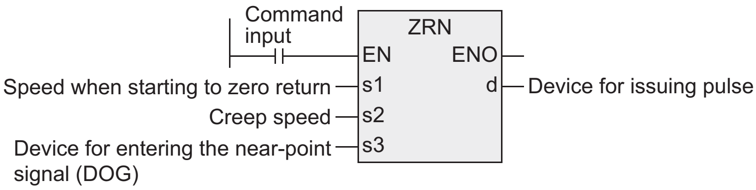
Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Speed when starting to zero return	ANY16	ANY32
	s2	Creep speed	ANY16	ANY32
	s3	Device for entering the near-point signal (DOG)	Bit	
Output variable	ENO	Execution state	Bit	
	d	Device for issuing pulse	Bit	

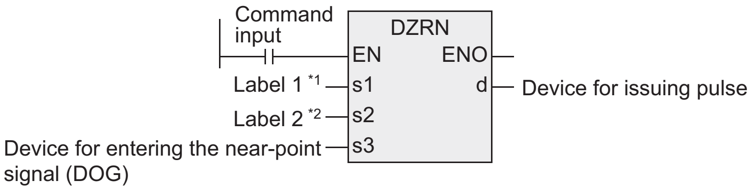
3. Applicable devices

Operand type	Bit Devices: System user X	Y	M	T	C	S	D□.b	Word Devices: Digit designation KnX	KnY	KnM	KnS	System user T	C	D	R	Special unit U□\G□	Index V	Z	Modifier	Others: Constant K	H	Real Number E	Character String "□"	Pointer P
s1								●	●	●	●	●	●	●	▲3	▲2	●	●	●	●	●			
s2								●	●	●	●	●	●	●	▲3	▲2	●	●	●	●	●			
s3	●	●	●			●	▲1												●					
d		▲4																	●					

▲: Refer to "Cautions".

Function and operation explanation





*1. The speed when starting to zero return is defined.

*2. The creep speed is defined.

Cautions about writing during RUN

1) Avoid writing during RUN while ZRN or DZRN is being executed (during pulse output). (FX3U, FX3UC, FX3G PLCs.)
   If, during pulse output, writing during RUN is attempted in the circuit block including this instruction, it must be noted that the pulse output slows down and stops.
2) While ZRN or DZRN is being executed, the following writing during RUN is prohibited. (FX1S, FX1N, FX1NC PLCs.)
   Program change of circuit block including this instruction.
   Program change of circuit block immediately before or immediately after the circuit block including this instruction.
   Addition or deletion of circuit block immediately before or immediately after the circuit block including this instruction.

Function changes by version

This instruction includes the function changes as shown in the table below depending on the version.
→ **As for the explanation of the instruction and contents of function changes, refer to the positioning control manual.**

1. FX3U. FX3UC PLCs

[V2.20 or later]
When the special auxiliary relay corresponding to (d) is turned ON, the output destination of clear signal is changed to the output number specified by the special data register corresponding to (d).

2. FX1S PLC

[Before V2.00]
After resetting the "rotating direction signal" (normal rotation output) used in other positioning instruction by RST, start to zero return.
When started to zero return while the "rotating direction signal" is in normal rotation output state, the motor moves to the normal rotation, not zero return.
[After V2.00]
When driving of other positioning instruction is turned OFF, the rotating direction signal is turned OFF at the same time, and processing of before V2.00 is not required.

3. FX1S, FX1N PLCs

[Before V2.00]
If the zero return speed does not reach the speed specified by the instruction, change the acceleration or deceleration speed (D8148) to other value than given below.
(For example, if the current value is 150, change to 149 or 151.)
50, 75, 90, 100, 125, 150, 225, 250, 375, 450, 500, 750, 900, 1120, 1500, 2250, 4499, 4500
[After V2.00]
The above processing is not required.

Cautions

1) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
When specifying the device, use the global label.
2) Instruction driving timing (FX1S, FX1N, FX1NC PLCs)
This is an instruction allowed to program as many times as desired, but you are advised to design the instruction driving timing according to the following cautions.
   a) Do not drive simultaneously the positioning instruction using the same output relay (Y000 or Y001).
   If driven simultaneously, it is handled as double coil, and normal function is disabled.
   b) After turning OFF the direction input of the instruction, drive again after the following condition is established.
   Condition: Re-driving is allowed after one operation cycle or more from the OFF moment of "pulse output mode monitor (Y000: [M8147], Y001: [M8148]" of the positioning instruction driven previous time.
   This is because one or more OFF operation is required for re-driving of the positioning instruction.
   c) We recommend the step-ladder instruction (STL) as a method of programming correctly the positioning instruction according to the cautions mentioned above.
3) Not applicable to DOG search function, please start the zero return operation from the front side of the near-point signal. (FX1S, FX1N, FX1NC PLCs.)
4) Not applicable to zero-point signal of servo motor, please adjust to the position of near-point signal (DOG) when fine adjustment of zero point is necessary. (FX1S, FX1N, FX1NC PLCs.)

5) While zero return, the numeric values of the current value registers (Y000: [D8141, D8140], Y001: [D8143, D8142]) move in the decreasing direction. (FX1S, FX1N, FX1NC PLCs)
When zero return in reverse direction, please control the output relay (Y) wired as "rotating direction signal" by the program in the following procedure.
   1) Set (ON) Y□□□ (rotating direction signal).
   2) Execute zero return instruction.
   3) Reset (OFF) Y□□□ (rotating direction signal) by execution complete flag (M8012) of zero return instruction.

6) Applicable devices are limited.
▲1: FX3U, FX3UC PLCs only are applicable, index (V, Z) decoration is disabled.
▲2: FX3U, FX3UC PLCs only are applicable.
▲3: FX3U, FX3UC, FX3G PLCs only are applicable.
▲4: <FX3U, FX3UC, FX3G PLCs>
Please specify Y000, Y001, Y002*1 of transistor output of basic unit, or Y000, Y001, Y002*3, Y003*3 of high speed output special adapter*2.
*1 Y002 cannot be used in the 14-point type or 24-point type of FX3G PLC.
*2 High speed output special adapter can be connected only in FX3U PLC.
*3 When using Y002, Y003 in high speed output special adapter, a second high speed output special adapter is needed.

Points

- When using FX3U PLC of relay output type, high speed output special adapter is needed.
- The output of high speed output special adapter is a differential line driver.

<FX1S, FX1N, FX1NC PLCs>
Specify Y000 or Y001.
As the output of the PLC, please use the transistor output.

7) Specify the speed of return zero point in (s1).
The setting range is as follows.
   a) In the case of 16-bit operation
   - FX3U, FX3UC, FX3G PLCs
     10 to 32,767 (Hz)
   - FX1S, FX1N PLCs
     10 to 32,767 (Hz)
   - FX1NC PLC
     10 to 10,000 (Hz)

   b) In the case of 32-bit operation
   - FX3U, FX3UC, FX3G PLCs

Pulse output destination		Setting range
FX3U PLC	High speed output special adapter	10 to 200,000 (HZ)
FX3U, FX3UC, FX3G PLCs	Basic unit (transistor output)	10 to 100,000 (HZ)

   - FX1S, FX1N PLCs
     10 to 100,000 (Hz)
   - FX1NC PLC
     10 to 10,000 (Hz)

7.14.6 PLSV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	×	○	○	×	×	×

Outline

This instruction issues a variable speed pulse with the rotating direction.

→ **As for explanation of the instruction, see the positioning control manual.**
→ **As for cautions of use of high speed output special adapter, see the positioning control manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
PLSV	16 bits	Continuous	PLSV [EN, s → ENO, d1, d2]	PLSV(EN,s,d1,d2);
DPLSV	32 bits	Continuous	DPLSV [EN, s → ENO, d1, d2]	DPLSV(EN,s,d1,d2);

2. Set data

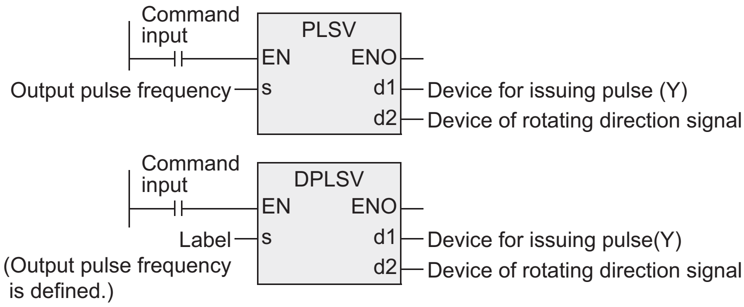
Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Output pulse frequency	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d1)	Device for issuing pulse (Y)	Bit	
	(d2)	Device of rotating direction signal	Bit	

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲4	▲5	●	●	●	●	●			
(d1)		▲1																	●					
(d2)		▲2	●			●	▲3												●					

▲: Refer to "Cautions".

Function and operation explanation



Cautions about writing during RUN

1) Avoid writing during RUN while PLSV or DPLSV is being executed (during pulse output). (FX3U, FX3UC, FX3G PLCs)
If, during pulse output, writing during RUN is attempted in circuit block including this instruction, it must be noted that the pulse output becomes as follows.

	Operation when written during RUN while instruction is being executed
When operating with acceleration or deceleration*1	Pulse output slows down and stops.
When operating without acceleration or deceleration	Pulse output stops immediately.

*1. Only available for FX3U/FX3UC PLC Ver.2.20 or later and FX3G PLC.

2) While PLSV or DPLSV is being executed, the following writing during RUN is prohibited. (FX1S, FX1N, FX1NC PLCs.)
Program change of circuit block including this instruction.
Program change of circuit block immediately before or immediately after the circuit block including this instruction.
Addition or deletion of circuit block immediately before or immediately after the circuit block including this instruction.

Function changes by version

This instruction includes the function changes as shown in the table below depending on the version.
→ **As for the explanation of the instruction and contents of function changes, refer to the positioning control manual.**

Corresponding version			Added function	Outline of function
FX3G	FX3U	FX3UC		
Ver. 1.00 or later	Ver. 2.20 or later	Ver. 2.20 or later	Acceleration or deceleration operation function	When M8338 is turned ON, the operation accelerates or decelerates up to (s1) by setting of acceleration time or deceleration time corresponding to (d1) when (s1) is changed.

FX1N, FX1NC, FX1S are not applicable to this function.

Cautions

1) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
When specifying the device, use the global label.

2) Instruction driving timing (FX1S, FX1N, FX1NC PLCs)
This is an instruction allowed to program as many times as desired, but you are advised to design the instruction driving timing according to the following cautions.
   a) Do not drive simultaneously the positioning instruction using the same output relay (Y000 or Y001).
   If driven simultaneously, it is handled as double coil, and normal function is disabled.
   b) After turning OFF the direction input of the instruction, drive again after the following condition is established.
   Condition: Re-driving is allowed after one operation cycle or more from the OFF moment of "pulse output mode monitor (Y000: [M8147], Y001: [M8148]" of the positioning instruction driven previous time.
   This is because one or more OFF operation is required for re-driving of the positioning instruction.
   c) We recommend the step-ladder instruction (STL) as a method of programming correctly the positioning instruction according to the cautions mentioned above.

3) When output pulse frequency is changed to K0 during pulse output (FX1S, FX1N, FX1NC PLCs), the PLC stops the pulse output.
When sending output again, once turn OFF the flag during pulse output (Y000: [M8147], Y001: [M8148]), and after lapse of one operation cycle, set again (change) the output pulse frequency to other value than K0.
   a) Within one operation cycle, if the value is changed to other value than K0, the output maintains the stop function. In this case, write K0 again for more than one operation cycle, or once turn OFF the command input.

4) Applicable devices are limited.
▲1: <FX3U, FX3UC, FX3G PLCs>
Please specify Y000, Y001, Y002*1 of transistor output of basic unit, or Y000, Y001, Y002*3, Y003*3 of high speed output special adapter*2.
*1. Y002 cannot be used in the 14-point type or 24-point type of FX3G PLC.
*2. High speed output special adapter can be connected only in FX3U PLC.
*3. When using Y002, Y003 in high speed output special adapter, a second high speed output special adapter is needed.

Points

- When using FX3U PLC of relay output type, high speed output special adapter is needed.
- The output of high speed output special adapter is a differential line driver.

<FX1S, FX1N, FX1NC PLCs>
Specify Y000 or Y001.
As the output of the PLC, please use the transistor output.

▲2: <FX3U, FX3UC, FX3G PLCs>
When high speed output special adapter is used at the pulse output destination in the FX3U PLC, as the rotating direction signal, use the output shown in the table below.

Connection position of high speed output special adapter	Pulse output	Rotating direction signal
First unit	(d1)=Y000	(d2)=Y004
	(d1)=Y001	(d2)=Y005
Second unit	(d1)=Y002	(d2)=Y006
	(d1)=Y003	(d2)=Y007

When built-in transistor output is used at the pulse output destination in the FX3U, FX3UC, FX3G PLCs, as the rotating direction signal, use the transistor output.

▲3: FX3U, FX3UC PLCs only are applicable, index (V, Z) decoration is disabled.
▲4: FX3U, FX3UC, FX3G PLCs only are applicable.
▲5: FX3U, FX3UC PLCs only are applicable.

5) The output pulse frequency is specified by (s).
The setting range is as follows.
a) In the case of 16-bit operation
- FX3U, FX3UC, FX3G PLCs
  -32,768 to -1, +1 to 32,767 (Hz)
- FX1S, FX1N PLCs
  -32,768 to -1, +1 to 32,767 (Hz)
- FX1NC PLC
  -10,000 to -1, +1 to 10,000 (Hz)

b) In the case of 32-bit operation
- FX3U, FX3UC, FX3G PLCs

Pulse output destination		Setting range
FX3U PLC	High speed output special adapter	-200,000 to -1, +1 to 200,000 (HZ)
FX3U, FX3UC, FX3G PLCs	Basic unit (transistor output)	-100,000 to -1, +1 to 100,000 (HZ)

- FX1S, FX1N PLCs
  -100,000 to -1, +1 to 100,000 (Hz)
- FX1NC PLC
  -10,000 to -1, +1 to 10,000 (Hz)

6) Output number of rotating direction signal (d2)
Operation changes as follows depending on polarity of output pulse frequency.
[+ (Positive)] → (d2): ON
[- (Negative)] → (d2): OFF

7.14.7 DRVI

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	×	○	○	×	×	×

Outline

This instruction performs one-speed positioning by relative drive. The moving distance from the present position is specified together with plus or minus sign, and this is also called increment (relative) driving method.

→ As for explanation of the instruction, see the positioning control manual.
→ As for cautions of use of high speed output special adapter, see the positioning control manual.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DRVI	16 bits	Continuous	DRVI: EN / ENO, s1 / d1, s2 / d2	DRVI(EN,s1,s2,d1,d2);
DDRVI	32 bits	Continuous	DDRVI: EN / ENO, s1 / d1, s2 / d2	DDRVI(EN,s1,s2,d1,d2);

2. Set data

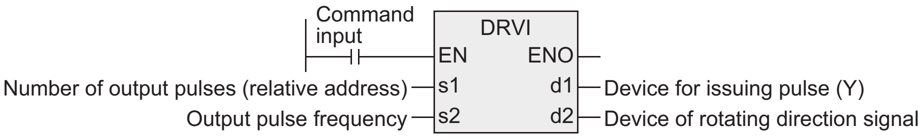
Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s1)	Number of output pulses (relative address)	ANY16	ANY32
	(s2)	Output pulse frequency	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d1)	Device for issuing pulse (Y)	Bit	
	(d2)	Device of rotating direction signal	Bit	

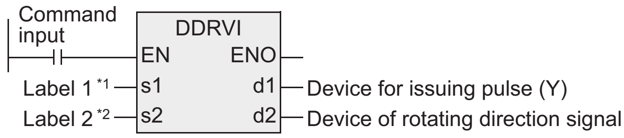
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲4	▲5	●	●	●	●	●			
(s2)								●	●	●	●	●	●	●	▲4	▲5	●	●	●	●	●			
(d1)		▲1																	●					
(d2)		▲2	●			●	▲3												●					

▲: Refer to "Cautions".

Function and operation explanation





*1. Number of output pulses (relative address) is defined.
*2. Output frequency pulse is defined.

Cautions about writing during RUN

1) Avoid writing during RUN while DRVI or DDRVI is being executed (during pulse output).
(FX3U, FX3UC, FX3G PLCs)
If, during pulse output, writing during RUN is attempted in circuit block including this instruction, it must be noted that the pulse output slows down and stops.
2) While DRVI or DDRVI is being executed, the following writing during RUN is prohibited.
(FX1S, FX1N, FX1NC PLCs.)
Program change of circuit block including this instruction.
Program change of circuit block immediately before or immediately after the circuit block including this instruction.
Addition or deletion of circuit block immediately before or immediately after the circuit block including this instruction.

Cautions

1) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
When specifying the device, use the global label.
2) Instruction driving timing (FX1S, FX1N, FX1NC PLCs)
This is an instruction allowed to program as many times as desired, but you are advised to design the instruction driving timing according to the following cautions.
   a) Do not drive simultaneously the positioning instruction using the same output relay (Y000 or Y001).
   If driven simultaneously, it is handled as double coil, and normal function is disabled.
   b) After turning OFF the direction input of the instruction, drive again after the following condition is established.
   Condition: Re-driving is allowed after one operation cycle or more from the OFF moment of "pulse output mode monitor (Y000: [M8147], Y001: [M8148]" of the positioning instruction driven previous time.
   This is because one or more OFF operation is required for re-driving of the positioning instruction.
   c) We recommend the step-ladder instruction (STL) as a method of programming correctly the positioning instruction according to the cautions mentioned above.

3) Applicable devices are limited.
▲1: <FX3U, FX3UC, FX3G PLCs>

Please specify Y000, Y001, Y002[*1] of transistor output of basic unit, or Y000, Y001, Y002[*3], Y003[*3] of high speed output special adapter[*2].

*1. Y002 cannot be used in the 14-point type or 24-point type of FX3G PLC.
*2. High speed output special adapter can be connected only in FX3U PLC.
*3. When using Y002, Y003 in high speed output special adapter, a second high speed output special adapter is needed.

Points

- When using FX3U PLC of relay output type, high speed output special adapter is needed.
- The output of high speed output special adapter is a differential line driver.

<FX1S, FX1N, FX1NC PLCs>
Specify Y000 or Y001.
As the output of the PLC, please use the transistor output.

▲2: <FX3U, FX3UC, FX3G PLCs>

When high speed output special adapter is used at the pulse output destination in the FX3U PLC, as the rotating direction signal, use the output shown in the table below.

Connection position of high speed output special adapter	Pulse output	Rotating direction signal
First unit	(d1)=Y000	(d2)=Y004
	(d1)=Y001	(d2)=Y005
Second unit	(d1)=Y002	(d2)=Y006
	(d1)=Y003	(d2)=Y007

When built-in transistor output is used at the pulse output destination in the FX3U, FX3UC, FX3G PLCs, as the rotating direction signal, use the transistor output.

▲3: FX3U, FX3UC PLCs only are applicable, index (V, Z) decoration is disabled.
▲4: FX3U, FX3UC, FX3G PLCs only are applicable.
▲5: FX3U, FX3UC PLCs only are applicable.

4) Number of output pulses is specified by (s1).
The setting range is as follows.
a) In the case of 16-bit operation
-32,768 to +32,767 (excluding 0)
b) In the case of 32-bit operation
-999,999 to +999,999 (excluding 0)

5) The output pulse frequency is specified by (s2).
The setting range is as follows.
a) In the case of 16-bit operation
- In the case of FX3U, FX3UC, FX3G PLCs
  10 to 32,767 (Hz)
- In the case of FX1S, FX1N PLCs
  10 to 32,767 (Hz)
- In the case of FX1NC PLC
  10 to 10,000 (Hz)

b) In the case of 32-bit operation
- In the case of FX3U, FX3UC, FX3G PLCs

Pulse output destination		Setting range
FX3U PLC	High speed output special adapter	10 to 200,000 (HZ)
FX3U, FX3UC, FX3G PLCs	Basic unit (transistor output)	10 to 100,000 (HZ)

- FX1S, FX1N PLCs
  10 to 100,000 (Hz)
- FX1NC PLC
  10 to 10,000 (Hz)

6) Output number of rotating direction signal (d2)
Operation changes as follows depending on polarity of output pulse frequency.
[+ (Positive)] → (d2): ON
[- (Negative)] → (d2): OFF

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

7.14.8 DRVA

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	×	○	○	×	×	×

Outline

This instruction performs one-speed positioning by absolute drive. The moving distance from the origin (0 point) is specified, and this is also called absolute driving method.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DRVA	16 bits	Continuous	DRVA EN ENO s1 d1 s2 d2	DRVA(EN,s1,s2,d1,d2);
DDRVA	32 bits	Continuous	DDRVA EN ENO s1 d1 s2 d2	DDRVA(EN,s1,s2,d1,d2);

2. Set data

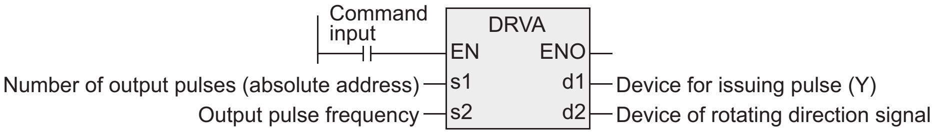
Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s1)	Number of output pulses (absolute address)	ANY16	ANY32
	(s2)	Output pulse frequency	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d1)	Device for issuing pulse (Y)	Bit	
	(d2)	Device of rotating direction signal	Bit	

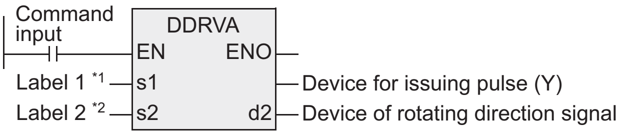
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲4	▲5	●	●	●	●	●			
(s2)								●	●	●	●	●	●	●	▲4	▲5	●	●	●	●	●			
(d1)		▲1																	●					
(d2)		▲2	●			●	▲3												●					

▲: Refer to "Cautions".

Function and operation explanation





*1. Number of output pulses (absolute address) is defined.
*2. Output frequency pulse is defined.

Cautions about writing during RUN

1) Avoid writing during RUN while DRVA or DDRVA is being executed (during pulse output). (FX3U, FX3UC, FX3G PLCs)
If, during pulse output, writing during RUN is attempted in circuit block including this instruction, it must be noted that the pulse output slows down and stops.
2) While DRVA or DDRVA is being executed, the following writing during RUN is prohibited. (FX1S, FX1N, FX1NC PLCs.)
Program change of circuit block including this instruction.
Program change of circuit block immediately before or immediately after the circuit block including this instruction.
Addition or deletion of circuit block immediately before or immediately after the circuit block including this instruction.

Cautions

1) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
When specifying the device, use the global label.
2) Instruction driving timing (FX1S, FX1N, FX1NC PLCs)
This is an instruction allowed to program as many times as desired, but you are advised to design the instruction driving timing according to the following cautions.
   a) Do not drive simultaneously the positioning instruction using the same output relay (Y000 or Y001). If driven simultaneously, it is handled as double coil, and normal function is disabled.
   b) After turning OFF the direction input of the instruction, drive again after the following condition is established.
   Condition: Re-driving is allowed after one operation cycle or more from the OFF moment of "pulse output mode monitor (Y000: [M8147], Y001: [M8148]" of the positioning instruction driven previous time.
   This is because one or more OFF operation is required for re-driving of the positioning instruction.
   c) We recommend the step-ladder instruction (STL) as a method of programming correctly the positioning instruction according to the cautions mentioned above.
   If, during pulse output, writing during RUN is attempted in circuit block including this instruction, it must be noted that the pulse output slows down and stops.
3) Applicable devices are limited.
▲1: <FX3U, FX3UC, FX3G PLCs>
Please specify Y000, Y001, Y002*1 of transistor output of basic unit, or Y000, Y001, Y002*3, Y003*3 of high speed output special adapter*2.
*1. Y002 cannot be used in the 14-point type or 24-point type of FX3G PLC.
*2. High speed output special adapter can be connected only in FX3U PLC.
*3. When using Y002, Y003 in high speed output special adapter, a second high speed output special adapter is needed.

Points

- When using FX3U PLC of relay output type, high speed output special adapter is needed.
- The output of high speed output special adapter is a differential line driver.

<FX1S, FX1N, FX1NC PLCs>
Specify Y000 or Y001.
As the output of the PLC, please use the transistor output.

▲2: <FX3U, FX3UC, FX3G PLCs>
When high speed output special adapter is used at the pulse output destination in the FX3U PLC, as the rotating direction signal, use the output shown in the table below.

Connection position of high speed output special adapter	Pulse output	Rotating direction signal
First unit	(d1) =Y000	(d2) =Y004
	(d1) =Y001	(d2) =Y005
Second unit	(d1) =Y002	(d2) =Y006
	(d1) =Y003	(d2) =Y007

When built-in transistor output is used at the pulse output destination in the FX3U, FX3UC, FX3G PLCs, as the rotating direction signal, use the transistor output.

▲3: FX3U, FX3UC PLCs only are applicable, index (V, Z) decoration is disabled.
▲4: FX3U, FX3UC, FX3G PLCs only are applicable.
▲5: FX3U, FX3UC PLCs only are applicable.

4) Number of output pulses is specified by (s1).
The setting range is as follows.
a) In the case of 16-bit operation
- In the case of FX3U, FX3UC, FX3G PLCs
-32,768 to +32,767
- FX1S, FX1N, FX1NC PLCs
-32,768 to +32,767

b) In the case of 32-bit operation
- In the case of FX3U, FX3UC, FX3G PLCs
-999,999 to +999,999
- FX1S, FX1N, FX1NC PLCs
-999,999 to +999,999

5) The output pulse frequency is specified by (s2).
The setting range is as follows.
a) In the case of 16-bit operation
- In the case of FX3U, FX3UC, FX3G PLCs
10 to 32,767 (Hz)
- In the case of FX1S, FX1N PLCs
10 to 32,767 (Hz)
- In the case of FX1NC PLC
10 to 10,000 (Hz)

b) In the case of 32-bit operation
- In the case of FX3U, FX3UC, FX3G PLCs

Pulse output destination		Setting range
FX3U PLC	High speed output special adapter	10 to 200,000 (HZ)
FX3U, FX3UC, FX3G PLCs	Basic unit (transistor output)	10 to 100,000 (HZ)

- FX1S, FX1N PLCs
10 to 100,000 (Hz)
- FX1NC PLC
10 to 10,000 (Hz)

6) Output number of rotating direction signal (d2)
Operation is as follows by judging the difference between the output pulse frequency (target position) and present position.
[+ (Positive)] → (d2) : ON
[- (Negative)] → (d2) : OFF

7.15 Real Time Clock Control

7.15.1 TCMP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

The comparison time and the time data are compared, and the bit device is turned ON or OFF depending on the magnitude of difference.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
TCMP	16 bits	Continuous	TCMP EN ENO s1 d s2 s3 s	TCMP(EN,s1,s2,s3,s,d);
TCMPP	16 bits	Pulse	TCMPP EN ENO s1 d s2 s3 s	TCMPP(EN,s1,s2,s3,s,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Comparison time "hour" (Setting range: 0 to 23)	ANY16
	s2	Comparison time "minute" (Setting range: 0 to 59)	ANY16
	s3	Comparison time "second" (Setting range: 0 to 59)	ANY16
	s	Time data (hour, minute, second) "hour" (3 points occupied)	ANY16
Output variable	ENO	Execution state	Bit
	d	Device to be turned ON or OFF depending on the comparison result. (3 points occupied)	Bit

3. Applicable devices

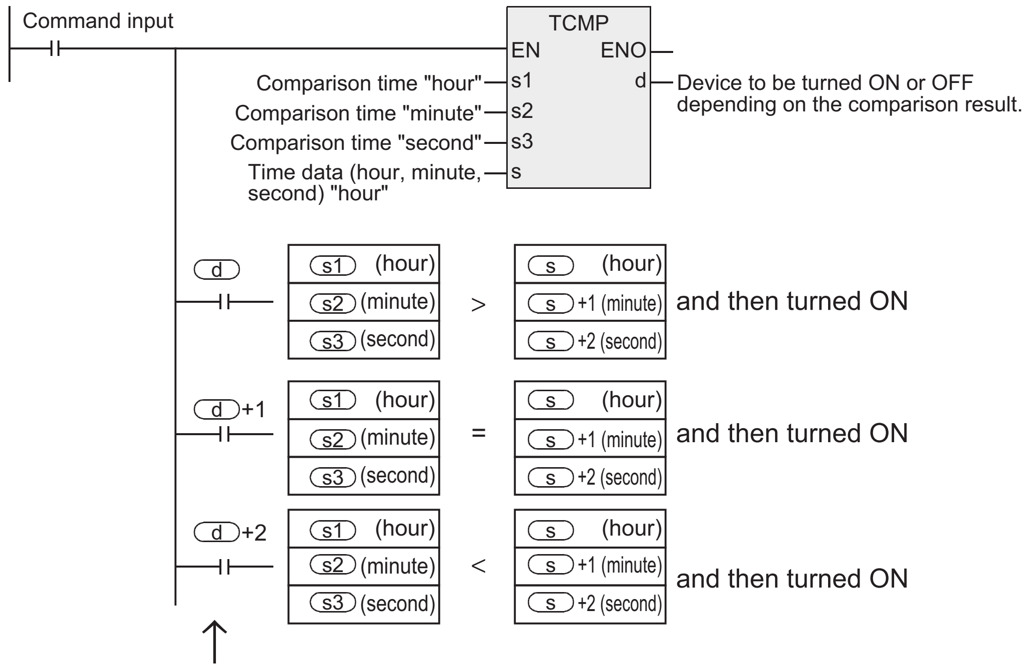
Operand type	Bit Devices							Word Devices													Others					
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
s2								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
s3								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
s												●	●	●	▲2	▲3			●					
d		●	●			●	▲1												●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (TCMP)

The time of comparison time (hour, minute, second) (s1), (s2), (s3) is compared with the time data (hour, minute, second) of the device specified by (s), and the device specified by (d) is turned ON or OFF depending on the magnitude of difference.



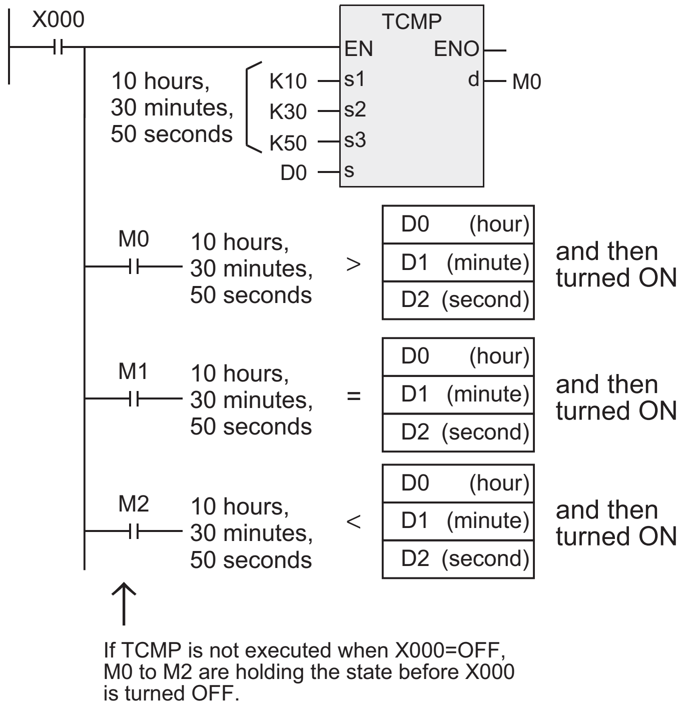
If TCMP is not executed when command contact is changed from ON to OFF, (d), (d)+1, (d)+2 are holding the state before the command contact is turned OFF.

Cautions

1) Number of devices occupied
   Three devices are occupied each in (s) and (d).
   Be careful not to overlap with the devices used in machine control.
2) When using the time (hour, minute, second) of the clock data of the real time clock built in the PLC, first read out the value of the special data register by using TRD, TRDP, and specify the word device in each operand.
3) Applicable devices are limited.
   ▲1: FX3U, FX3UC PLCs only are applicable, index (V, Z) decoration is disabled.
   ▲2: FX3U, FX3UC, FX3G PLCs only are applicable.
   ▲3: FX3U, FX3UC PLCs only are applicable.

Program example

[Structured ladder]



[ST]

```
M0:=TCMP(X000,K10,K30,K50,D0);
M1:=TCMP(X000,K10,K30,K50,D0);
M2:=TCMP(X000,K10,K30,K50,D0);
```

(s1) :Comparison time "hour" is specified.
(s2) :Comparison time "minute" is specified.
(s3) :Comparison time "second" is specified.

(s) :Time data "hour" is specified.
(s)+1 :Time data "minute" is specified.
(s)+2 :Time data "second" is specified.

(d) , (d)+1, (d)+2
:Three bit devices are turned ON or OFF depending on the comparison result.

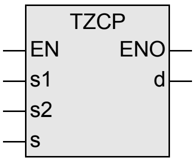
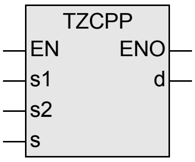
7.15.2 TZCP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

The comparison time of higher and lower points and the time data are compared, and the bit device is turned ON or OFF depending on the magnitude of difference.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
TZCP	16 bits	Continuous		TZCP(EN,s1,s2,s,d);
TZCPP	16 bits	Pulse		TZCPP(EN,s1,s2,s,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Comparison lower limit time (hour, minute, second) "hour" (3 points occupied)	ANY16
	s2	Comparison higher limit time (hour, minute, second) "hour" (3 points occupied)	ANY16
	s	Time data (hour, minute, second) "hour" (3 points occupied)	ANY16
Output variable	ENO	Execution state	Bit
	d	Device to be turned ON or OFF depending on the comparison result. (3 points occupied)	Bit

3. Applicable devices

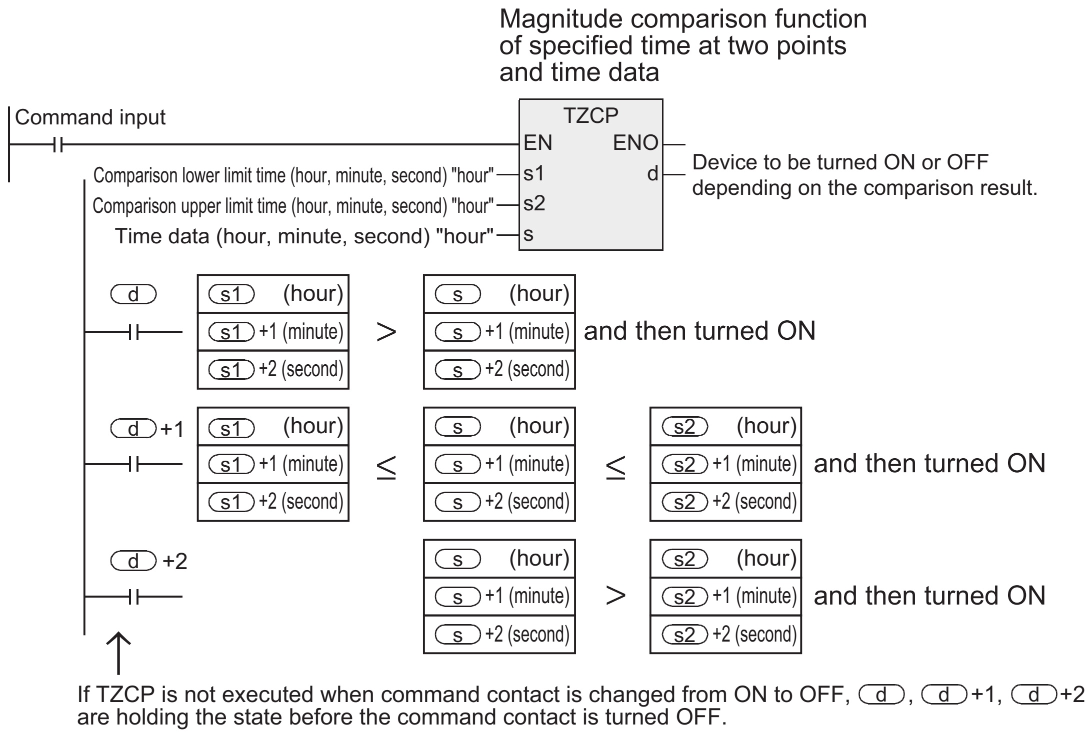
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1												●	●	●	▲2	▲3			●					
s2												●	●	●	▲2	▲3			●					
s												●	●	●	▲2	▲3			●					
d		●	●			●	▲1												●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (TZCP, TZCPP)

The comparison time (hour, minute, second) of higher and lower points and the time data (hour, minute, second) of the device specified by (s) are compared, and the device specified by (d) is turned ON or OFF depending on the magnitude of difference.

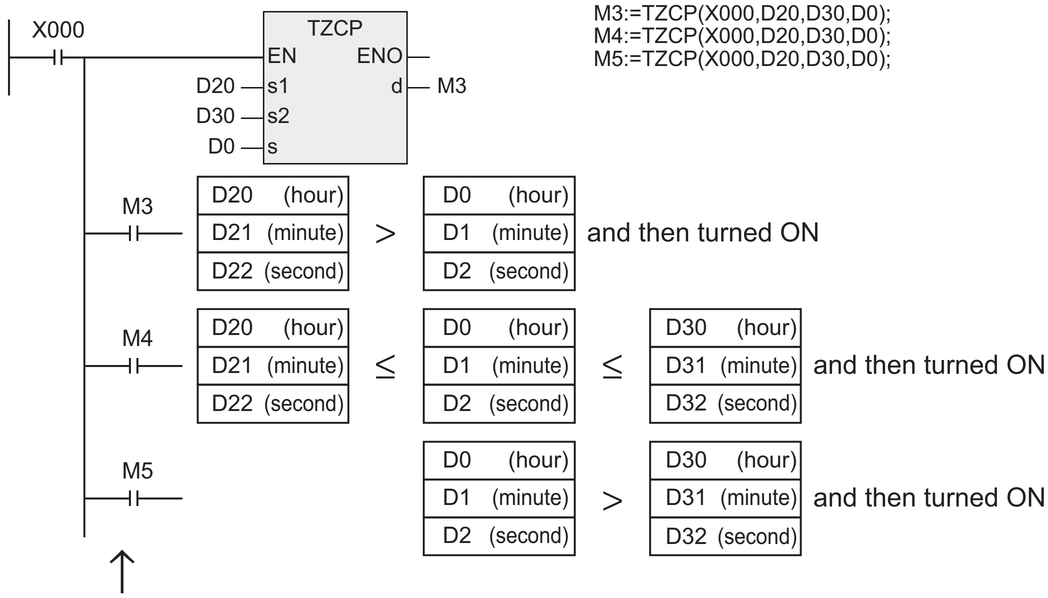


Cautions

1) Number of devices occupied
   Three devices are occupied each in (s1), (s2), (s3) and (d).
   Be careful not to overlap with the devices used in machine control.
2) When using the time (hour, minute, second) of the clock data of the real time clock built in the PLC, first read out the value of the special data register by using TRD, TRDP, and specify the word device in each operand.
3) Applicable devices are limited.
   ▲1: FX3U, FX3UC PLCs only are applicable, index (V, Z) decoration is disabled.
   ▲2: FX3U, FX3UC, FX3G PLCs only are applicable.
   ▲3: FX3U, FX3UC PLCs only are applicable.

Program example

[Structured ladder]



If TZCP instruction is not executed when X000 is changed from ON to OFF,
M3 to M5 are holding the state before X000 is turned OFF.

[ST]

```
M3:=TZCP(X000,D20,D30,D0);
M4:=TZCP(X000,D20,D30,D0);
M5:=TZCP(X000,D20,D30,D0);
```

(s1), (s1) +1, (s1) +2 :Comparison time lower limit is specified by "hour," "minute," "second."

(s2), (s2) +1, (s2) +2 :Comparison time upper limit is specified by "hour," "minute," "second."

(s), (s) +1, (s) +2 :Time data is specified by "hour," "minute," "second."

(d), (d) +1, (d) +2 :Three bit devices are turned ON or OFF depending on the comparison result.

The range of "hour" is 0 to 23.

The range of "minute" is 0 to 59.

The range of "second" is 0 to 59.

7.15.3 TADD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

Two time data are added and stored in the word device.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
TADD	16 bits	Continuous	TADD EN ENO s1 d s2	TADD(EN,s1,s2,d);
TADDP	16 bits	Pulse	TADDP EN ENO s1 d s2	TADDP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s1)	Device for storing "hour" of addition time data (hour, minute, second) (3 points occupied)	ANY16
	(s2)	Device for storing "hour" of addition time data (hour, minute, second) (3 points occupied)	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Device for storing the added result of two time data (hour, minute, second) (3 points occupied)	ANY16

3. Applicable devices

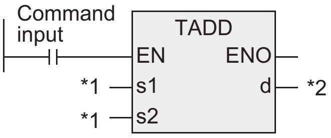
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)												●	●	●	▲1	▲2			●					
(s2)												●	●	●	▲1	▲2			●					
(d)												●	●	●	▲1	▲2			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (TADD)

Time data (hour, minute, second) of the device specified by (s1), and time data (hour, minute, second) of the device specified by (s2) are added, and the result is stored in the device specified by (d).



*1. Device for storing "hour" of addition time data (hour, minute, second)

*2. Device for storing the added result of two time data (hour, minute, second)

((s1), (s1)+1, (s1)+2)+((s2), (s2)+1, (s2)+2)

→ ((d), (d)+1, (d)+2)

(s1)	+	(s2)	→	(d)	
(s1) (hour)		(s2) (hour)		(d) (hour)	The range of "hour" is 0 to 23.
(s1)+1 (minute)		(s2)+1 (minute)		(d)+1 (minute)	The range of "minute" is 0 to 59.
(s1)+2 (second)		(s2)+2 (second)		(d)+2 (second)	The range of "second" is 0 to 59.

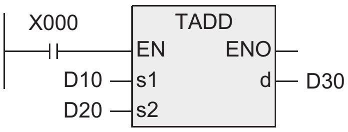
1) If the operation result exceeds 24 hours, the carry flag is turned ON, and the time by subtracting 24 hours from the simple sum is stored as the operation result.
2) If the operation result is 0 (0 hour, 0 minute, 0 second), the zero flag is turned ON.

Cautions

1) Number of devices occupied
   Three devices are occupied each in (s1), (s2) and (d).
   Be careful not to overlap with the devices used in machine control.
2) When using the time (hour, minute, second) of the clock data of the real time clock built in the PLC, first read out the value of the special data register by using TRD, TRDP, and specify the word device in each operand.
3) Applicable devices are limited.
   ▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
   ▲2: FX3U, FX3UC PLCs only are applicable.

Program example

[Structured ladder]



(D10,D11,D12)+(D20,D21,D22)
→ (D30,D31,D32)

[ST]

```
TADD(X000,D10,D20,D30);
```

(s1)	+	(s2)	→	(d)
(s1) 10 (hour)		(s2) 3 (hour)		(d) 13 (hour)
(s1)+1 30 (minute)		(s2)+1 10 (minute)		(d)+1 40 (minute)
(s1)+2 10 (second)		(s2)+2 5 (second)		(d)+2 15 (second)
10 hours, 30 minutes, 10 seconds		3 hours, 10 minutes, 5 seconds		13 hours, 40 minutes, 15 seconds

If the operation result exceeds 24 hours

(s1)	+	(s2)	→	(d)	
18 (hour)		10 (hour)		4 (hour)	← 18+10=28(≥24)
10 (minute)		20 (minute)		30 (minute)	
30 (second)		5 (second)		35 (second)	
18 hours, 10 minutes, 30 seconds		10 hours, 20 minutes, 5 seconds		4 hours, 30 minutes, 35 seconds	

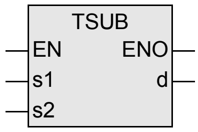
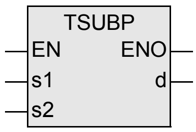
7.15.4 TSUB

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

Two time data are subtracted and stored in the word device.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
TSUB	16 bits	Continuous		TSUB(EN,s1,s2,d);
TSUBP	16 bits	Pulse		TSUBP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	"Hour" of subtraction time data (hour, minute, second) (3 points occupied)	ANY16
	s2	"Hour" of subtraction time data (hour, minute, second) (3 points occupied)	ANY16
Output variable	ENO	Execution state	Bit
	d	The subtracting result of two time data (hour, minute, second) is stored (3 points occupied)	ANY16

3. Applicable devices

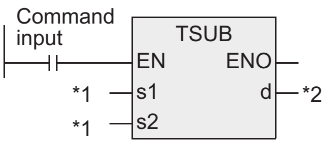
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1												●	●	●	▲1	▲2			●					
s2												●	●	●	▲1	▲2			●					
d												●	●	●	▲1	▲2			●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (TSUB, TSUBP)

From the time data (hour, minute, second) of the device specified by s1, the time data (hour, minute, second) of the device specified by s2 is subtracted, and the result is stored in the device specified by d.



*1. Device for storing "hour" of subtraction time data (hour, minute, second)

*2. Device for storing the subtracted result of two time data (hour, minute, second)

(s1, s1+1, s1+2) - (s2, s2+1, s2+2)

→(d, d+1, d+2)

s1		s2		d	
s1 (hour)	-	s2 (hour)	→	d (hour)	The range of "hour" is 0 to 23.
s1+1 (minute)		s2+1 (minute)		d+1 (minute)	The range of "minute" is 0 to 59.
s1+2 (second)		s2+2 (second)		d+2 (second)	The range of "second" is 0 to 59.

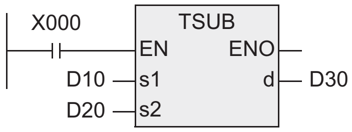
If the operation result is less than 0 hour, the borrow flag is turned ON, and the time by adding 24 hours to the simple difference is stored as the operation result. If the operation result is 0 (0 hour, 0 minute, 0 second), the zero flag is turned ON.

Cautions

1) Number of devices occupied
Three devices are occupied each in s1, s2 and d.
Be careful not to overlap with the devices used in machine control.
2) When using the time (hour, minute, second) of the clock data of the real time clock built in the PLC, first read out the value of the special data register by using TRD, TRDP, and specify the word device in each operand.
3) Applicable devices are limited.
▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

Program example

[Structured ladder]



(D10,D11,D12) - (D20,D21,D22)
→(D30,D31,D32)

[ST]

```
TSUB(X000,D10,D20,D30);
```

D 10	10 (hour)	-	D 20	3 (hour)	→	D 30	7 (hour)
D 11	30 (minute)		D 21	10 (minute)		D 31	20 (minute)
D 12	10 (second)		D 22	5 (second)		D 32	5 (second)

10 hours, 30 minutes, 10 seconds   3 hours, 10 minutes, 5 seconds   7 hours, 20 minutes, 5 seconds

If the operation result is less than 0 hour

s1		s2		d	
5 (hour)	-	18 (hour)	→	11 (hour)	← 5-18=-13(<0)
20 (minute)		10 (minute)		10 (minute)	
40 (second)		5 (second)		35 (second)	

5 hours, 20 minutes, 40 seconds   18 hours, 10 minutes, 5 seconds   11 hours, 10 minutes, 35 seconds

7.15.5 HTOS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction converts the "hour, minute, second" unit (time) data into second unit data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
HTOS	16 bits	Continuous	HTOS EN ENO s d	HTOS(EN,s,d);
HTOSP	16 bits	Pulse	HTOSP EN ENO s d	HTOSP(EN,s,d);
DHTOS	32 bits	Continuous	DHTOS EN ENO s d	DHTOS(EN,s,d);
DHTOSP	32 bits	Pulse	DHTOSP EN ENO s d	DHTOSP(EN,s,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Head device for storing the time data (hour, minute, second) before conversion	ANY16	
Output variable	ENO	Execution state	Bit	
	d	Device for storing the time data (hour, minute, second) after conversion	ANY16	ANY32

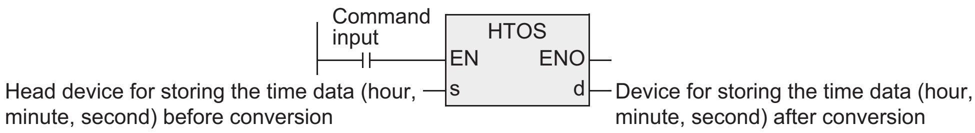
3. Applicable devices

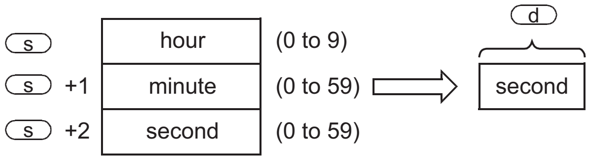
Operand type	Bit Devices: System user							Word Devices: Digit designation				System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●			●					

Function and operation explanation

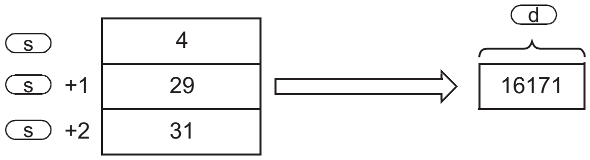
1. 16-bit operation (HTOS/HTOSP)

The time data (hour, minute, second) of the device specified by (s) is converted into the second unit, and the result is stored in the device specified by (d).



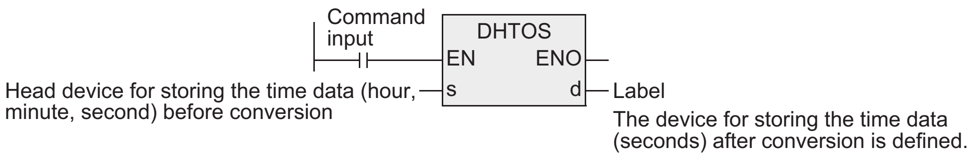


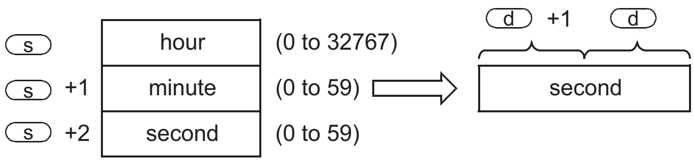
For example, when 4 hours, 29 minutes, 31 seconds is specified, the operation is as follows.



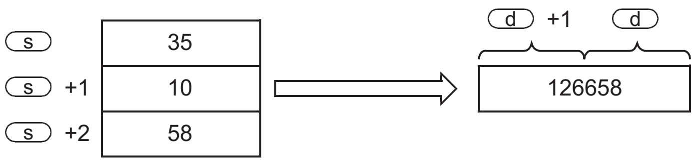
2. 32-bit operation (DHTOS/DHTOSP)

The time data (hour, minute, second) of the device specified by (s) is converted into the second unit, and the result is stored in the device specified by (d)





For example, when 35 hours, 10 minutes, 58 seconds is specified, the operation is as follows.



Cautions

1) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
When specifying the device, use the global label.

Error

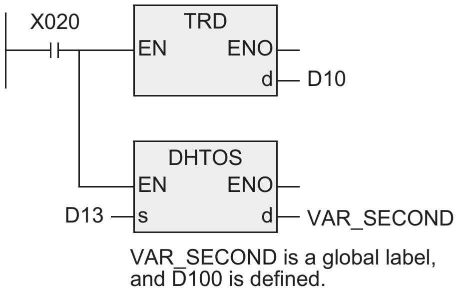
In the following case, it is an operation error, and the error flag (M8067) is turned ON, and the error code is stored in D8067.

1) When the data of the device specified by (s) is out of the range. (Error code: K6706)

Program example

This is a program for converting the time data being read out from the real-time clock built in the PLC, when the X020 is ON, into the second unit, and storing in D100, D101.

[Structured ladder]

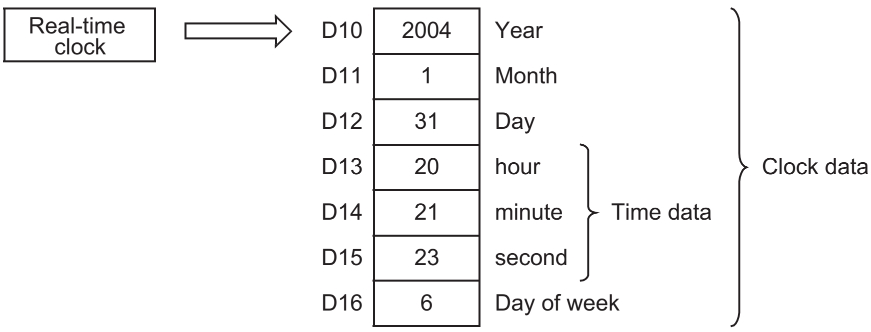


[ST]

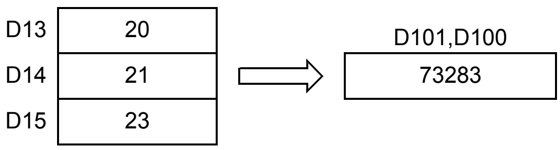
```
D10:=TRD(X020);
VAR_SECOND:=DHTOS(X020,D13);
```

Operation

1) Reading operation of clock data by TRD



2) Conversion operation to seconds by DHTOS



7.15.6 STOH

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction converts the time data in second unit into time data in "hour, minute, second" unit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
STOH	16 bits	Continuous	STOH EN ENO s d	STOH(EN,s,d);
STOHP	16 bits	Pulse	STOHP EN ENO s d	STOHP(EN,s,d);
DSTOH	32 bits	Continuous	DSTOH EN ENO s d	DSTOH(EN,s,d);
DSTOHP	32 bits	Pulse	DSTOHP EN ENO s d	DSTOHP(EN,s,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Device for storing the time data in second unit before conversion	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Device for storing the time data in "hour, minute, second" unit after conversion	ANY16	

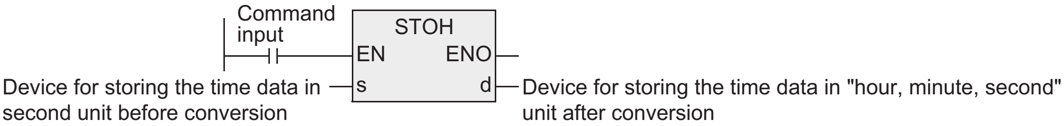
3. Applicable devices

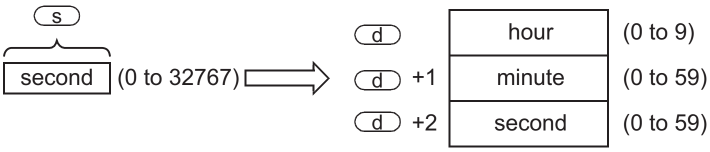
Operand type	Bit Devices: System user							Word Devices: Digit designation				System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	●	●			●					
(d)									●	●	●	●	●	●	●	●			●					

Function and operation explanation

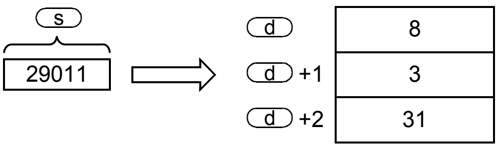
1. 16-bit operation

The second unit data of the device specified by (s) is converted into "hour, minute, second" unit, and the result is stored in the device specified by (d).



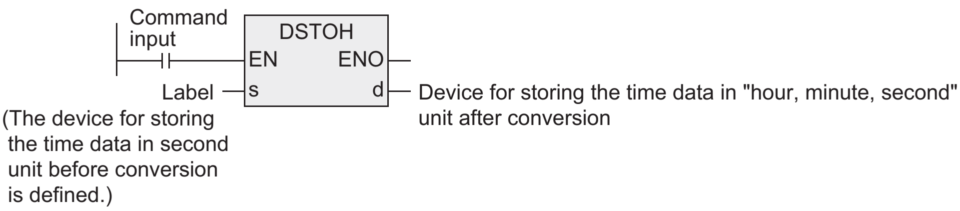


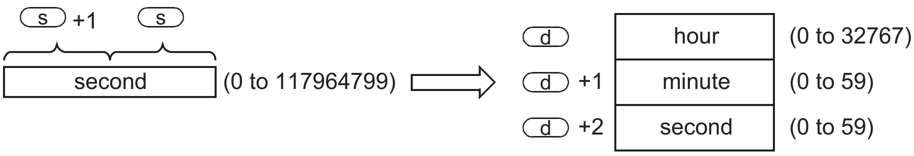
For example, when 29011 seconds is specified, the operation is as follows.



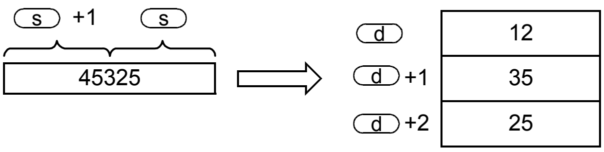
2. 32-bit operation

The second unit data of the device specified by (s) is converted into "hour, minute, second" unit, and the result is stored in the device specified by (d).





For example, when 45325 seconds is specified, the operation is as follows.



Cautions

1) When handling 32-bit data in structured program, you cannot designate 16-bit device directly unlike the simple project.
When handling 32-bit data, please use the label.
However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
When specifying the device, use the global label.

Error

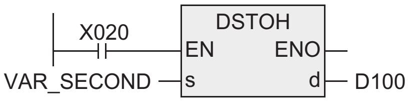
In the following case, it is an operation error, and the error flag (M8067) is turned ON, and the error code is stored in D8067.

1) When the data of the device specified by (s) is out of the range. (Error code: K6706)

Program example

This is a program for converting the second unit data stored in D0, D1 when the X020 is turned ON, into the "hour, minute, second" unit, and storing the result in D100, D101, D102.

[Structured ladder]



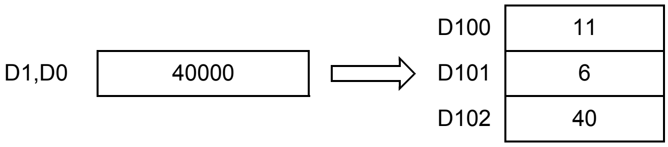
VAR_SECOND is a global label,
and D100 is defined.

[ST]

```
D100:=DSTOH(X020,VAR_SECOND)
```

Operation

1) Conversion into the "hour, minute, second" unit by STOHP instruction (when 40000 seconds is specified by D1, D0)



7.15.7 TRD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

The clock data is read out in the real-time clock built in the PLC.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
TRD	16 bits	Continuous	TRD —EN ENO— d—	TRD(EN,d);
TRDP	16 bits	Pulse	TRDP —EN ENO— d—	TRDP(EN,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	(d)	The reading destination of clock data and the head device are specified (7 points occupied).	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)												●	●	●	▲1	▲2			●					

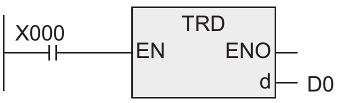
▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (TRD)

The clock data (D8013 to D8019) of the real-time clock built in the PLC is read out into the device specified by (d) in the following format.

[Structured ladder]



[ST]

```
D0:=TRD(X000);
```

This instruction reads out the real-time clock data of the PLC into seven data registers.

	Device	Item	Clock data		Device	Item
Special data register	D8018	Year (solar calendar)	0 to 99 (Lower 2 digits of year)	→	D 0	Year (solar calendar)
	D8017	Month	1 to 12	→	D 1	Month
	D8016	Day	1 to 31	→	D 2	Day
	D8015	hour	0 to 23	→	D 3	hour
	D8014	minute	0 to 59	→	D 4	minute
	D8013	second	0 to 59	→	D 5	second
	D8019	Day of week	0 (Sunday) to 6 (Saturday)	→	D 6	Day of week

Cautions

1. Number of devices occupied

Seven devices specified by (d) are occupied.
Be careful not to overlap with the devices used in machine control.

2. Applicable devices are limited.

▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

7.15.8 TWR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

The clock data is written into the real-time clock built in the PLC.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
TWR	16 bits	Continuous	TWR EN ENO s	TWR(EN,s);
TWRP	16 bits	Pulse	TWRP EN ENO s	TWRP(EN,s);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	The writing source of clock data and the head device are specified (7 points occupied).	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

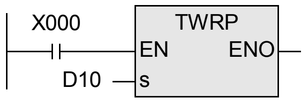
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s												●	●	●	▲1	▲2			●					

▲: Refer to "Cautions".

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

The setting clock data stored in the device specified by (s) is written into the clock data (D8013 to D8019) of the real-time clock built in the PLC.



1) D8018 (year data) can be also changed over to four-digit mode. (See Program example.)

	Device	Item	Clock data		Device	Item	
Time setting data	D 10	Year (solar calendar)	0 to 99 (Lower 2 digits of year)	→	D8018	Year (solar calendar)	Special data register
	D 11	Month	1 to 12	→	D8017	Month	
	D 12	Day	1 to 31	→	D8016	Day	
	D 13	hour	0 to 23	→	D8015	hour	
	D 14	minute	0 to 59	→	D8014	minute	
	D 15	second	0 to 59	→	D8013	second	
	D 16	Day of week	0 (Sunday) to 6 (Saturday)	→	D8019	Day of week	

2) When TWR or TWRP is executed, the clock data of the real-time clock is changed immediately. Therefore, in the device to be specified by (s), you are advised to transfer the clock data of several minutes ahead, and execute the instruction when reaching the exact time.
3) When setting the clock data (time setting) by this instruction, you are not required to control the special auxiliary relay M8015 (time stopping and time setting).
4) The day of week of FX1S, FX1N, FX1NC PLCs is set automatically depending on the content of the date regardless of the written numeric value.
5) If a numeric value of non-existing date is entered, the clock data is not changed. Enter correct clock data, and write again.

Cautions

1. Number of devices occupied

The device specified by (s) occupies the subsequent seven devices.
Be careful not to overlap with the devices used in machine control.

2. Applicable devices are limited.

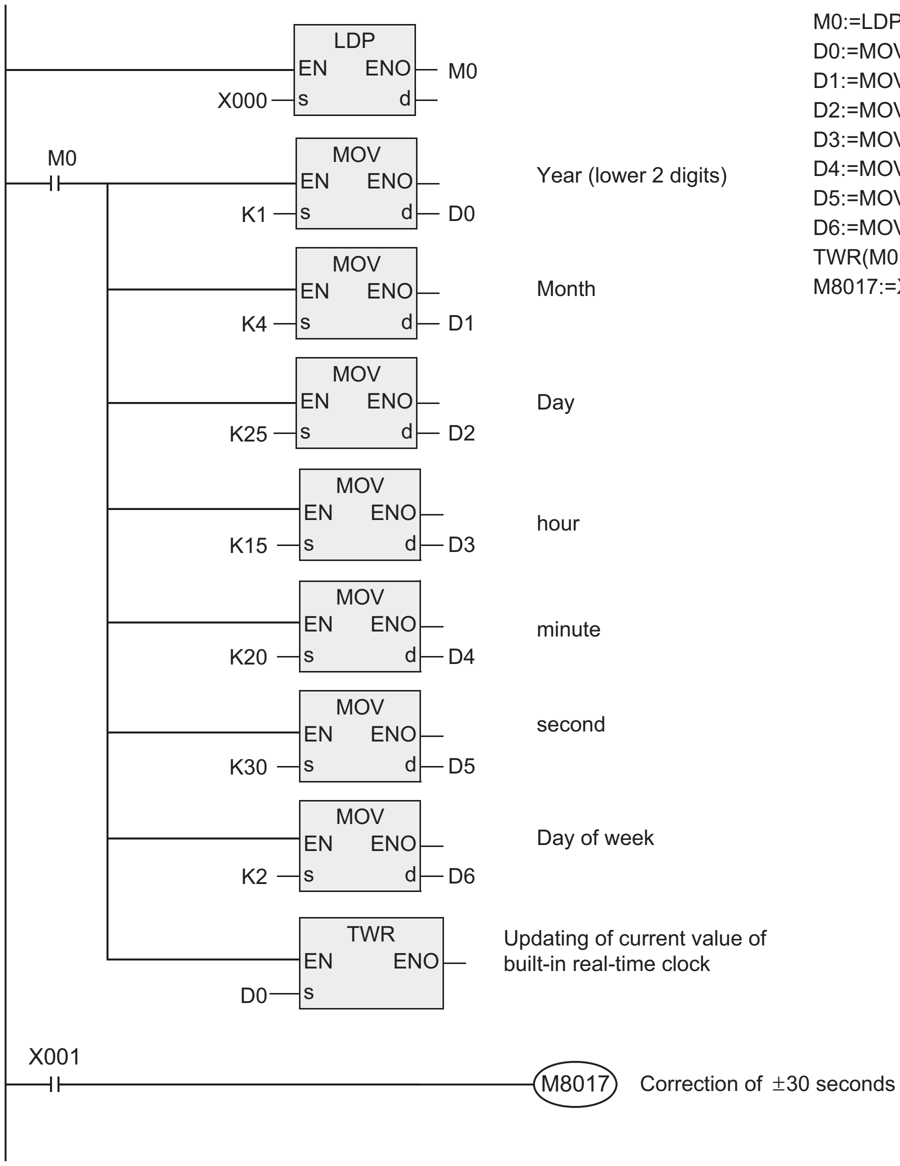
▲1: FX3U, FX3UC, FX3G PLCs only are applicable.
▲2: FX3U, FX3UC PLCs only are applicable.

Program example

1. Setting example of clock data (time)

To set the real-time clock. In the case of 15 hours, 20 minutes, 30 seconds, Tuesday, April 25, 2001.

[Structured ladder]

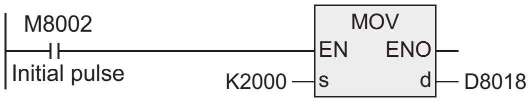


[ST]

```
M0:=LDP(TRUE,X000);
D0:=MOV(M0,K1);
D1:=MOV(M0,K4);
D2:=MOV(M0,K25);
D3:=MOV(M0,K15);
D4:=MOV(M0,K20);
D5:=MOV(M0,K30);
D6:=MOV(M0,K2);
TWR(M0,D0);
M8017:=X001;
```

1) When setting the time, first set the time of several minutes ahead, and turn ON the X000 when reaching the exact time, then the set time is written into the real-time clock, and the clock data is updated.
2) Every time the X001 is turned ON, ±30 seconds can be corrected.

3) When handling the year in four digits, please add the following program.
D8018 operates in four-digit year mode from the second scan after RUN of PLC.



a) Usually, the PLC operates in two-digit year mode. After RUN of PLC, by executing the above instruction, and transferring K2000 (fixed value) to D8018 (year) for one operation cycle only, the operation is changed to four-digit mode.
b) This program must be executed every time the PLC is set to RUN. By transferring K2000, only the year display is changed to four-digit mode, and the present time is not changed.
c) In the case of four-digit mode of the year, 80 to 99 correspond to 1980 to 1999, and 00 to 79 correspond to 2000 to 2079.
Example: 80=1980, 99=1999, 00=2000, 79=2079

4) When connecting with the data access unit of FX-10DU, FX-20DU, FX-25DU types, please set the year in two-digit mode. If set in four-digit mode, the year is not displayed correctly in the present versions of these DU types.
When the PLC is in four-digit mode, if the clock is set from FX-10DU, 20DU, 25DU, it must be noted that the mode is changed to two-digit mode.

7.15.9 HOUR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	△	○	○	×	×	×

Outline

This instruction adds and measures the ON time duration of input contact in one-hour unit.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
HOUR	16 bits	Continuous	HOUR (EN, s → ENO, d1, d2)	HOUR(EN,s,d1,d2);
DHOUR	32 bits	Continuous	DHOUR (EN, s → ENO, d1, d2)	DHOUR(EN,s,d1,d2);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	ON time duration of d2 (set in one-hour unit)	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d1)	Present value of one-hour unit (data register for power failure hold is specified)	ARRAY [1..2] OF ANY16	ARRAY [1..3] OF ANY16
	(d2)	Device of alarm output destination	Bit	

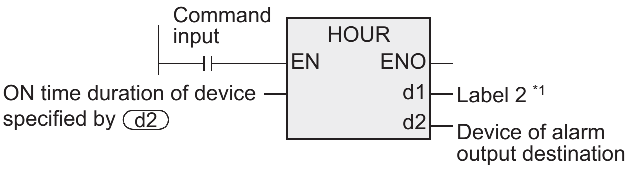
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit designation				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲2	▲3	●	●	●	●	●			
(d1)														●	▲2				●					
(d2)		●	●			●	▲1												●					

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation



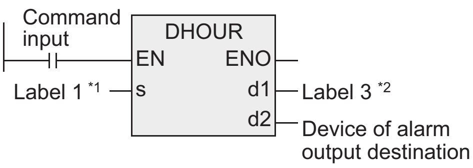
*1. In label 2, the device for storing the present value of one-hour unit is defined.

When the cumulative total of ON time duration of command input exceeds the time of the device specified by s, the device specified by d2 is turned ON. In d1+1, the present value of less than one hour is stored in one-second unit.

s	:	The time until d2 is turned ON is specified in one-hour unit.
d1	:	Present value of one-hour unit
d1+1	:	Present value of less than one hour (one-second unit)
d2	:	Device of alarm output destination To be turned ON when the present value d1 exceeds the specified time of s.

1) To use the present value data continuously after the PLC power source is turned OFF, please specify the data register for power failure hold in the device specified by d1.
When the data register for general purpose is used, the present value data is cleared when the PLC power source is turned OFF, or when changed from STOP to RUN.
2) Measurement continues even after the alarm output (device specified by d2) is turned ON.
3) Measurement stops when the present value of the device specified by d1 reaches the maximum value of 16-bit figure. When desired to measure continuously, please clear the present values of d1 to d1+1.

2. 32-bit operation



*1. In label 1, the ON time of the device specified by d2 is defined.

*2. In label 3, the device for storing the present value of one-hour unit is defined.

[s+1, s]	:	Time setting until d2 is turned ON Specify by s1+1 (upper digit), s1 (lower digit).
[d1+1, d1]	:	Present value of one-hour unit Store in d1+1 (upper digit), d1 (lower digit).
d1+2	:	Present value of less than one hour (one-second unit)
d2	:	Specification of alarm output To be turned ON when the present value d1, d1+1 exceeds the time specified by s.

1) To use the present value data continuously after the PLC power source is turned OFF, please specify the data register for power failure hold in the device specified by d1.
When the data register for general purpose is used, the present value data is cleared when the PLC power source is turned OFF, or when changed from STOP to RUN.
2) Measurement continues even after the alarm output (device specified by d2) is turned ON.
3) Measurement stops when the present value of the device specified by d1 reaches the maximum value of 32-bit figure. When desired to measure continuously, please clear the present values of d1 to d1+2.

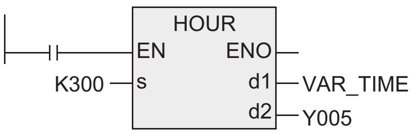
Cautions

1) When handling array data or 32-bit data in structured program, you cannot specify 16-bit device directly unlike the simple project.
   When handling array data or 32-bit data, please use the label.
   However, the 32-bit counter is a 32-bit long device, and 32-bit data can be specified directly.
   When specifying the device, use the global label.
2) FX2N, FX2NC PLCs are supporting the instruction at V3.00 or later.
3) Number of devices occupied
   The device specified by (d1) occupies two devices (16-bit operation) or three devices (32-bit operation).
4) Be careful not to overlap with the devices used in machine control.
   ▲1: FX3U, FX3UC PLCs only are applicable, index (V, Z) decoration is disabled.
   ▲2: FX3U, FX3UC, FX3G PLCs only are applicable.
   ▲3: FX3U, FX3UC PLCs only are applicable.

Program example

When the cumulative total of ON time duration of X000 exceeds 300 hours, Y005 is turned ON.
In D201, the present time of less than one hour is stored in one-second unit.

[Structured ladder]



VAR_TIME is a global label,
and D200 is defined.

[ST]

```
VAR_TIME:=HOUR(X000,K300);
Y005:=HOUR(X000,K300);
```

(s)	:	Time setting for turning ON (d2) Specified in one-hour unit.
(d1)	:	Present value of one-hour unit
(d1) +1	:	Present value of less than one hour (one-second unit)
(d2)	:	Specification of alarm output To be turned ON when the present value of the device specified by (d1) exceeds the specified time of the device specified by (s). (In this example, to be turned ON when reaching 300 hours + 1 second.)

7.16 External Device

7.16.1 GRY

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	×	×	×

Outline

This instruction converts a binary value into a gray code, and transfers it.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
GRY	16 bits	Continuous	GRY EN ENO s d	GRY(EN,s,d);
GRYP	16 bits	Pulse	GRYP EN ENO s d	GRYP(EN,s,d);
DGRY	32 bits	Continuous	DGRY EN ENO s d	DGRY(EN,s,d);
DGRYP	32 bits	Pulse	DGRYP EN ENO s d	DGRYP(EN,s,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Conversion source data or word device storing conversion source data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Word device storing data after conversion	ANY16	ANY32

3. Applicable devices

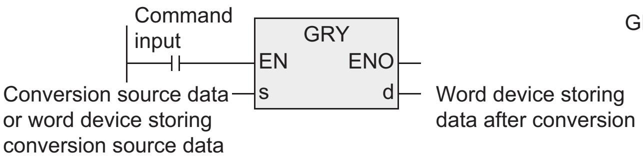
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(d)									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions"

Function and operation explanation

1. 16-bit operation (GRY, GRYP)

[Structured ladder]

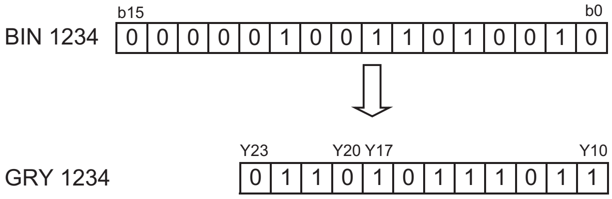


[ST]

GRY(EN,s,d);

This instruction converts and transfers data from the source (binary) to the destination (gray code).

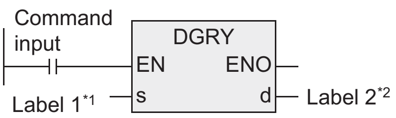
When the data specified by (s) is K1234 and the device specified by (d) is K3Y10



1) The device specified by (s) can store a value from 0 to 32,767.

2. 32-bit operation (DGRY, DGRYP)

[Structured ladder]



[ST]

DGRY(EN,s,d);

*1. Label 1 is defined as the conversion source data or the device that stores the conversion source data.

*2. Label 2 is defined as the device that stores the data after conversion.

1) A binary value can be converted into a gray code of up to 32 bits.
2) The device specified by (s) can store a value from 0 to 2,147,483,647.

Cautions

1) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project.
   Use a label to handle array data or 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) The data conversion speed depends on the scan time of the PLC.
3) Some restrictions to applicable devices
   ▲1: Applicable to the FX3U, FX3UC and FX3G PLCs only.
   ▲2: Applicable to the FX3U and FX3UC PLCs only.

7.16.2 GBIN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	×	×	×	×	×

Outline

This instruction converts a gray code into a binary value, and transfers it.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
GBIN	16 bits	Continuous	GBIN EN ENO s d	GBIN(EN,s,d);
GBINP	16 bits	Pulse	GBINP EN ENO s d	GBINP(EN,s,d);
DGBIN	32 bits	Continuous	DGBIN EN ENO s d	DGBIN(EN,s,d);
DGBINP	32 bits	Pulse	DGBINP EN ENO s d	DGBINP(EN,s,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Conversion source data or word device storing conversion source data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Word device storing data after conversion	ANY16	ANY32

3. Applicable devices

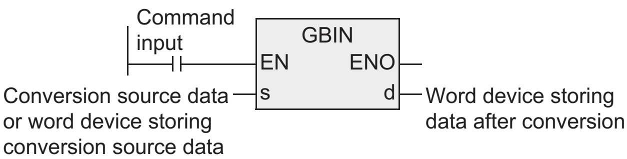
Operand type	Bit Devices							Word Devices													Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
d									●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions"

Function and operation explanation

1. 16-bit operation (GBIN, GBINP)

[Structured ladder]

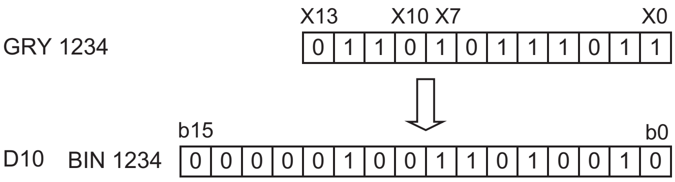


[ST]

GBIN(EN,s,d);

This instruction converts and transfers data from the source (gray code) to the destination (binary).

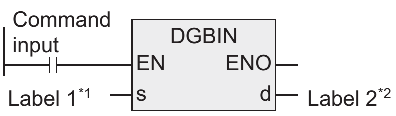
When the device specified by (s) is K3X000 and the device specified by (d) is D10



1) This instruction can be used for detecting an absolute position by a gray code type encoder.
2) The device specified by (s) can store a value from 0 to 32,767.

2. 32-bit operation (DGBIN, DGBINP)

[Structured ladder]



[ST]

DGBIN(EN,s,d);

*1. Label 1 is defined as the conversion source data or the device that stores the conversion source data.

*2. Label 2 is defined as the device that stores the data after conversion.

1) A gray code can be converted into a binary value of up to 32 bits.
2) The device specified by (s) can store a value from 0 to 2,147,483,647.

Cautions

1) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project.
Use a label to handle array data or 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
2) When an input relay (X) is specified as (s), the response delay will be "Scan time of PLC + Input filter constant."
The input filter value can be changed in X000 to X017 using REFF, REFFP or D8020 (filter adjustment) so that the filter constant delay is eliminated.
*1. The FX3G and FX2N-16M PLCs use X000 to X007.
3) Some restrictions to applicable devices
▲1: Applicable to the FX3U, FX3UC and FX3G PLCs only.
▲2: Applicable to the FX3U and FX3UC PLCs only.

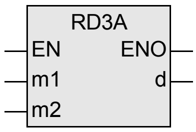
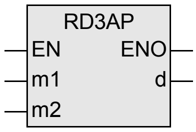
7.16.3 RD3A

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	△	○	×	×	×	×

Outline

This instruction reads an analog input value from the analog block FX0N-3A or FX2N-2AD.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RD3A	16 bits	Continuous		RD3A(EN,m1,m2,d);
RD3AP	16 bits	Pulse		RD3AP(EN,m1,m2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	m1	Special block number • FX1N, FX1NC, FX2N, FX2NC, FX3U and FX3UC series PLCs: K0 to K7 • FX3UC-32MT-LT (-2) series PLC: K1 to K7	ANY16
	m2	Analog input channel	ANY16
Output variable	ENO	Execution state	Bit
	d	Word device storing the read data	ANY16

3. Applicable devices

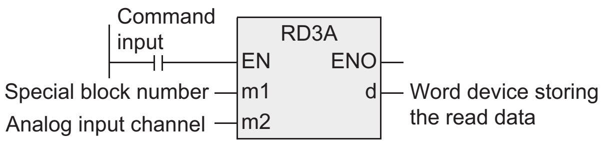
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
m1								●	●	●	●	●	●	●	▲1		●	●	●	●	●			
m2								●	●	●	●	●	●	●	▲1		●	●	●	●	●			
d									●	●	●	●	●	●	▲1		●	●	●					

▲: Refer to "Cautions"

Function and operation explanation

1. 16-bit operation (RD3A)

[Structured ladder]



<For FX3U and FX3UC PLCs>

This instruction reads an analog input value from the analog block FX0N-3A or FX2N-2AD.

(m1) : Special block number
FX3U and FX3UC (D, DSS) series PLCs :K0 to K7
FX3UC-32MT-LT(-2) :K1 to K7 (K0 indicates the built-in CC-Link/LT master.)

(m2) : Analog input channel number
FX0N-3A : K1(ch1), K2(ch2)
FX2N-2A : K21(ch1), K22(ch2)

(d) : Read data
A value read from the analog block is stored.
FX0N-3A : 0 to 255 (8 bits)
FX2N-2AD : 0 to 4095 (12 bits)

<For FX1N and FX1NC PLCs>

This instruction reads an analog input value from the analog block FX0N-3A.

(m1) : Special block number
K0 to K7

(m2) : Analog input channel number
K1 or K2

(d) : Read data
A value read from the analog block is stored.
FX0N-3A : 0 to 255 (8 bits)

<For FX2N and FX2NC PLCs>

This instruction reads an analog input value from the analog block FX0N-3A or FX2N-2AD.

(m1) : Special block number
K0 to K7

(m2) : Analog input channel number
FX0N-3A : K1(ch1), K2(ch2)
FX2N-2AD : K21(ch1), K22(ch2)

(d) : Read data
A value read from the analog block is stored.
FX0N-3A : 0 to 255 (8 bits)
FX2N-2AD : 0 to 4095(12 bits)

Cautions

1) The FX2N and FX2NC PLCs of V3.00 or later support RD3A instruction.
2) Some restrictions to applicable devices
▲1: Applicable to the FX3U and FX3UC PLCs only.

7.16.4 WR3A

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	△	○	×	×	×	×

Outline

This instruction writes a digital value to the analog block FX0N-3A and FX2N-2DA.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
WR3A	16 bits	Continuous	WR3A EN ENO m1 m2 s	WR3A(EN,m1,m2,s);
WR3AP	16 bits	Pulse	WR3AP EN ENO m1 m2 s	WR3AP(EN,m1,m2,s);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(m1)	Special block number • FX1N, FX1NC, FX2N, FX2NC, FX3U and FX3UC series PLCs: K0 to K7 • FX3UC-32MT-LT (-2) series PLC: K1 to K7	ANY16
	(m2)	Analog output channel	ANY16
	(s)	Data to be written or word device storing data to be written.	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

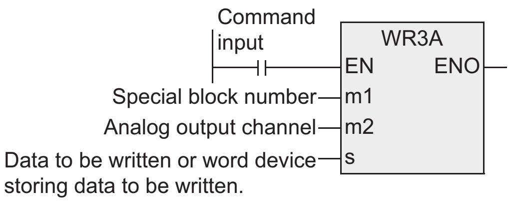
Operand type	Bit Devices							Word Devices													Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(m1)								●	●	●	●	●	●	●	▲1		●	●	●	●	●			
(m2)								●	●	●	●	●	●	●	▲1		●	●	●	●	●			
(s)									●	●	●	●	●	●	▲1		●	●	●					

▲: Refer to "Cautions"

Function and operation explanation

1. 16-bit operation (WR3A, WR3AP)

[Structured ladder]



[ST]

```
WR3A(EN,m1,m2,s);
```

<For FX3U and FX3UC PLCs>

This instruction writes a digital value to the analog block FX0N-3A or FX2N-2DA.

(m1) : Special block number
FX3U and FX3UC (D, DSS) series PLCs: :K0 to K7
FX3UC-32MT-LT(-2) :K1 to K7 (K0 indicates the built-in CC-Link/LT master.)

(m2) : Analog output channel number
FX0N-3A : K1(ch1)
FX2N-2DA : K21(ch1), K22(ch2)

(s) : Data to be written
This specifies the value to be written to the analog block.
FX0N-3A : 0 to 255 (8 bits)
FX2N-2DA : 0 to 4095 (12 bits)

<For FX1N and FX1NC PLCs>

This instruction writes a digital value to the analog block FX0N-3A.

(m1) : Special block number
K0 to K7

(m2) : Analog output channel number
Valid for K1 only

(s) : Data to be written
This specifies the value to be written to the analog block.
FX0N-3A : 0 to 255 (8 bits)

<For FX2N and FX2NC PLCs>

This instruction writes a digital value to the analog block FX0N-3A or FX2N-2DA.

(m1) : Special block number
K0 to K7

(m2) : Analog output channel number
FX0N-3A : K1(ch1)
FX2N-2AD : K21(ch1), K22(ch2)

(s) : Data to be written
This specifies the value to be written to the analog block.
FX0N-3A : 0 to 255 (8 bits)
FX2N-2DA : 0 to 4095(12 bits)

Cautions

1) The FX2N and FX2NC PLCs of V3.00 or later support RD3A instruction.
2) Some restrictions to applicable devices.
▲1: Applicable to the FX3U and FX3UC PLCs only.

7.17 Extension Function

7.17.1 EXTR_IN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	△	×	×	×	×	×

Outline

This instruction writes the operation control instructions and parameters of the memory for extension function.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
EXTR_IN	16 bits	Continuous	EXTR_IN EN ENO s sd1 sd2 sd3	EXTR_IN(EN,s,sd1,sd2,sd3);
EXTRP_IN	16 bits	Pulse	EXTRP_IN EN ENO s sd1 sd2 sd3	EXTRP_IN(EN,s,sd1,sd2,sd3);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Function number • K11: Inverter operation control instruction • K13: Writing inverter parameter	ANY16
	sd1	Inverter station number	ANY
	sd2	• When issuing inverter operation control instruction: inverter instruction code (hexadecimal) • When writing inverter parameter: Inverter parameter number (decimal)	ANY
	sd3	Value written in inverter	ANY
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s																				●	●			
sd1	●	●	●			●		●	●	●	●	●	●	●			●	●	●	●	●			
sd2	●	●	●			●		●	●	●	●	●	●	●			●	●	●	●	●			
sd3	●	●	●			●		●	●	●	●	●	●	●			●	●	●	●	●			

Function and operation explanation1 (Inverter operation control)

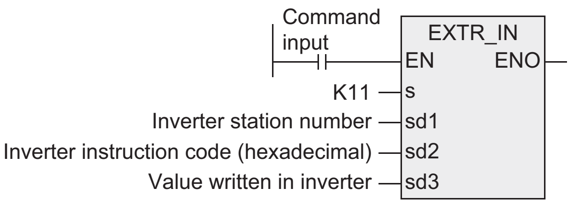
→ **For the details of the instruction, refer to Communication Control Manual.**

This instruction is for using the optional memory for extension functions.

When K11 is set to the device specified by (s), the control values necessary for inverter operation are written in the PLC.

1. 16-bit operation (EXTR_IN, EXTRP_IN)

As for the inverter*1 of the station number specified by (sd1), the control values (contents of the device specified by (sd3)) are written to the "instruction code" (contents of the device specified by (sd2))*2.



*1. General purpose inverters FREQROL-A500/E500/S500 (with communications functions) series made by Mitsubishi Electric Corporation

*2. Refer to the "Instruction code List" described later.
Also refer to the pages describing in detail the computer links from the inverter manual.

2. Inverter instruction codes

The table below shows the inverter instruction codes and their functions of the device specified by (sd2).
For the instruction codes, refer to the pages describing in detail the computer links from the inverter manual.

Instruction codes of inverter specified by (sd2) (hexadecimal)	Contents to be written	Applicable inverters		
		A500	E500	S500
HFB	Operation mode	✓	✓	✓
HF3	Special monitor selection No.	✓		
HFA	Operation command	✓	✓	✓
HEE	Writing set frequency (EEPROM)	✓	✓	✓
HED	Writing set frequency (RAM)	✓	✓	✓
HFD	Inverter reset	✓	✓	✓
HF4	Batch-clearing error contents	✓	✓	✓
HFC	Clearing all parameters	✓	✓	✓
HFC	User clear	✓		

3. Related devices

Special auxiliary relays	Function
M8104	ON when installing extended ROM cassette
M8154	Function to be defined for each EXTR instruction
M8155	Communications port being used by EXTR instruction
M8156	Communication error by EXTR instruction
M8157	Communication error by EXTR instruction (Latch)*1

Special data register	Function
D8104	Extended ROM type code
D8105	Extended ROM version
D8154	Time waiting for EXTR instruction response
D8155	Step number of instruction occupying communications port
D8156	Communication error by EXTR instruction
D8157	Communication error by EXTR instruction (Latch)*1 K1 if no error

*1. Cleared when changing from STOP to RUN.

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

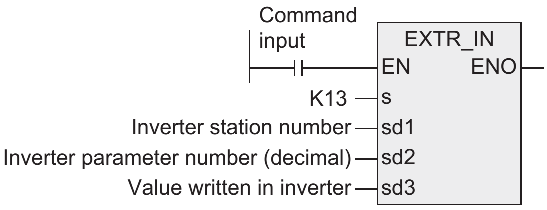
Function and operation explanation2 (Writing inverter parameters)

→ **For the details of the instruction, refer to Communication Control Manual.**

This instruction is for using the optional memory for extension functions.
When K13 is set to device specified by (s), the inverter parameter is written.

1. 16-bit operation (EXTR-IN, EXTRP-IN)

The value (contents of the device specified by (sd3)) is written to the parameter (contents of the device specified by (sd2)) of the inverter of the station number specified by (sd1).



*1. General purpose inverters FREQROL-A500/E500/S500 (with communications functions) series made by Mitsubishi Electric Corporation

2. Related devices

The same as the inverter operation control described above.

Caution

1) The FX2N and FX2NC PLCs of V3.00 or later support the EXTR_IN instruction.

7.17.2 EXTR_OUT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
×	×	△	×	×	×	×	×

Outline

This instruction is for the short mail transmission of the memory for extension function, inverter operation monitoring instruction, and parameter readout.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
EXTR_OUT	16 bits	Continuous	EXTR_OUT EN — ENO s — sd3 sd1 sd2	EXTR_OUT(EN,s,sd1,sd2,sd3);
EXTRP_OUT	16 bits	Pulse	EXTRP_OUT EN — ENO s — sd3 sd1 sd2	EXTRP_OUT(EN,s,sd1,sd2,sd3);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Function number • K0: Short mail transmission • K10: Inverter operation monitoring instruction • K12: Inverter parameter readout	ANY16
	sd1	• When transmitting short mail: Mail center, phone number of transmission destination and waiting time • When issuing inverter operation monitoring instruction and reading parameter: Inverter station number	ANY
	sd2	• When transmitting short mail: Transmission message format and message text • When issuing inverter operation monitoring instruction: inverter instruction code (hexadecimal) • When reading inverter parameter: Inverter parameter number (decimal)	ANY
Output variable	ENO	Execution state	Bit
	sd3	• When transmitting short mail: operation status • When issuing inverter operation monitoring instruction and reading parameter: Destination device storing readout value	ANY

3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit specification				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s																				●	●			
sd1	●	●	●			●		●	●	●	●	●	●	●			●	●	●	●	●			
sd2	●	●	●			●		●	●	●	●	●	●	●			●	●	●	●	●			
sd3	●	●	●			●		●	●	●	●	●	●	●			●	●	●	●	●			

Function and operation explanation1 (Transmitting short mail)

→ **For the details of the instruction, refer to Communication Control Manual.**

This instruction is for using the optional memory for extension function.

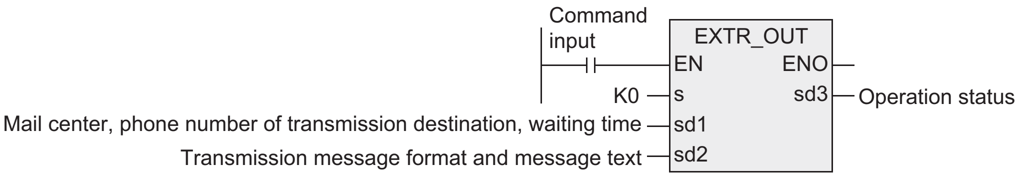
When K0 is set to the device specified by (s), the PLC transmits the short mail.

The short mail is transmitted by the PLC. The NTT DoCoMo and other firms mobile phones capable of receiving the short mail are notified.

The PLC connects to the NTT DoCoMo short mail center through the modem.

1. 16-bit operation (EXTR_OUT, EXTRP_OUT)

Message (contents of the device specified by (sd2)) is transmitted to the mail center (mail center specified by (sd1)).



2. Contents of message

The message should be as follows.

- Number of characters : Up to 50 half width characters (25 full width characters)
- Type of characters : Numbers, Kanji, Katakana, Hiragana, symbols, etc.
- Available characters : Use the character codes specified by the short mail service
- Receiving message : Received automatically.

3. Related devices

Special auxiliary relays	Function
M8104	ON when installing extended ROM cassette
M8154	Function to be defined for each EXTR instruction
M8155	Communications port being used by EXTR instruction
M8156	Communication error by EXTR instruction
M8157	Communication error by EXTR instruction (Latch)*1

Special data register	Function
D8104	Extended ROM type code
D8105	Extended ROM version
D8154	Time waiting for EXTR instruction response
D8155	Step number of instruction occupying communications port
D8156	Communication error by EXTR instruction
D8157	Communication error by EXTR instruction (Latch)*1 K1 if no error

*1. Cleared when changing from STOP to RUN.

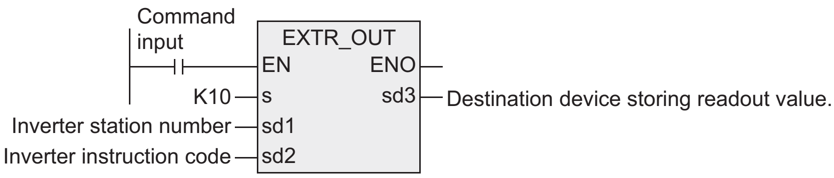
Function and operation explanation2 (Monitoring inverter operation)

→ **For the details of the instruction, refer to Communication Control Manual.**

This instruction is for using the optional memory for extension functions.
When K10 is set to the device specified by (s), the inverter operation is monitored.

1. 16-bit operation (EXTR_OUT, EXTRP_OUT)

As for the inverter*1 of the station number specified by (sd1), the operation condition of the inverter corresponding to the "instruction code"*2 (contents of the device specified by (sd2)) is read to the device specified by (sd3).



*1. General purpose inverters FREQROL-A500/E500/S500 (with communications functions) series made by Mitsubishi Electric Corporation

*2. Refer to the instruction code list described later.
Also refer to the pages describing in detail the computer links from the inverter manual.

2. Inverter instruction code

The table below shows the inverter instruction codes and their functions of the device specified by (sd2).
For the instruction codes, refer to the pages describing in detail the computer links from the inverter manual.

(sd2) Inverter instruction code (hexadecimal)	Contents to be read out	Applicable inverters		
		A500	E500	S500
H7B	Operation mode	✓	✓	✓
H6F	Output frequency [rotational speed]	✓	✓	✓
H70	Output current	✓	✓	✓
H71	Output voltage	✓	✓	
H72	Special monitor	✓		
H73	Special monitor selection number	✓		
H74	Error contents	✓	✓	✓
H75	Error contents	✓	✓	✓
H76	Error contents	✓	✓	
H77	Error contents	✓	✓	
H7A	Inverter status monitor	✓	✓	✓
H6E	Set frequency (E2PROM) readout	✓	✓	✓
H6D	Set frequency (RAM) readout	✓	✓	✓
H7F	Link parameter extension setting	Not commanded by (s2) for this instruction. Use EXTR K12 instruction to set "second parameter specification code" for automatic processing.		
H6C	Second parameter switchover			

3. Related devices

The same as the short mail transmission described above.

Function and operation explanation3 (Reading inverter parameters)

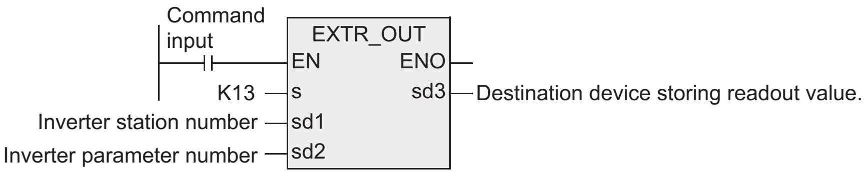
→ **For the details of the instruction, refer to Communication Control Manual.**

This instruction is for using the optional memory for extension functions.

When K12 is set to the device specified by (s), the inverter parameter is read out to the PLC.

1. 16-bit operation (EXTR_OUT, EXTRP_OUT)

The value of the parameter specified by (sd2) is read out from the inverter*1 of the station number specified by (sd1) to the device specified by (sd3).



*1. General purpose inverters FREQROL-A500/E500/S500 (with communications functions) series made by Mitsubishi Electric Corporation

2. Related devices

The same as the short mail transmission described above.

Caution

The FX2N and FX2NC PLCs of V3.00 or later support the EXTR_IN instruction.

7.18 Others

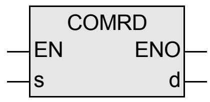
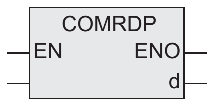
7.18.1 COMRD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction reads the comment data for registered devices written to the PLC by programming software such as GX Works2.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
COMRD	16 bits	Continuous		COMRD(EN,s,d);
COMRDP	16 bits	Pulse		COMRDP(EN,s,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Device for which comment to be read is registered	ANY_SIMPLE
Output variable	ENO	Execution state	Bit
	(d)	Head device storing read comment	String

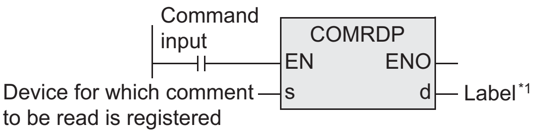
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)	●	●	●			●						●	●	●	●				●					
(d)												●	●	●	●				●					

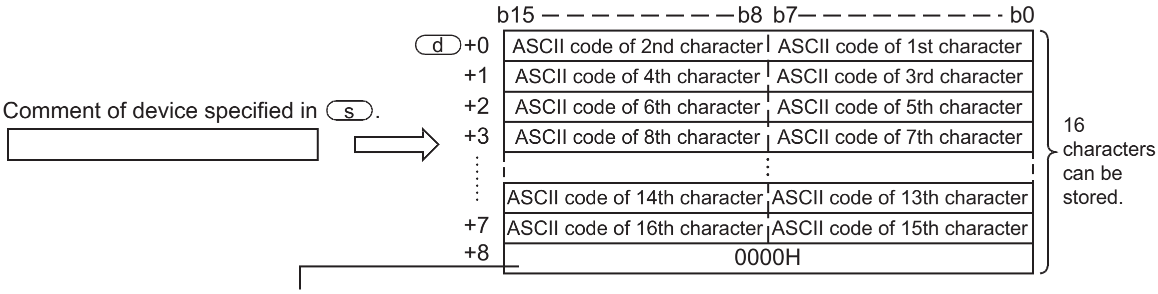
Function and operation explanation

1. 16-bit operation (COMRD/COMRDP)

1) The comment registered for the device specified by (s) is read, and stored in ASCII code in the device specified by (d).

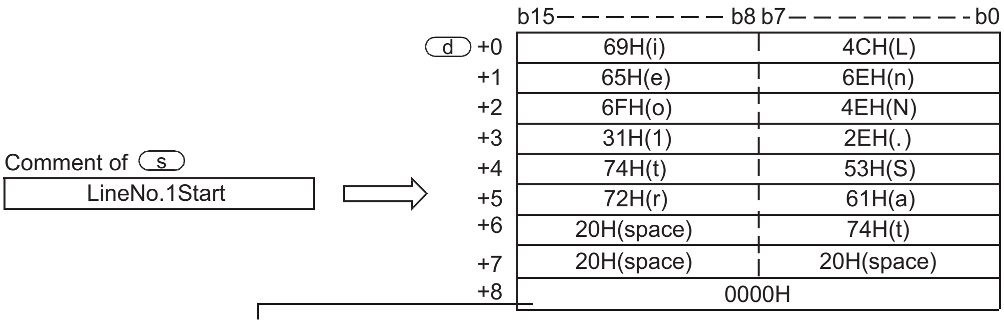


*1. This defines the head of the device that stores the comment read out.



When the comment is made up of an even number of characters:
- When M8091 is OFF, "0000H" is stored to the device following the last character.
- When M8091 is ON, the device following the last character does not change.

For example, when the comment of the device specified by (s) is "Line No.1 Start", it is stored in the device specified by (d) as shown below.



When the comment is made up of an odd number of characters:
- When M8091 is OFF, "00H" is written to the high order byte of the device that stores the last character.
- When M8091 is ON, the high order byte of the device that stores the last character does not change.

2) The last data of the device specified by (d) is as follows depending on the ON/OFF status of M8091.

ON/OFF status	Contents of processing
M8091=OFF	• When the comment is made up of an odd number of characters, "00H" is written to the high order one byte (8 bits) of the device storing the last character of the comment. • When the comment is made up of an even number of characters, "00H" is written to the device that follows the device storing the last character of the comment.
M8091=ON	• When the comment is made up of an odd number of characters, the high order one byte (8 bits) of the device storing the last character of the comment does not change. • When the comment is made up of an even number of characters, the device that follows the device storing the last character of the comment does not change.

Related devices

Device	Name	Description
M8091	Output character number selector signal	Refer to the above explanation.

Cautions

1) When handling character string data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project.
   Use a label to handle character string data.
   Use a global label to specify a label.
2) Specify a device number in the device specified by (s) for which a comment is registered in the PLC.
   If a comment is not registered for the device specified by (d), "20H" (space) is stored in the device specified by (d) for the number of characters in the comment (16 half-width characters).
3) The FX3UC PLC of V2.20 or later supports this instruction.

Error

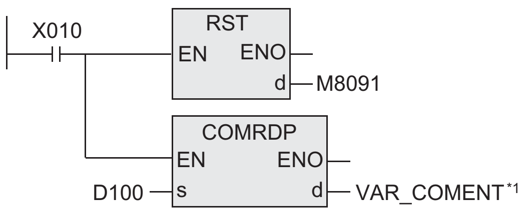
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When a comment is not registered for the device specified by (s) (error code: K6706)
2) When the range of points used from the device specified by (d) for the comment exceeds the corresponding device range (error code: K6706)
   The comment, however, is written up to that point.

Program examples

In the program shown below, the comment "Target Line A" registered to D100 is stored in ASCII code in D0 and later when X010 is set to ON. (When M8091 is OFF)

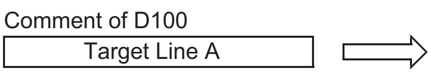
[Structured ladder]



*1. VAR_COMENT is a global label and is defined as D0.

[ST]

```
RST(X010,M8091);
VAR_COMENT:=COMRDP(X010,D100);
```



	b15 — b8	b7 — b0
D0	61H(a)	54H(T)
D1	67H(g)	72H(r)
D2	74H(t)	65H(e)
D3	4CH(L)	20H(space)
D4	6EH(n)	69H(i)
D5	20H(space)	65H(e)
D6	20H(space)	41H(A)
D7	20H(space)	20H(space)
D8	0000H	

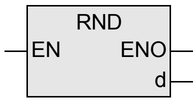
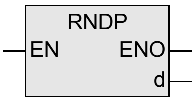
7.18.2 RND

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction generates random numbers.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RND	16 bits	Continuous		RND(EN,d);
RNDP	16 bits	Pulse		RNDP(EN,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
Output variable	ENO	Execution state	Bit
	(d)	Head device storing a random number	ANY16

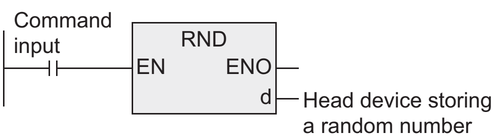
3. Applicable devices

Operand type	Bit Devices							Word Devices													Others					
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)									●	●	●	●	●	●	●	●			●					

Function and operation explanation

1. 16-bit operation (RND/RNDP)

This instruction generates a pseudo-random number within the range from 0 to 32767, and stores it as a random number to the device specified by (d).
In the pseudo-random number sequence, the source value of a random number is calculated at every time, and this instruction calculates a pseudo-random number using the source value.



Pseudo-random number calculation equation:

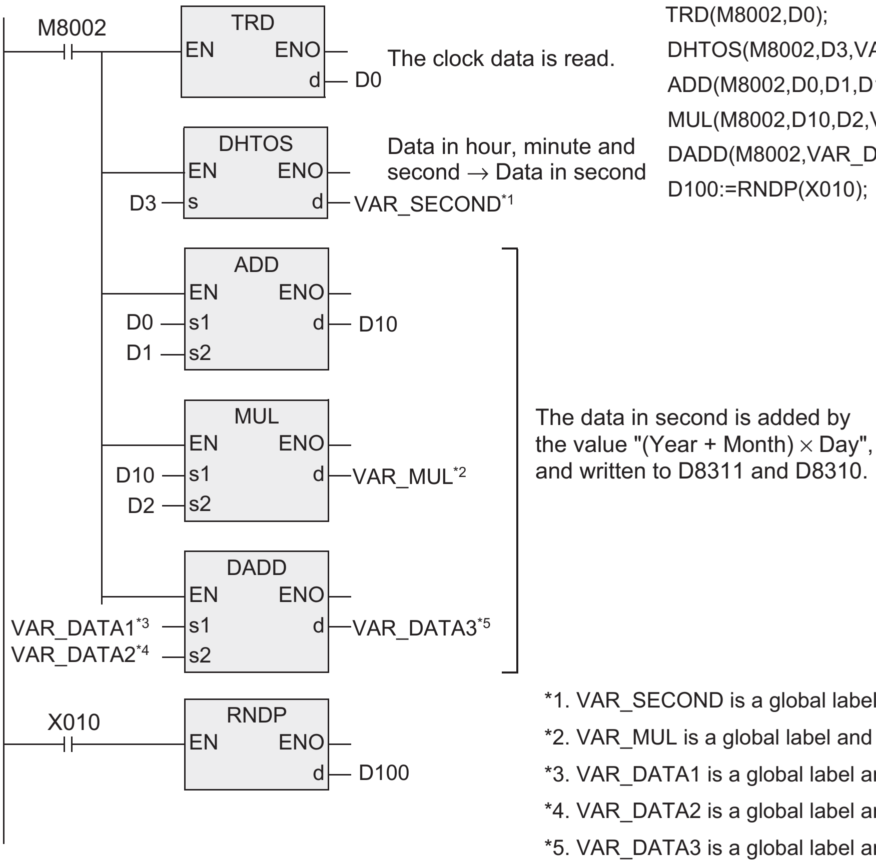
(D8311, D8310) = (D8311, D8310)[*1] × 103515245+12345...1)
d = "[D8311, D8310]>>16)& Logical product > 00007FFFh"

*1. To (D8311, D8310), write a non-negative value (0 to 2,147,483,647) only once when the PLC mode switches from STOP to RUN.
[K1 is written to (D8311, D8310) as the initial value when the power is restored.]

Program examples

In the program example shown below, a random number is stored to D100 every time X010 turns ON.
When the PLC mode switches from STOP to RUN, the time data converted into seconds and added by the value "(Year + Month) × Day" is written to (D8311 and D8310).

[Structured ladder]



[ST]

```
TRD(M8002,D0);
DHTOS(M8002,D3,VAR_SECOND);
ADD(M8002,D0,D1,D10);
MUL(M8002,D10,D2,VAR_MUL);
DADD(M8002,VAR_DATA1,VAR_DATA2,VAR_DATA3);
D100:=RNDP(X010);
```

*1. VAR_SECOND is a global label and is defined as D14.
*2. VAR_MUL is a global label and is defined as D12.
*3. VAR_DATA1 is a global label and is defined as D14.
*4. VAR_DATA2 is a global label and is defined as D12.
*5. VAR_DATA3 is a global label and is defined as D8310.

7.18.3 DUTY

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction generates the timing signal whose one cycle corresponds to the specified number of operation cycles.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DUTY	16 bits	Continuous	DUTY EN ENO n1 d n2	DUTY(EN,n1,n2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	n1	Number of scans (operation cycles) to remain ON	ANY16
	n2	Number of scans (operation cycles) to remain OFF	ANY16
Output variable	ENO	Execution state	Bit
	d	Timing clock output destination	Bit

3. Applicable devices

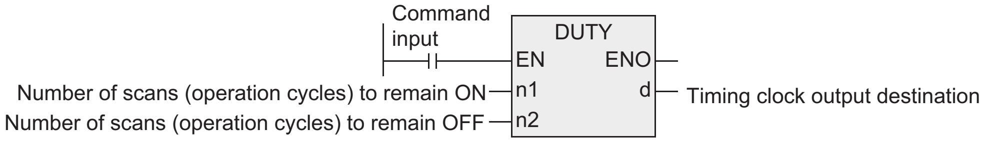
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
n1												●	●	●	●					●	●			
n2												●	●	●	●					●	●			
d			▲																●					

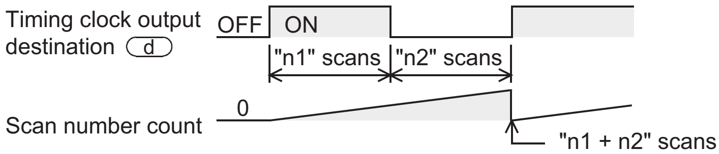
▲: Refer to "Cautions"

Function and operation explanation

1. 16-bit operation (DUTY)

1) The timing clock output of the device specified by (d) is set to ON and OFF with the ON duration for "n1" scans and OFF duration for "n2" scans.





2) Specify either one among M8330 to M8334 as the timing clock output destination device specified by (d).

3) The counted number of scans is stored in either one among D8330 to D8334 in accordance with the timing clock output destination device specified by (d).
The counted number of scans stored in either one among D8330 to D8334 is reset when the counted value reaches "n1 + n2" or when the command input (instruction) is set to ON.

Timing clock output destination	Scan counting device
M8330	D8330
M8331	D8331
M8332	D8332
M8333	D8333
M8334	D8334

4) When the command input is set to ON, the operation is started. The timing clock output destination device specified by (d) is set to ON or OFF by END instruction.
Even if the command input is set to OFF, the operation is not stopped. In the STOP mode, the operation is suspended. When the power of the PLC is turned OFF, the operation is stopped.

5) When "n1" and "n2" are set to "0", the device specified by (d) is set to the following status:

n1, n2 status	ON/OFF status of (d)
n1 = 0,n2 ≥ 0	d = fixed to OFF
n1 > 0,n2 = 0	d = fixed to ON

Related devices

Device	Name	Description
M8330	Timing clock output 1	Timing clock output in DUTY instruction
M8331	Timing clock output 2	
M8332	Timing clock output 3	
M8333	Timing clock output 4	
M8334	Timing clock output 5	
D8330	Counted number of scans for timing clock output 1	Counted number of scans for timing clock output 1 in DUTY instruction
D8331	Counted number of scans for timing clock output 2	Counted number of scans for timing clock output 2 in DUTY instruction
D8332	Counted number of scans for timing clock output 3	Counted number of scans for timing clock output 3 in DUTY instruction
D8333	Counted number of scans for timing clock output 4	Counted number of scans for timing clock output 4 in DUTY instruction
D8334	Counted number of scans for timing clock output 5	Counted number of scans for timing clock output 5 in DUTY instruction

Cautions

1) DUTY instruction can be used up to 5 times (points).
It is not permitted, however, to use the same timing clock output destination device (device specified by d) for two or more DUTY instructions.
2) The FX3UC PLC of V2.20 or later supports this instruction.
3) Some restrictions to applicable devices
▲: Specify M8330 to M8334.

Error

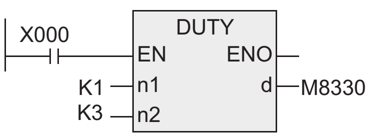
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "n1" and/or "n2" is less than "0" (error code: K6706)
2) When any device other than M8330 to M8334 is set to the device specified by d. (error code: K6705)

Program examples

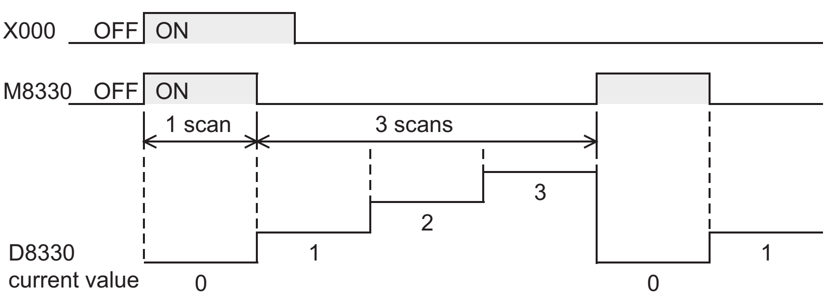
In the program shown below, when X000 is set to ON, M8330 is set to ON for 1 scan and OFF for 3 scans.

[Structured ladder]



[ST]

M8330:=DUTY(X000,K1,K3);



7.18.4 CRC

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This CRC instruction calculates the CRC (cyclic redundancy check) value which is an error check method used in communication.
In addition to CRC value, there are other error check methods such as parity check and sum check. For obtaining the horizontal parity value and sum check value, CCD instruction is available. CRC instruction uses "$X^{16} + X^{15} + X^2 + 1$" as a polynomial for generating the CRC value (CRC-16).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
CRC	16 bits	Continuous	CRC (EN → ENO, s → d, n)	CRC(EN,s,n,d);
CRCP	16 bits	Pulse	CRCP (EN → ENO, s → d, n)	CRCP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Head device storing data for which the CRC value is generated	ANY16
	(n)	Number of 8-bit (byte) data for which the CRC value is generated or the device storing the number of data	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Device storing the generated CRC value	ANY16

3. Applicable devices

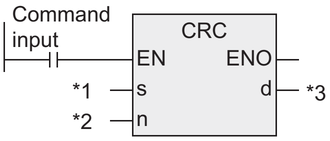
Operand type	Bit Devices: System user							Word Devices: Digit specification				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								▲	▲	▲	▲	●	●	●	●	●			●					
(d)									▲	▲	▲	●	●	●	●	●			●					
(n)														●	●					●	●			

▲: Refer to "Cautions"

Function and operation explanation

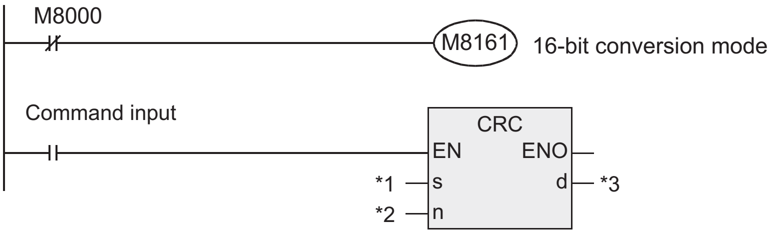
1. 16-bit operation

CRC value is generated for "n" 8-bit data (unit: byte) starting from a device specified in (s), and stored to the device specified by (d).
The 8-bit conversion mode and 16-bit conversion mode are available in this instruction, and the mode can be switched by turning ON or OFF M8161. For the operation in each mode, refer to the later descriptions.
$X^{16} + X^{15} + X^2 + 1$" is used as a polynomial for generating the CRC value (CRC-16).



*1. Head device storing data for which the CRC value is generated
*2. Number of 8-bit (byte) data for which the CRC value is generated or the device storing the number of data
*3. Device storing the generated CRC value

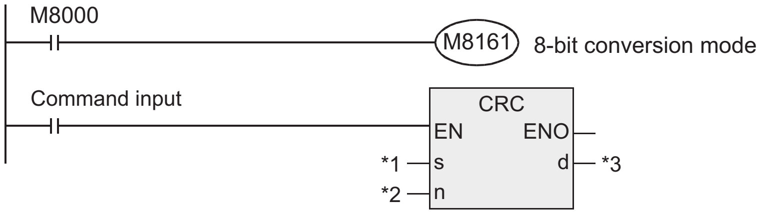
1) 16-bit conversion mode [M8161 = OFF]
   In this mode, the operation is executed for high-order 8 bits (byte) and low-order 8 bits (byte) of a device specified in (s).
   The operation result is stored to one 16-bit device specified in (d).



*1. Head device storing data for which the CRC value is generated
*2. Number of 8-bit (byte) data for which the CRC value is generated or the device storing the number of data
*3. Device storing the generated CRC value

			Example: (s) = D100, (d) = D0, n = 6		
			Device	Contents of target data: 8 bits	Contents of target data: 16 bits
Device storing data for which the CRC value is generated	(s)	Low-order byte	Low-order bits of D100	01H	0301H
		High-order byte	High-order bits of D100	03H	
	(s)+1	Low-order byte	Low-order bits of D101	03H	0203H
		High-order byte	High-order bits of D101	02H	
	(s)+2	Low-order byte	Low-order bits of D102	00H	1400H
		High-order byte	High-order bits of D102	14H	
	⋮	⋮	-		
	(s)+n/2-1	Low-order byte	-		
		High-order byte			
Device storing the generated CRC value	(d)	Low-order byte	Low-order bits of D0	E4H	41E4H
		High-order byte	High-order bits of D0	41H	

2) 8-bit conversion mode [M8161 = ON]
In this mode, the operation is executed only for low-order 8 bits (low-order byte) of device specified by (s).
With regard to the operation result, low-order 8 bits (byte) are stored to a device specified by (d), and high-order 8 bits (byte) are stored to a device specified by (d) +1.



*1. Head device storing data for which the CRC value is generated

*2. Number of 8-bit (byte) data for which the CRC value is generated or the device storing the number of data

*3. Device storing the generated CRC value

			Example: (s) = D100, (d) = D0, n = 6	
			Device	Contents of target data
Device storing data for which the CRC value is generated	(s)	Low-order byte	Low-order bits of D100	01H
	(s)+1	Low-order byte	Low-order bits of D101	03H
	(s)+2	Low-order byte	Low-order bits of D102	03H
	(s)+3	Low-order byte	Low-order bits of D103	02H
	(s)+4	Low-order byte	Low-order bits of D104	00H
	(s)+5	Low-order byte	Low-order bits of D105	14H
	⋮		-	
	(s)+n-1	Low-order byte	-	
Device storing the generated CRC value	(d)	Low-order byte	Low-order bits of D0	E4H
	(d)+1	Low-order byte	Low-order bits of D1	41H

2. Related devices

Related devices	Description	
M8161*1	ON	CRC instruction operates in the 8-bit mode.
	OFF	CRC instruction operates in the 16-bit mode.

*1. Cleared when the PLC mode is changed from RUN to STOP.

Cautions

1) In this instruction, "$X^{16} + X^{15} + X^2 + 1$" is used as a polynomial for generating the CRC value (CRC-16). There are many other standard polynomials for generating the CRC value. Note that the CRC value completely differs if an adopted polynomial is different.

Reference: Major polynomials for generating the CRC value

Name	Generating polynomial
CRC-12	$X^{12} + X^{11} + X^3 + X^2 + X + 1$
CRC-16	$X^{16} + X^{15} + X^2 + 1$
CRC-32	$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
CRC-CCITT	$X^{16} + X^{12} + X^5 + 1$

2) Some restrictions to applicable devices
▲: Be sure to specify four digits for the bit devices (K4□○○○).

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

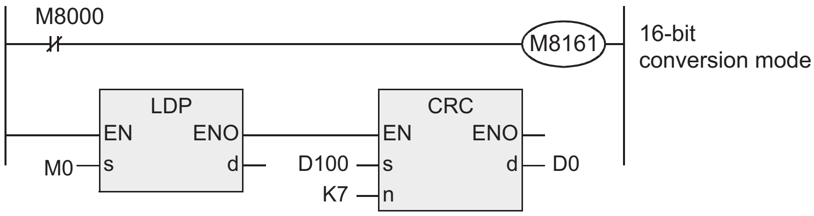
1) When any digits other than 4 digits are specified as the devices specified as s or d in digit specification of bit device (error code: K6706)
2) When n is outside the allowable range (1 to 256) (error code: K6706)
3) When a device specified by s+n-1 or d+1 is outside the allowable range (error code: K6706)

Program examples

In the program example shown below, the CRC value of the ASCII code "0123456" stored in D100 to D106 is generated and stored to D0 when M0 turns ON.

1. In the case of 16-bit mode

[Structured ladder]



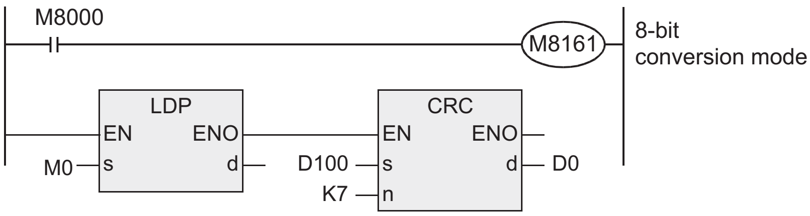
[ST]

```
M8161:=NOT M8000;
LDP(TRUE,M0);
D0:=CRC(TRUE,D100,K7);
```

	Contents of data			
			Target data	
Device storing data for which the CRC value is generated	D100	3130H	Low-order byte	30H
			High-order byte	31H
	D101	3332H	Low-order byte	32H
			High-order byte	33H
	D102	3534H	Low-order byte	34H
			High-order byte	35H
	D103	3736H	Low-order byte	36H
			-	-
Device storing the generated CRC value	D0	2ACFH	Low-order byte	CFH
			High-order byte	2AH

2. In the case of 8-bit mode

[Structured ladder]



[ST]

```
M8161:=M8000;
LDP(TRUE,M0);
D0:=CRC(TRUE,D100,K7);
```

	Contents of target data		
Device storing data for which the CRC value is generated	D100	Low-order byte	30H
	D101	Low-order byte	31H
	D102	Low-order byte	32H
	D103	Low-order byte	33H
	D104	Low-order byte	34H
	D105	Low-order byte	35H
	D106	Low-order byte	36H
Device storing the generated CRC value	D0	Low-order byte	CFH
	D1	Low-order byte	2AH

7.18.5 DHCMOV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction transfers the current value of a specified high speed counter or ring counter.
The function of this instruction varies depending on the PLC version.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DHCMOV	32 bits	Continuous	DHCMOV —EN ENO— —s d— —n	DHCMOV(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Device of high speed counter or ring counter handled as transfer source	ANY32
	n	Specification to clear the current value of high speed counter or ring counter (transfer source) after transfer [clear (k1), no processing (K0)]	ANY32
Output variable	ENO	Execution state	Bit
	d	Device handled as transfer destination	ANY32

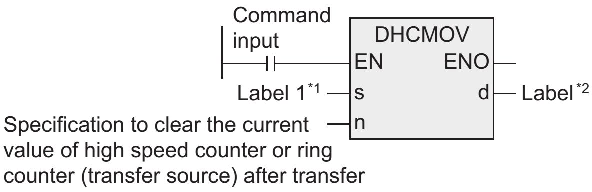
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s													▲	▲										
d														●	●									
n																				●	●			

▲: Refer to "Cautions"

Function and operation explanation

1. 32-bit operation (DHCMOV)



*1. This defines the device of the high speed counter or ring counter of the transfer destination.

*2. This defines the transfer destination device.

1) The current value of a high speed counter or ring counter specified by (s) is transferred to the device specified by (d).

Device (s)		[(d)+1, (d)] after instruction is executed
High speed counter	C235 to C255	Current value of high speed counter (s) → [(d)+1, (d)]
Ring counter*1	D8099	D8099→(d) "0" is stored in (d) +1.
	D8398	Current value of [D8399, D8398] → [(d)+1, (d)]

2) After transfer, the current value of the high speed counter or ring counter is processed as shown in the table below depending on the set value of "n".

"n" set value	Operation
K0(H0)	Does not clear the current value (no processing).
K1(H1)	Clears the current value to "0".

*1. Ring counters (D8099 and D8398) cannot be specified in FX3UC PLCs earlier than Ver. 2.20 (not inclusive).

2. High speed counter current value update timing and the effect of DHCMOV instruction

1) High speed counter current value update timing
When a pulse is input to an input terminal for a high speed counter (C235 to C255), the high speed counter executes up-counting or down-counting.
If the current value of a high speed counter is handled in an instruction such as the normal MOV instruction, the current value is updated at the timing shown in the table below. As a result, it is affected by the program scan time.

	Current value update timing
Hardware counter	When OUT instruction for the counter is executed
Software counter	Every time a pulse is input

By using DHCMOV instruction, the current value can be updated and transferred when it is executed.

2) Effect of DHCMOV instruction
   a) By using both input interrupt and DHCMOV instruction, the current value of a high speed counter can be received at the rising edge or falling edge of an external input (at reception of input interrupt).
   → **Refer to the Program example 2.**
   b) When DHCMOV instruction is used just before a comparison instruction (CMP, ZCP or comparison contact instruction), the latest value of a high speed counter is used in comparison. Unlike the comparison instruction for high speed counters (DHSCS, DHSCR or DHSZ), the following points must be kept in mind when using the DHCMOV instruction.
   - When the current value of a hardware counter is compared using CMP, ZCP or comparison contact instruction (not using a designated high speed counter comparison instruction), a hardware counter does not change into a software counter.
   → **FX Structured Programming Manual (Device & Common)**
   - When the number of high speed software counter comparison instructions is reduced, the total frequency limitation is decreased.
   → **FX Structured Programming Manual (Device & Common)**
   - When it is necessary to execute comparison and change an output contact (Y) as soon as the current value of a high speed counter changes, use a designated high seed counter comparison instruction (DHSCS, DHSCR or DHSZ)
   - DHCMOV instruction can be used as many times as necessary.

Cautions

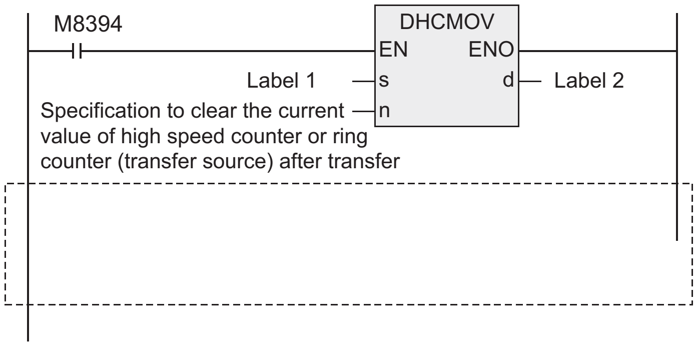
When programming DHCMOV instruction in an input interrupt program, the following points should be observed.
For assignment of pointers for input interrupt and inputs, refer to the table shown in 5) below.

1) Program EI and FEND instructions in the main program.
   They are necessary to execute an input interrupt program.

→ **For EI and FEND, refer to Sections 7.1.6 and 7.1.7.**

2) When programming DHCMOV instruction in the first line in an input interrupt program, be sure to use the pattern program shown below. Be sure to use the command contact M8394.

[Interrupt program]
(Event : I□□□)

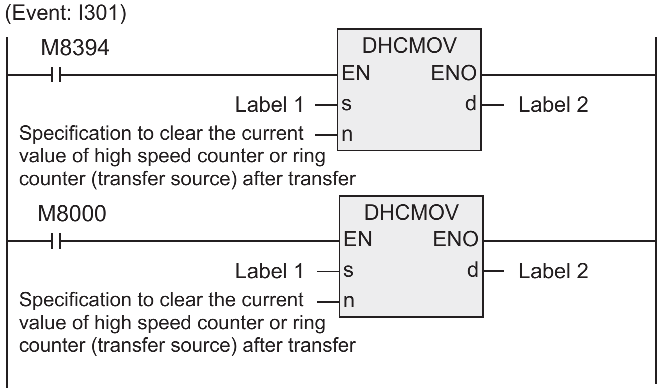


Pattern program for using the DHCMOV instruction in an interrupt program.

Program to be processed as input interrupt

3) If two or more DHCMOV instructions are used in one input interrupt program, only the first instruction (just after the interrupt pointer) is executed when the interrupt is generated.
   The rest of the interrupt, including additional instructions, is executed according to normal interrupt processing.
   Do not use M8394 as the command contact for the DHCMOV instructions following the first.

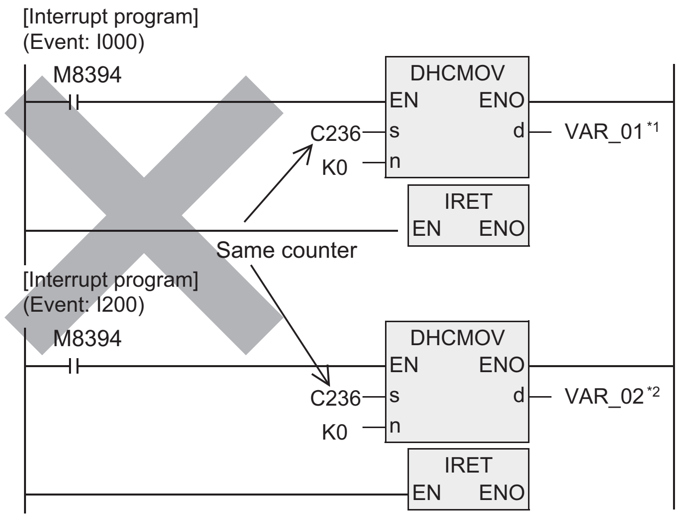
[Interrupt program]
(Event: I301)



When the input X003 turns from OFF to ON (that is, when input interrupt is accepted)
(s) → [(d)+1,(d)]

When this instruction is executed in interrupt program
(s)'→ [(d)'+1,(d)']

4) It is not permitted to use DHCMOV instruction for the same counter in two or more input interrupt programs.



*1. VAR_01 is a global label and is defined as D0.

*2. VAR_01 is a global label and is defined as D2.

5) While input interrupts are disabled by the interrupt disable flags (shown in the table below), DHCMOV instructions are not executed when they are placed inside a corresponding interrupt.

Interrupt disable flag	Corresponding interrupt pointer	Input number corresponding to interrupt pointer
M8050*1	I000,I001	X000
M8051*1	I100,I101	X001
M8052*1	I200,I201	X002
M8053*1	I300,I301	X003
M8054*1	I400,I401	X004
M8055*1	I500,I501	X005

*1. When the PLC mode is changed from RUN to STOP, if an input interrupt is generated while input interrupts are disabled by something other than the interrupt disable flags M8050 to M8055 (after execution of DI instruction and before execution of EI instruction), DHCMOV instruction is immediately executed, but execution of the interrupt program is held. The interrupt program will be executed after EI instruction is executed and interrupt are enabled.

6) Some restrictions to the applicable devices

▲: Only the high speed counters (C235 to C255) and ring counters (D8099, D8398)*1 can be specified.

*1. The FX3UC PLCs of before Ver. 2.20 (not inclusive) cannot specify the ring counter (D8099, D8398).

Function change depending on the version

The function of this instruction changes depending on the version as shown in the table below.

Applicable version		Item	Outline of function
FX3U	FX3UC		
Ver. 2.20 or later	Ver. 2.20 or later	Target device	Ring counter (D8099 and D8398) can be specified in the device specified by (s).

Error

An operation error occurs in the following case. The error flag M8067 turns ON, and the error code is stored in D8067.

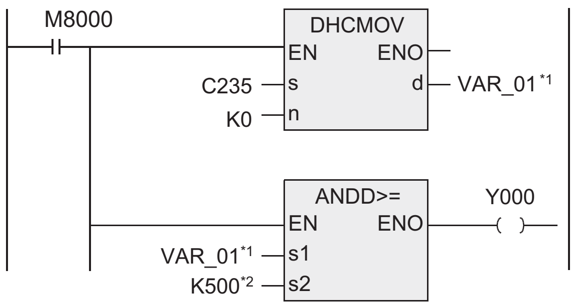
1) When a device specified in (s) or (d) is outside the allowable range (error code: K6705)

Program examples

1. Program examples 1

In the program example below, the current value of the high speed counter C235 is compared in each operation cycle, and then the output Y000 is set to ON if the current value is "K500" or more (when the current value of C235 is not cleared.)

[Structured ladder]



[ST]

```
DHCMOV(M8000,C235,K0,VAR_01);

Y000:=ANDD>=(M8000,VAR_11,K500);
```

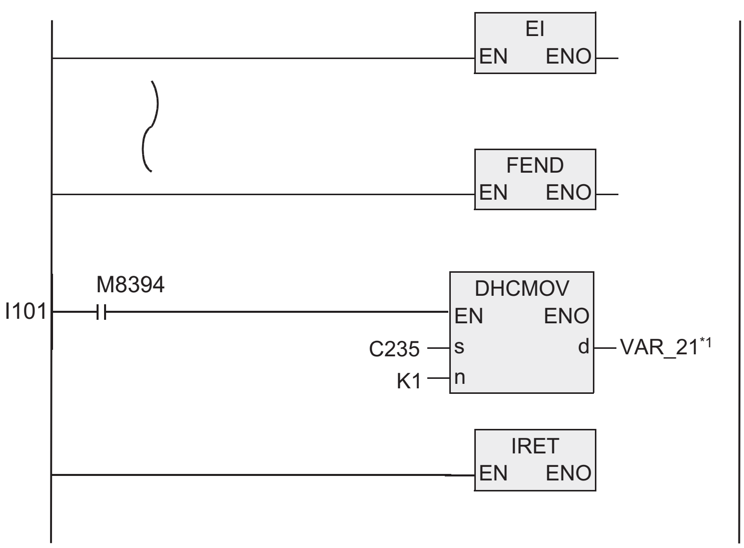
*1. VAR_01 is a global label and is defined as D0.

*2. K0: The current value of the high speed counter is not cleared when DHCMOV instruction is executed.
K1: The current value of the high speed counter is cleared when DHCMOV instruction is executed.

2. Program examples 2

In the program example below, the current value of C235 is transferred to D201 and D200, and the current value of C235 is cleared when X001 turns from OFF to ON.

[Structured ladder]



[ST]

```
EI(TRUE);
  }
FEND(TRUE);
VAR_21:=DHCMOV(M8394,C235,K1);
IRET(TRUE);
```

*1. VAR_21 is a global label and is defined as D200.

*2. K0: The current value of the high speed counter is not cleared when DHCMOV instruction is executed.
K1: The current value of the high speed counter is cleared when DHCMOV instruction is executed.

7.19 Block Data Operation

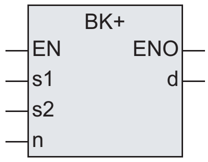
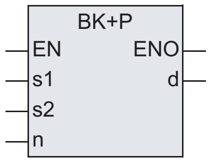
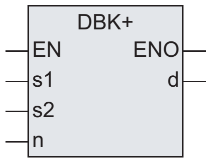
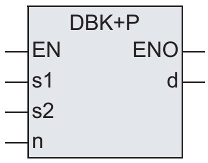
7.19.1 BK+

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction adds binary block data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BK+	16 bits	Continuous		BK+(EN,s1,s2,n,d);
BK+P	16 bits	Pulse		BK+P(EN,s1,s2,n,d);
DBK+	32 bits	Continuous		DBK+(EN,s1,s2,n,d);
DBKP+P	32 bits	Pulse		DBK+P(EN,s1,s2,n,d);

2. Set data

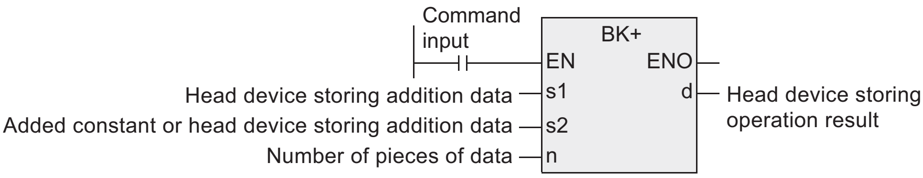
Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Head device storing addition data	ANY16	ANY32
	s2	Added constant or head device storing addition data	ANY16	ANY32
	n	Number of pieces of data	Bit	
Output variable	ENO	Execution state	ANY16	ANY32
	d	Head device storing operation result	ANY16	ANY32

3. Applicable devices

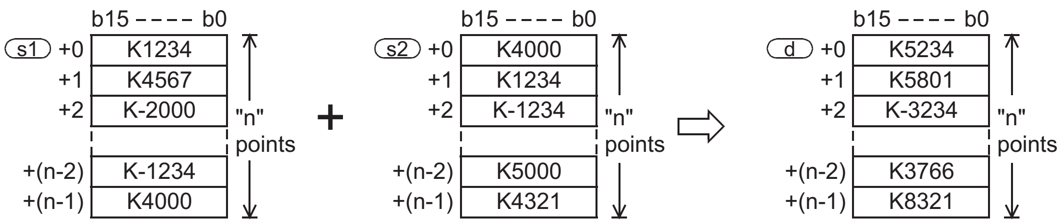
Operand type	Bit Devices							Word Devices												Others					
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)												●	●	●	●				●					
(s2)												●	●	●	●				●	●	●			
(d)												●	●	●	●				●					
(n)														●	●					●	●			

Function and operation explanation

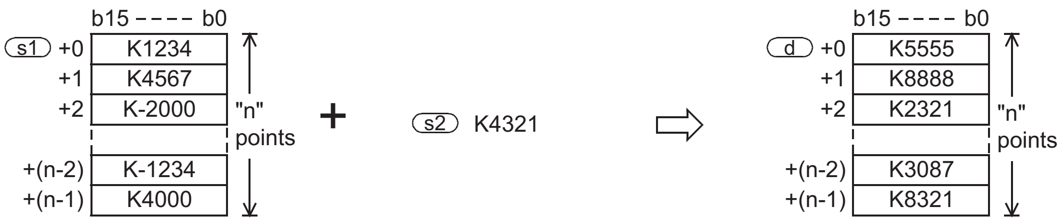
1. 16-bit operation (BK+/BK+P)



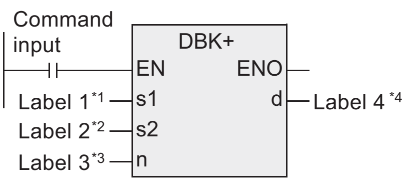
1) "n" 16-bit binary data starting from the device specified by (s2) are added to "n" 16-bit binary data starting from the device specified by (s1), and the operation result is stored in "n" points starting from the device specified by (d).



2) A (16-bit) constant from -32768 to 32767 can be directly specified in the device specified by (s2).

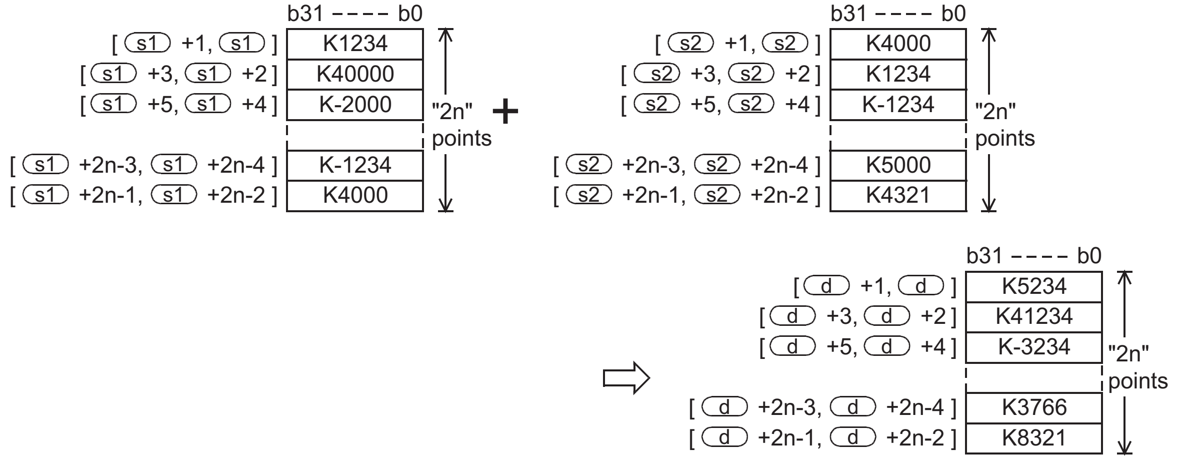


2. 32-bit operation (DBK+/DBK+P)

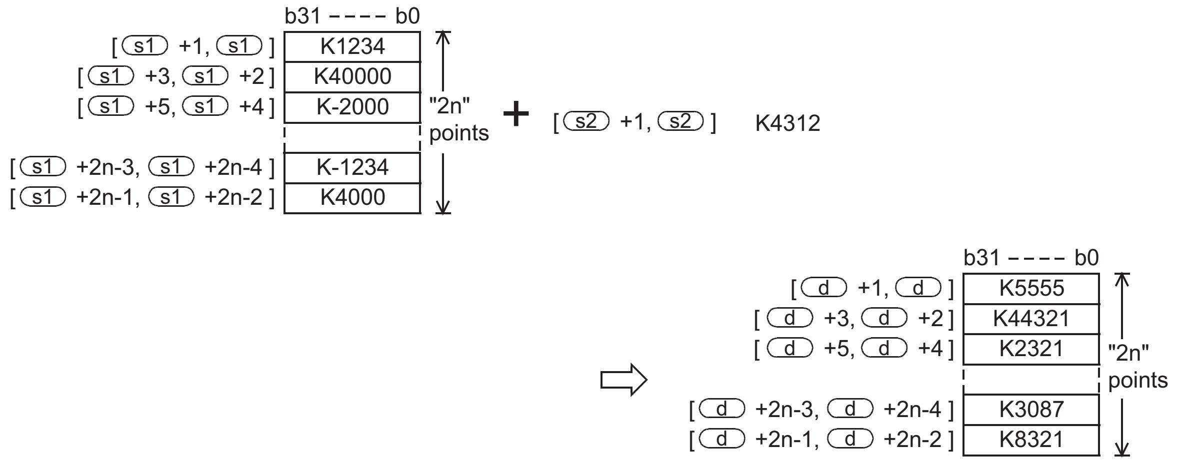


*1. This defines the head of the device that stores the addition data.
*2. This defines the added constant or the head of the device that stores the addition data.
*3. This defines the number of pieces of data.
*4. This defines the head of the device that stores the operation result.

1) "2n" 32-bit binary data starting from the device specified by (s2) are added to "2n" 32-bit binary data starting from the device specified by (s1), and the operation result is stored in "2n" points starting from the device specified by (d).



2) A (32-bit) constant from -2,147,483,648 to 2,147,483,647 can be directly specified in the device specified by (s2).



Related instruction

Instruction	Description
BK-	Subtracts binary block data.

Cautions

1) When underflow or overflow occurs in the operation result, the following processing is executed. At this time, the carry flag does not turn ON.

   a) In the case of 16-bit operation

K32767(H7FFF)	+	K2(H0002)	→	K-32767(H8001)
K-32768(H8000)	+	K-2(HFFFE)	→	K32766(H7FFE)

   b) In the case of 32-bit operation

K2,147,483,647(H7FFFFFFF)	+	K2(H00000002)	→	K-2,147,483,647(H80000001)
K-2,147,483,648(H80000000)	+	K-2(HFFFFFFFE)	→	K2,147,483,646(H7FFFFFFE)

2) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project.
   Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.

3) The FX3UC PLC of V. 2.20 or later supports this instruction.

Error

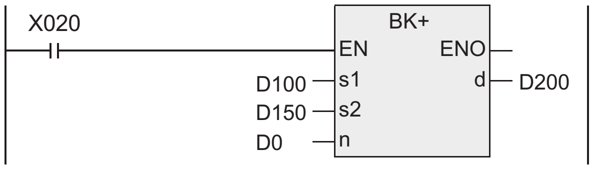
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "n" ("2n" in 32-bit operation) devices starting from the devices specified by (s1), (s2) and/or (d) exceed the corresponding device range (error code: K6706).
2) When "n" ("2n" in 32-bit operation) devices starting from the device specified by (s1) overlap "n" ("2n" in 32-bit operation) devices starting from the device specified by (d) (error code: K6706.)
3) When "n" ("2n" in 32-bit operation) devices starting from the device specified by (s2) overlap "n" ("2n" in 32-bit operation) devices starting from the device specified by (d) (error code: K6706.)

Program examples

In the program shown below, the specified number of pieces of data stored in D150 to D0 are added to the specified number of pieces of data stored in D100 to D0 when X020 is set to ON, and the operation result is stored in D200 and later

[Structured ladder]



[ST]

BK+(X020,D100,D150,D0,D200);

	b15 ---- b0			b15 ---- b0			b15 ---- b0
D100	6789	+	D150	1234	⇨	D200	8023
D101	7821		D151	2032		D201	9853
D102	5432		D152	-3252		D202	2180
D103	3520		D153	-1000		D203	2520

D0 = 4

7.19.2 BK-

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction subtracts binary block data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BK-	16 bits	Continuous	BK- EN ENO s1 d s2 n	BK-(EN,s1,s2,n,d);
BK-P	16 bits	Pulse	BK-P EN ENO s1 d s2 n	BK-P(EN,s1,s2,n,d);
DBK-	32 bits	Continuous	DBK- EN ENO s1 d s2 n	DBK-(EN,s1,s2,n,d);
DBK-P	32 bits	Pulse	DBK-P EN ENO s1 d s2 n	DBK-P(EN,s1,s2,n,d);

2. Set data

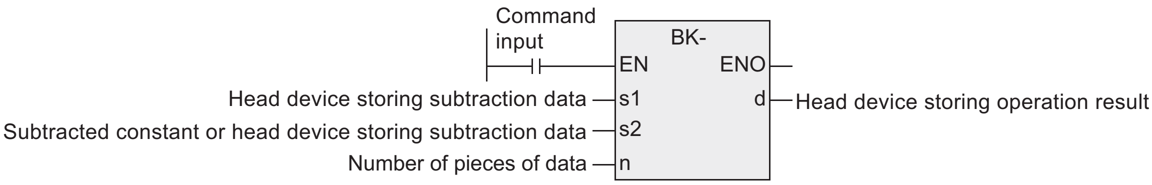
Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Head device storing subtraction data	ANY16	ANY32
	s2	Subtracted constant or head device storing subtraction data	ANY16	ANY32
	n	Number of pieces of data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head device storing operation result	ANY16	ANY32

3. Applicable devices

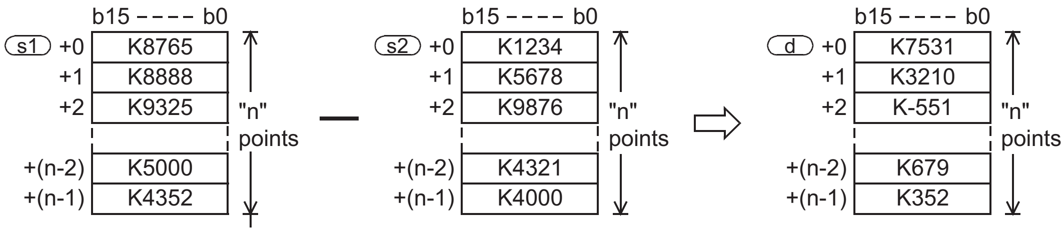
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	X	Y	Modifier	K	H	E	"□"	P
s1												●	●	●	●				●					
s2												●	●	●	●				●	●	●			
d												●	●	●	●				●					
n														●	●					●	●			

Function and operation explanation

1. 16-bit operation (BK-/BK-P)

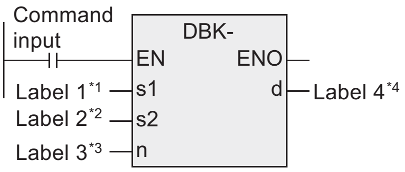


1) "n" 16-bit binary data starting from the device specified by (s2) are subtracted from "n" 16-bit binary data starting from the device specified by (s1), and the operation result is stored in "n" points starting from the device specified by (d).



2) A (16-bit) constant from -32768 to 32767 can be directly specified in (s2).

2. 32-bit operation (DBK-/DBK-P)



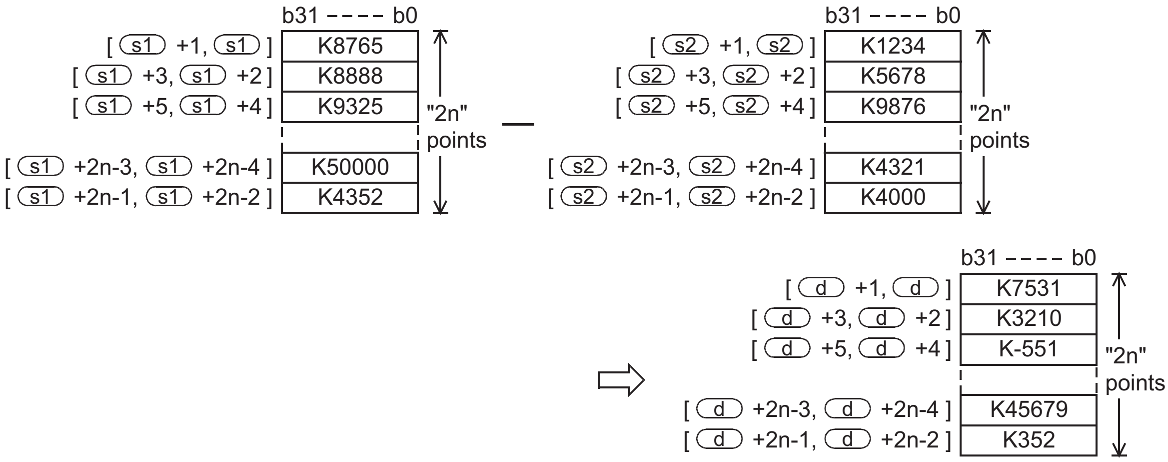
*1. This defines the head of the device that stores the subtraction data.

*2. This defines the subtracted constant or the head of the device that stores the subtraction data.

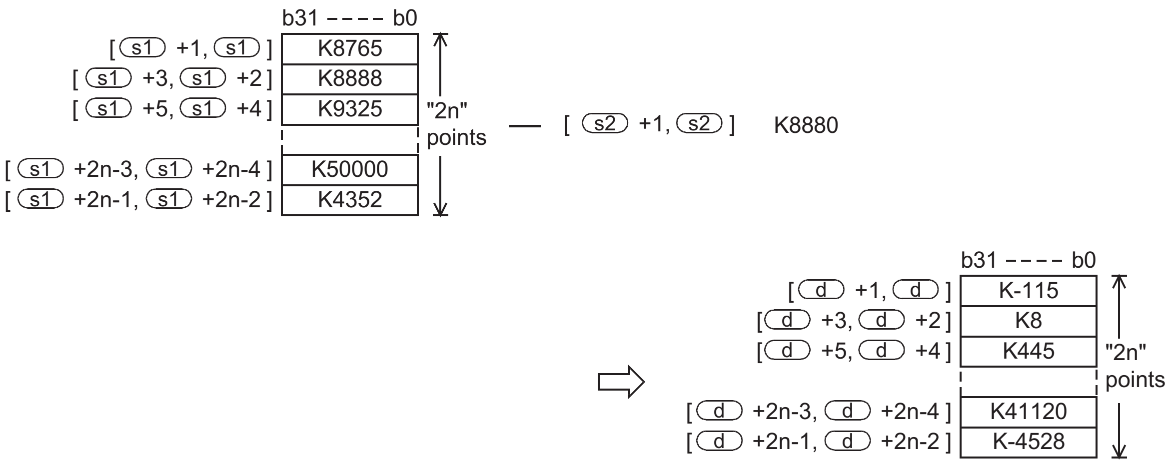
*3. This defines the number of pieces of data.

*4. This defines the head of the device that stores the operation result.

1) "2n" 32-bit binary data starting from the device specified by (s2) are subtracted from "2n" 32-bit binary data starting from the device specified by (s1), and the operation result is stored in "2n" points starting from the device specified by (d).



2) A (32-bit) constant from -2,147,483,648 to 2,147,483,647 can be directly specified in the device specified by (s2).



Related instruction

Instruction	Description
BK+	Adds binary block data.

Cautions

1) When underflow or overflow occurs in the operation result, the following processing is executed. At this time, the carry flag does not turn ON.
   a) In the case of 16-bit operation

K-32768(H8000)	-	K2(H0002)	→	K32766(H7FFE)
K32767(H7FFF)	-	K-2(HFFFE)	→	K-32767(H8001)

   b) In the case of 32-bit operation

K-2,147,483,648(H80000000)	-	K2(H00000002)	→	K2,147,483,646(H7FFFFFFE)
K2,147,483,647(H7FFFFFFF)	-	K-2(HFFFFFFFE)	→	K-2,147,483,647(H80000001)

2) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
3) The FX3UC PLC of V. 2.20 or later supports this instruction.

Error

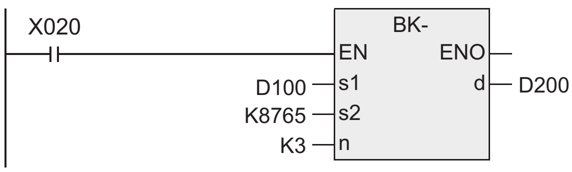
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "n" ("2n" in 32-bit operation) devices starting from the devices specified by (s1), (s2) and/or (d) exceed the corresponding device range (error code: K6706).
2) When "n" ("2n" in 32-bit operation) devices starting from the device specified by (s1) overlap "n" ("2n" in 32-bit operation) devices starting from the device specified by (d) (error code: K6706.)
3) When "n" ("2n" in 32-bit operation) devices starting from the device specified by (s2) overlap "n" ("2n" in 32-bit operation) devices starting from the device specified by (d) (error code: K6706.)

Program examples

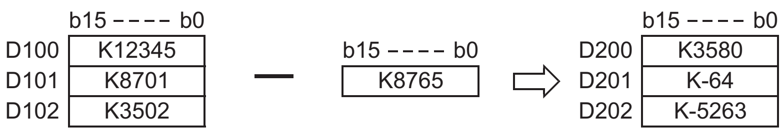
In the program shown below, the constant "8765" is subtracted from the data stored in D100 to D102 when X010 is set to ON, and the operation result is stored in D200 and later.

[Structured ladder]



[ST]

```
BK-(X020,D100,K8765,K3,D200);
```



7.19.3 BKCMP=, BKCMP>, BKCMP<, BKCMP<>, BKCMP<=, BKCMP>=

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

These instructions compare block data in the comparison condition set in each instruction.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
BKCMP=	16 bits	Continuous	BKCMP= EN ENO s1 d s2 n	BKCMP=(EN,s1,s2,n,d);
BKCMP>	16 bits	Continuous	BKCMP> EN ENO s1 d s2 n	BKCMP>(EN,s1,s2,n,d);
BKCMP<	16 bits	Continuous	BKCMP< EN ENO s1 d s2 n	BKCMP<(EN,s1,s2,n,d);
BKCMP<>	16 bits	Continuous	BKCMP<> EN ENO s1 d s2 n	BKCMP<>(EN,s1,s2,n,d);
BKCMP<=	16 bits	Continuous	BKCMP<= EN ENO s1 d s2 n	BKCMP<=(EN,s1,s2,n,d);
BKCMP>=	16 bits	Continuous	BKCMP>= EN ENO s1 d s2 n	BKCMP>=(EN,s1,s2,n,d);
BKCMP=P	16 bits	Pulse	BKCMP=P EN ENO s1 d s2 n	BKCMP=P(EN,s1,s2,n,d);
BKCMP>P	16 bits	Pulse	BKCMP>P EN ENO s1 d s2 n	BKCMP>P(EN,s1,s2,n,d);

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BKCMP<P	16 bits	Pulse	BKCMP<P EN ENO s1 d s2 n	BKCMP<P(EN,s1,s2,n,d);
BKCMP<>P	16 bits	Pulse	BKCMP<>P EN ENO s1 d s2 n	BKCMP<>P(EN,s1,s2,n,d);
BKCMP<=P	16 bits	Pulse	BKCMP<=P EN ENO s1 d s2 n	BKCMP<=P(EN,s1,s2,n,d);
BKCMP>=P	16 bits	Pulse	BKCMP>=P EN ENO s1 d s2 n	BKCMP>=P(EN,s1,s2,n,d);
DBKCMP=	32 bits	Continuous	DBKCMP= EN ENO s1 d s2 n	DBKCMP=(EN,s1,s2,n,d);
DBKCMP>	32 bits	Continuous	DBKCMP> EN ENO s1 d s2 n	DBKCMP>(EN,s1,s2,n,d);
DBKCMP<	32 bits	Continuous	DBKCMP< EN ENO s1 d s2 n	DBKCMP<(EN,s1,s2,n,d);
DBKCMP<>	32 bits	Continuous	DBKCMP<> EN ENO s1 d s2 n	DBKCMP<>(EN,s1,s2,n,d);
DBKCMP<=	32 bits	Continuous	DBKCMP<= EN ENO s1 d s2 n	DBKCMP<=(EN,s1,s2,n,d);
DBKCMP>=	32 bits	Continuous	DBKCMP>= EN ENO s1 d s2 n	DBKCMP>=(EN,s1,s2,n,d);

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DBKCMP=P	32 bits	Pulse	DBKCMP=P EN ENO s1 d s2 n	DBKCMP=P(EN,s1,s2,n,d);
DBKCMP>P	32 bits	Pulse	DBKCMP>P EN ENO s1 d s2 n	DBKCMP>P(EN,s1,s2,n,d);
DBKCMP<P	32 bits	Pulse	DBKCMP<P EN ENO s1 d s2 n	DBKCMP<P(EN,s1,s2,n,d);
DBKCMP<>P	32 bits	Pulse	DBKCMP<>P EN ENO s1 d s2 n	DBKCMP<>P(EN,s1,s2,n,d);
DBKCMP<=P	32 bits	Pulse	DBKCMP<=P EN ENO s1 d s2 n	DBKCMP<=P(EN,s1,s2,n,d);
DBKCMP>=P	32 bits	Pulse	DBKCMP>=P EN ENO s1 d s2 n	DBKCMP>=P(EN,s1,s2,n,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Comparison value or device storing comparison value	ANY16	ANY32
	s2	Head device storing comparison source data	ANY16	ANY32
	n	Number of pieces of comparison data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head device storing comparison result	Bit	

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	X	Y	Modifier	K	H	E	"□"	P
s1												●	●	●	●				●	●	●			
s2												●	●	●	●				●					
d		●	●			●	▲												●					
n														●	●					●	●			

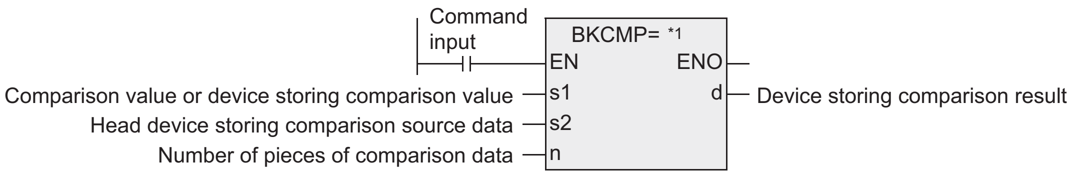
▲: Refer to "Cautions"

Function and operation explanation

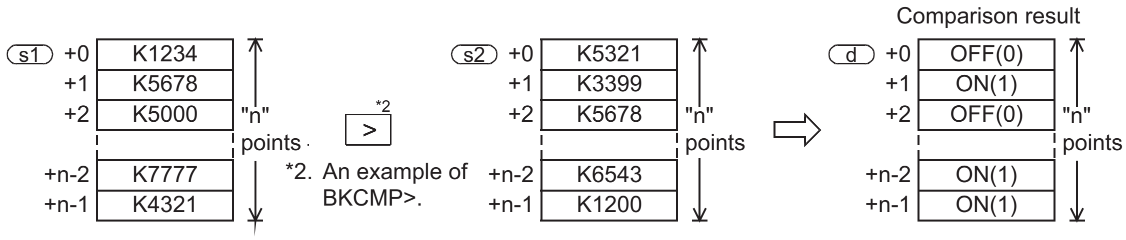
1. 16-bit operation

(BKCMP=, BKCMP>, BKCMP<, BKCMP<>, BKCMP<=, BKCMP>=, BKCMP=P, BKCMP>P, BKCMP<P, BKCMP<>P, BKCMP<=P, BKCMP>=P)

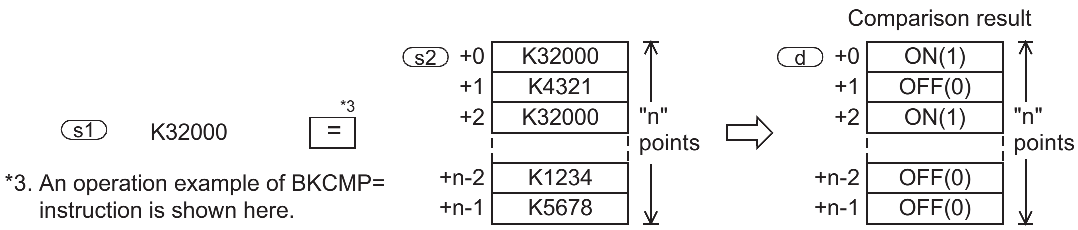
1) "n" 16-bit binary data starting from the device specified by (s1) are compared with "n" 16-bit binary data starting from the device specified by (s2), and the comparison result is stored in "n" points starting from the device specified by (d).



*1. BKCMP=, BKCMP>, BKCMP<, BKCMP<>, BKCMP<=, BKCMP>=, BKCMP=P, BKCMP>P, BKCMP<P, BKCMP<>P, BKCMP<=P, and BKCMP>=P are put in.



2) A constant can be directly specified in the device specified by (s1).



3) The table below shows the comparison result in each instruction.

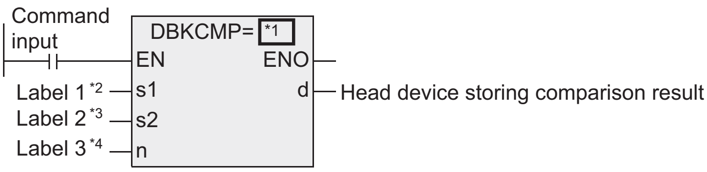
Instruction	Comparison result ON (1) condition	Comparison result OFF (0) condition
BKCMP=	(s1) = (s2)	(s1) ≠ (s2)
BKCMP>	(s1) > (s2)	(s1) ≤ (s2)
BKCMP<	(s1) < (s2)	(s1) ≥ (s2)
BKCMP<>	(s1) ≠ (s2)	(s1) = (s2)
BKCMP<=	(s1) ≤ (s2)	(s1) > (s2)
BKCMP>=	(s1) ≥ (s2)	(s1) < (s2)

4) When the comparison result is ON (1) in all of "n" points starting from the device specified by (d), M8090 (block comparison signal) turns ON.

2. 32-bit operation

(DBKCMP=, DBKCMP>, DBKCMP<, DBKCMP<>, DBKCMP<=, DBKCMP>=, DBKCMP=P, DBKCMP>P, DBKCMP<P, DBKCMP<>P, DBKCMP<=P, DBKCMP>=P)

1) "2n" 32-bit binary data starting from the device specified by s1 are compared with "2n" 32-bit binary data starting from the device specified by s, and the comparison result is stored in "2n" points starting from the device specified by d.

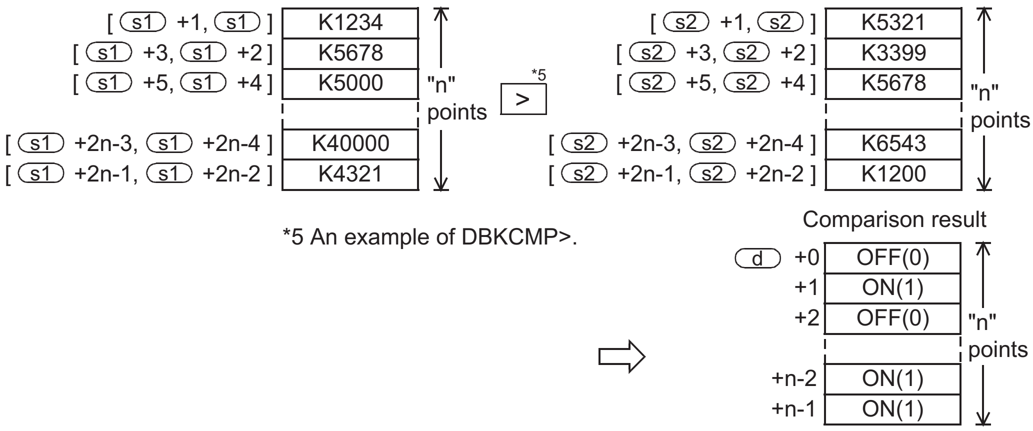


*1. DBKCMP=, DBKCMP>, DBKCMP<, DBKCMP<>, DBKCMP<=, DBKCMP>=, DBKCMP=P, DBKCMP>P, DBKCMP<P, DBKCMP<>P, DBKCMP<=P, and DBKCMP>=P are put in.

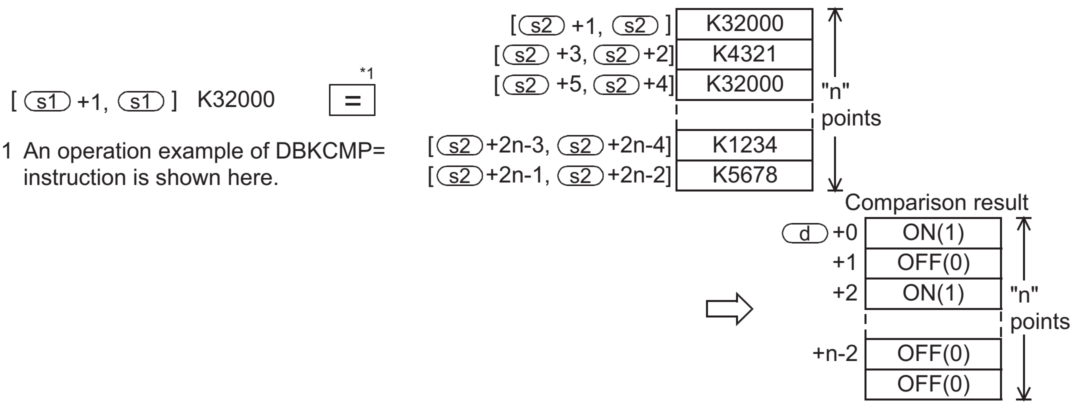
*2. This defines the comparison value or the device storing the comparison value.

*3. This defines the head of the device that stores the comparison source data.

*4. This defines the number of pieces of comparison data



2) A constant can be directly specified in the device specified by s1.



3) The table below shows the comparison result for each instruction.

Instruction	Comparison result ON (1) condition	Comparison result OFF (0) condition
DBKCMP=	[s1 +1, s1] = [s2 +1, s2]	[s1 +1, s1] ≠ [s2 +1, s2]
DBKCMP>	[s1 +1, s1] > [s2 +1, s2]	[s1 +1, s1] ≤ [s2 +1, s2]
DBKCMP<	[s1 +1, s1] < [s2 +1, s2]	[s1 +1, s1] ≥ [s2 +1, s2]
DBKCMP<>	s1 +1, s1] ≠ [s2 +1, s2]	[s1 +1, s1] = [s2 +1, s2]
DBKCMP<=	[s1 +1, s1] ≤ [s2 +1, s2]	[s1 +1, s1] > [s2 +1, s2]
DBKCMP>=	[s1 +1, s1] ≥ [s2 +1, s2]	[s1 +1, s1] < [s2 +1, s2]

4) When the comparison result is ON (1) in all of "n" points starting from the device specified by d, the M8090 (block comparison signal) turns ON.

Related device

Device	Name	Description
M8090	Block comparison signal	Turns ON when all comparison results are "ON (1)" in a block data instruction. DBKCMP=, DBKCMP>, DBKCMP<, DBKCMP<>, DBKCMP<=, DBKCMP>=

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
2) The FX3UC PLC of V. 2.20 or later supports this instruction.
3) When using 32-bit counters (high speed counters)
For comparing 32-bit counters (C200 to C255), be sure to use an instruction for 32-bit operation (such as DBKCMP= and DBKCMP>).
If an instruction for 16-bit operation (such as BKCMP= and BKCMP>) is used, an operation error is caused (error code: K6705)
4) Some restrictions to applicable devices
▲: D□.b cannot be indexed (v, z).

Error

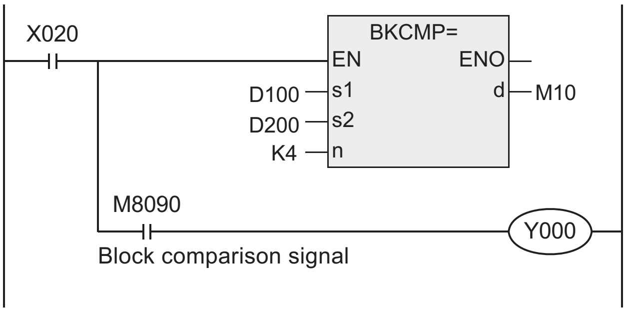
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "n" ("2n" in 32-bit operation) devices starting from the devices specified by (s1), (s2) and/or (d) exceed the corresponding device range (error code: K6706).
2) When data registers starting from the device specified by (d) specified as "D□.b" overlap "n" ("2n" in 32-bit operation) points starting from the device specified by (s1) (error code: K6706).
3) When data registers starting from the device specified by (d) specified as "D□.b" overlap "n" ("2n" in 32-bit operation) points starting from the device specified by (s2) (error code: K6706).
4) When a 32-bit counter (C200 to C255) is specified in the devices specified by (s1) and/or (s2) in 16-bit operation (error code: K6705).
For comparing 32-bit counters, be sure to use an instruction for 32-bit operation (such as DBKCMP=, DBKCMP> and DBKCMP<).

Program examples

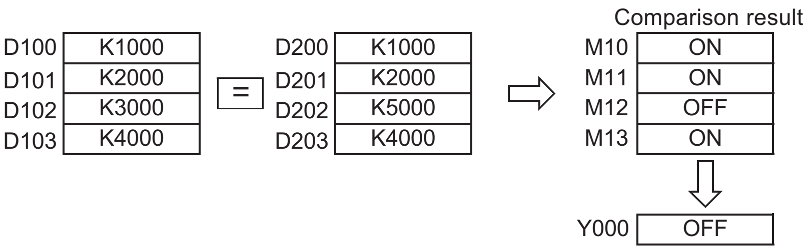
1) In the program shown below, four 16-bit binary data starting from D100 are compared with four 16-bit binary data starting from D200 by BKCMP= instruction when X020 is set to ON, and the comparison result is stored in four points starting from M10.
When the comparison result is "ON (1)" in all of the four points starting from M10, Y000 is set to ON.

[Structured ladder]



[ST]

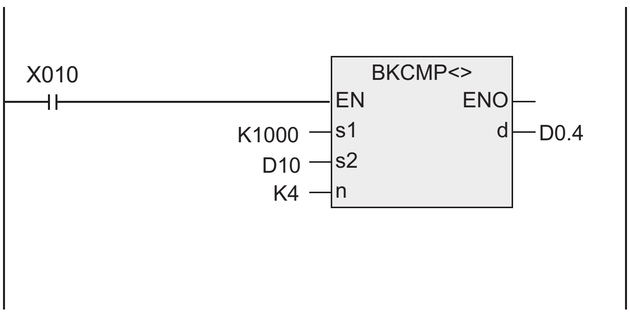
```
BKCMP=(X020,D100,D200,K4,M10);
Y000:=M8090;
```



(When all of M10 to M13 are ON, Y000 is set to ON.)

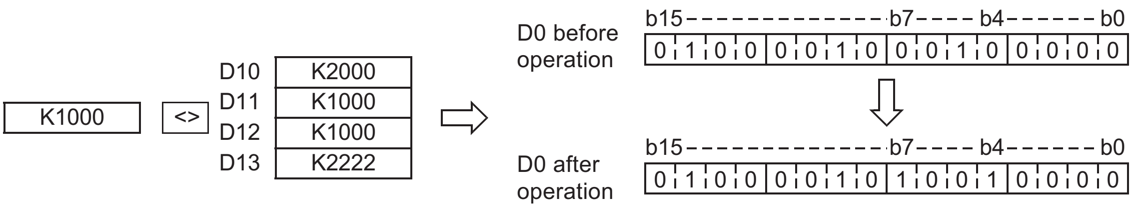
2) In the program shown below, the constant K1000 is compared with four data starting from D10 when X010 is set to ON, and the comparison result is stored in b4 to b7 of D0.

[Structured ladder]



[ST]

```
BKCMP<>(X010,K1000,D10,K4,D0.4);
Y000:=M8090;
```



7.20 Character String Control

7.20.1 STR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction converts binary data into character strings (ASCII codes).
On the other hand, the ESTR instruction converts floating point data into character strings.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
STR	16 bits	Continuous	STR EN ENO s1 d s2	STR(EN,s1,s2,d);
STRP	16 bits	Pulse	STRP EN ENO s1 d s2	STRP(EN,s1,s2,d);
DSTR	32 bits	Continuous	DSTR EN ENO s1 d s2	DSTR(EN,s1,s2,d);
DSTRP	32 bits	Pulse	DSTRP EN ENO s1 d s2	DSTRP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Head device storing the number of digits of a numeric value to be converted	ANY16	ANY16
	s2	Device storing binary data to be converted	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head device storing converted character string	String	String

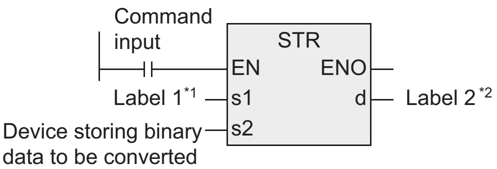
3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit specification				System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	X	Y	Modifier	K	H	E	"□"	P
s1												●	●	●	●				●					
s2								●	●	●	●	●	●	●	●		●	●	●	●	●			
d												●	●	●	●				●					

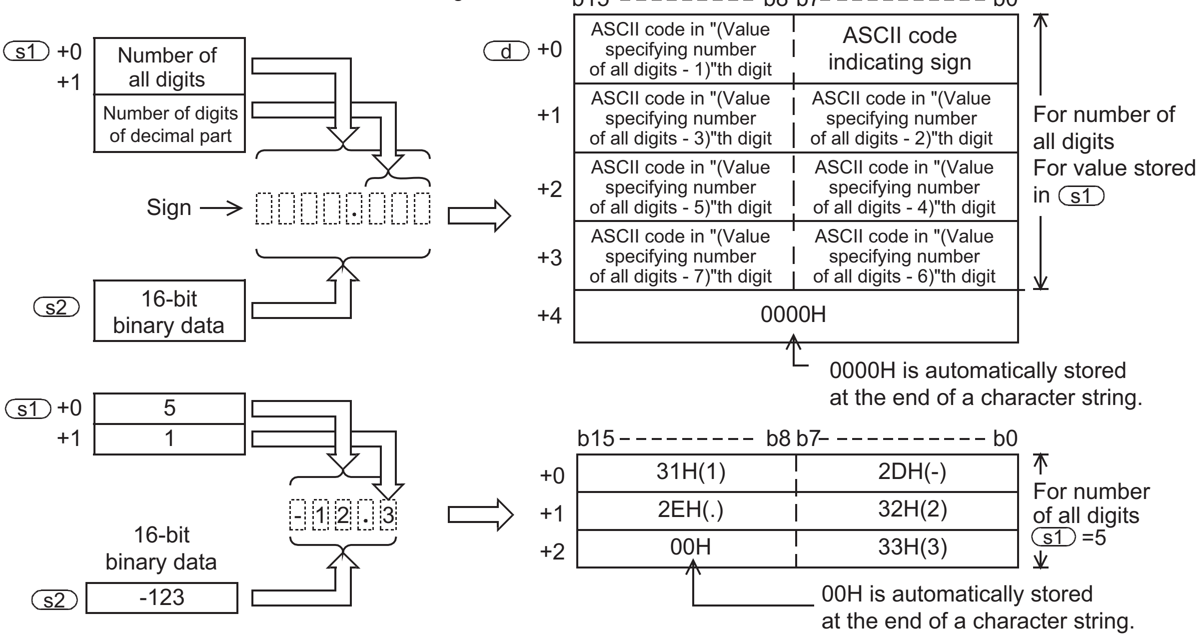
Function and operation explanation

1. 16-bit operation (STR/STRP)

1) All digits (specified by (s1)) of 16-bit binary data of the device specified by (s2) are converted into character string while the decimal point is added to the position specified by the device storing the number of digits of the decimal part ((s1) + 1), and stored in the device specified by (d) and later.



*1. This defines the head of the device that stores the number of digits of the value to be converted.
*2. This defines the head of the device that stores the data converted into character strings.

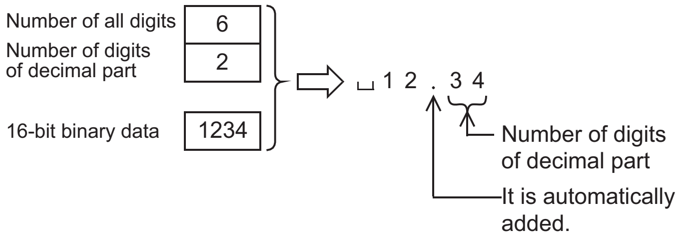


2) Set the number of all digits (s1) in the range from 2 to 8.
3) Set the number of digits of the decimal part (s1) + 1 in the range from 0 to 5.
Be sure to satisfy "(number of digits of decimal part) ≤ (number of all digits - 3)".
4) 16-bit binary data to be converted stored in (s2) should be within the range from -32768 to 32767.

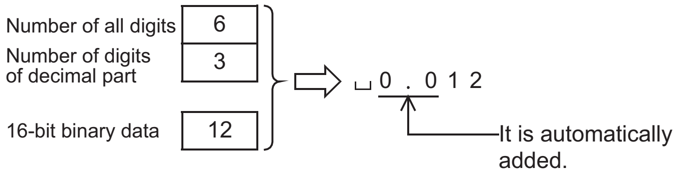
5) Converted character string data is stored in (d) and later as shown below.

a) As the sign, "space" (20H) is stored when the 16-bit binary data stored in (s2) is positive, and "-" (2DH) is stored when the 16-bit binary data stored in (s2) is negative.

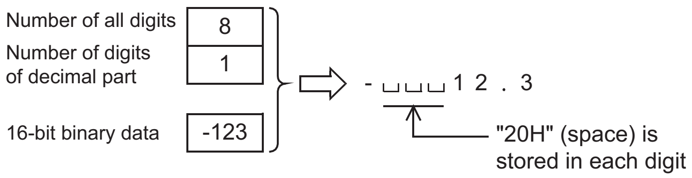
b) When the number of digits of the decimal part (s1) + 1 is set to any value other than "0", the decimal point "." (2EH) is automatically added in "number of digits of decimal part + 1"th digit.
When the number of digits of the decimal part (s1) + 1 is set to "0", the decimal point is not added.



c) When the number of digits of the decimal part (s1) + 1 is larger than the number of digits of 16-bit binary data stored in (s2), "0" (30H) is automatically added, and the data is shifted to the right end during conversion.



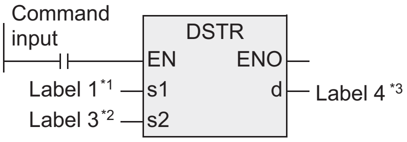
d) When the number of all digits stored in (s1) excluding the sign and decimal point is larger than the number of digits of 16-bit binary data stored in (s2), "space" (20H) is stored in each digit between the sign and the numeric value.
When the number of all digits stored in (s1) excluding the sign and decimal point is smaller than the number of digits of 16-bit binary data stored in (s2), an error is caused.



e) "00H" indicating the end of a character string is automatically stored at the end of a converted character string.
When the number of all digits is even, "0000H" is stored in the device after the last character.
When the number of all digits is odd, "00H" is stored in the high-order byte (8 bits) of the device storing the last character.

2. 32-bit operation (DSTR/DSTRP)

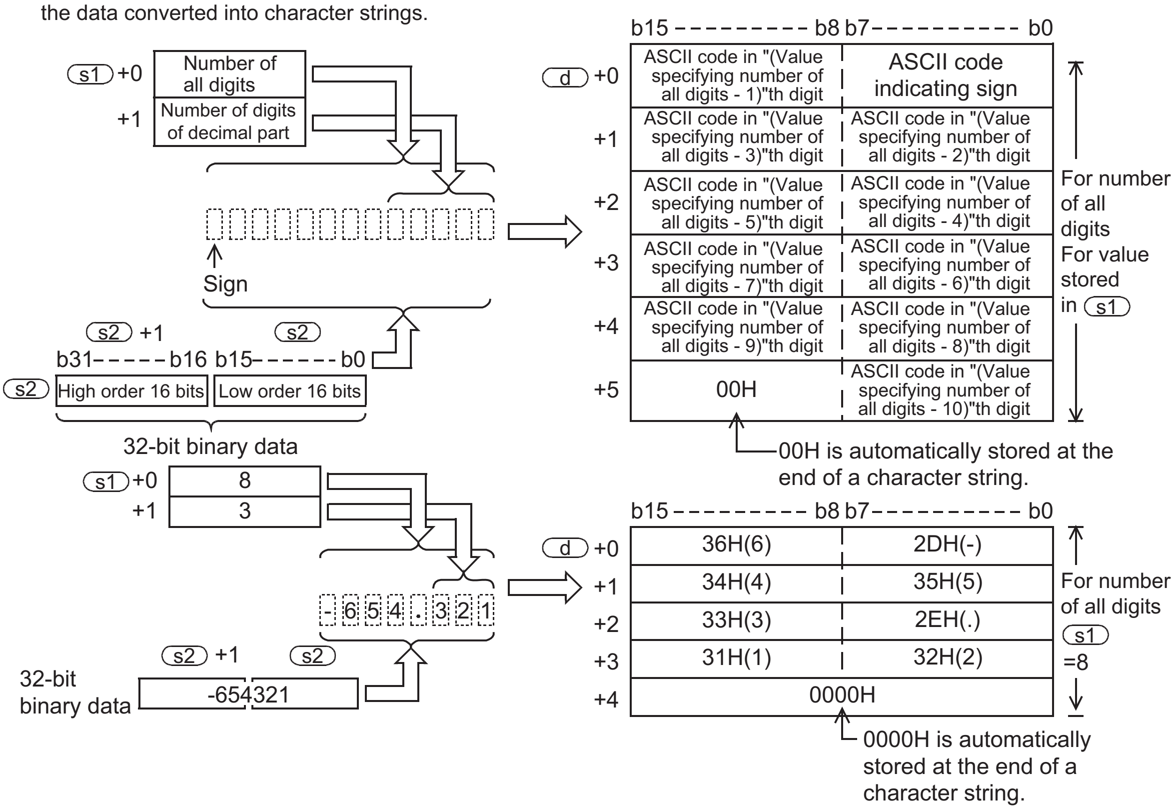
1) All digits (specified by (s1)) of 32-bit binary data stored in the device specified by (s2) are converted into ASCII codes while the decimal point is added to the position specified by the device storing the number of digits of the decimal part ((s1) + 1), and stored in the device specified by (d) and later.



*1. This defines the head of the device that stores the number of digits of the value to be converted.

*2. This defines the device that stores the binary data to be converted.

*3. This defines the head of the device that stores the data converted into character strings.



2) Set the number of all digits (s1) in the range from 2 to 13.

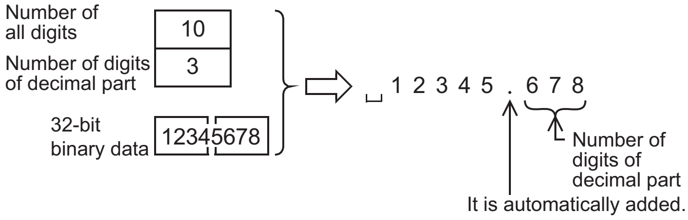
3) Set the number of digits of the decimal part (s1) + 1 in the range from 0 to 10.
Be sure to satisfy "(Number of digits of decimal part) ≤ (Number of all digits - 3)".

4) 32-bit binary data to be converted stored in (s2) should be within the range from -2,147,483,648 to +2,147,483,647.

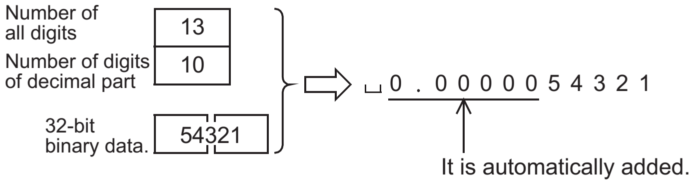
5) Converted character string data is stored in (d) and later as shown below.

a) For the sign, "space" (20H) is stored when the 32-bit binary data stored in (s2) is positive, and "-" (2DH) is stored when the 32-bit binary data stored in (s2) is negative.

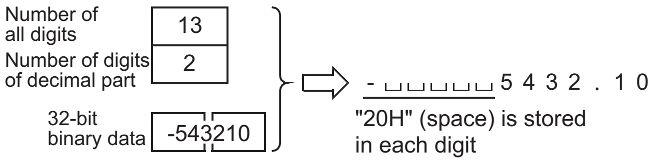
b) When the number of digits of the decimal part (s1) + 1 is set to any value other than "0", the decimal point "." (2EH) is automatically added in "number of digits of decimal part + 1"th digit.
When the number of digits of the decimal part (s1) + 1 is set to "0", the decimal point is not added.



c) When the number of digits of the decimal part (s1) + 1 is larger than the number of digits of 32-bit binary data stored in (s2), "0" (30H) is automatically added, and the data is shifted to the right end during conversion.



d) When the number of all digits stored in (s1) excluding the sign and decimal point is larger than the number of digits of 32-bit binary data stored in (s2), "space" (20H) is stored in each digit between the sign and the numeric value.
When the number of all digits stored in (s1) excluding the sign and decimal point is smaller than the number of digits of binary data stored in (s2), an error is caused.



e) "00H" indicating the end of a character string is automatically stored at the end of a converted character string.
When the number of all digits is even, "0000H" is stored in the device after the last character.
When the number of all digits is odd, "00H" is stored in the high-order byte (8 bits) of the device storing the last character.

Cautions

1) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle array data or 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

2) The FX3UC PLC of V. 2.20 or later supports this instruction.

Related instruction

Instruction	Description
DESTR	Converts binary floating point data into a character string (ASCII codes) with a specified number of digits.
DEVAL	Converts a character string (ASCII codes) into binary floating point data.
VAL	Converts a character string (ASCII codes) into binary data.

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the number of all digits stored in (s1) is outside the following range (error code: K6706).

	Setting range
16-bit operation	2 to 8
32-bit operation	2 to 13

2) When the number of digits of the decimal part stored in (s1) + 1 is outside the following range (error code: K6706).

	Setting range
16-bit operation	0 to 5
32-bit operation	0 to 10

3) When the relationship between the number of all digits stored in (s1) and the number of digits of the decimal part stored in (s1) + 1 does not satisfy the following (error code: K6706).
(Number of all digits - 3) ≥ number of digits of decimal part

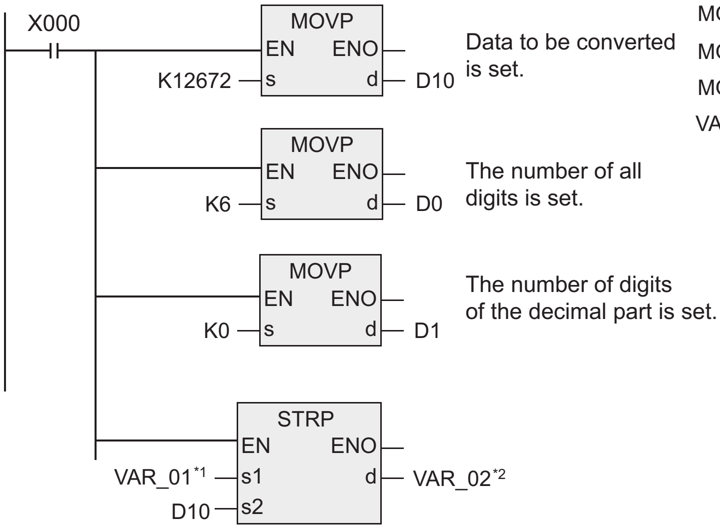
4) When the number of all digits stored in (s1) including the digit for sign and the digit for decimal point is smaller than the number of digits of the binary data stored in (s2) (error code: K6706).

5) When the devices (d) and later storing a character string exceed the corresponding device range (error code: K6707).

Program examples

In the program below, the 16-bit binary data stored in D10 is converted into a character string in accordance with the digit specification by D0 and D1 when X000 is set to ON, and then stored in D20 to D23.

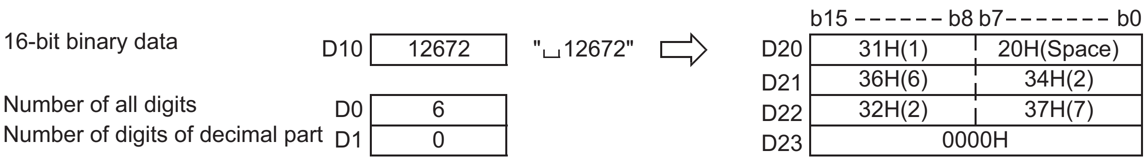
[Structured ladder]



[ST]

```
MOVP(X000,K12672,D10);
MOVP(X000,K6,D0);
MOVP(X000,K0,D1);
VAR_02:=STRP(X000,VAR_01,D10);
```

*1. VAR_01 is a global label and is defined as D0.
*2. VAR_02 is a global label and is defined as D20.



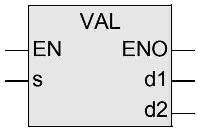
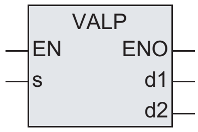
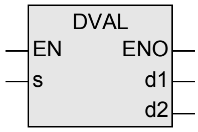
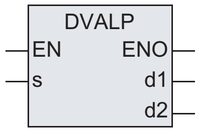
7.20.2 VAL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction converts a character string (ASCII codes) into binary data.
On the other hand, EVAL instruction converts a character string (ASCII codes) into floating point data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
VAL	16 bits	Continuous		VAL(EN,s,d1,d2);
VALP	16 bits	Pulse		VALP(EN,s,d1,d2);
DVAL	32 bits	Continuous		DVAL(EN,s,d1,d2);
DVALP	32 bits	Pulse		DVALP(EN,s,d1,d2);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s	Head device storing a character string to be converted into binary data.	String	String
Output variable	ENO	Execution state	Bit	
	d1	Head device storing the number of digits of the binary data acquired by conversion.	ANY16	ANY16
	d2	Head device storing the binary data acquired by conversion.	ANY16	ANY32

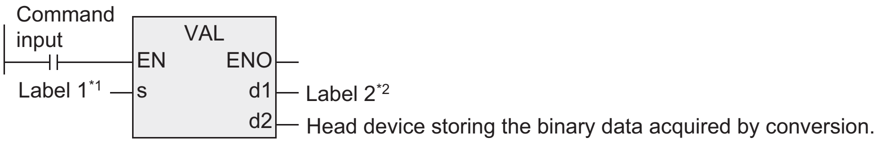
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	X	Y	Modifier	K	H	E	"□"	P
s												●	●	●	●				●					
d1												●	●	●	●				●					
d2								●	●	●	●	●	●	●	●	●			●					

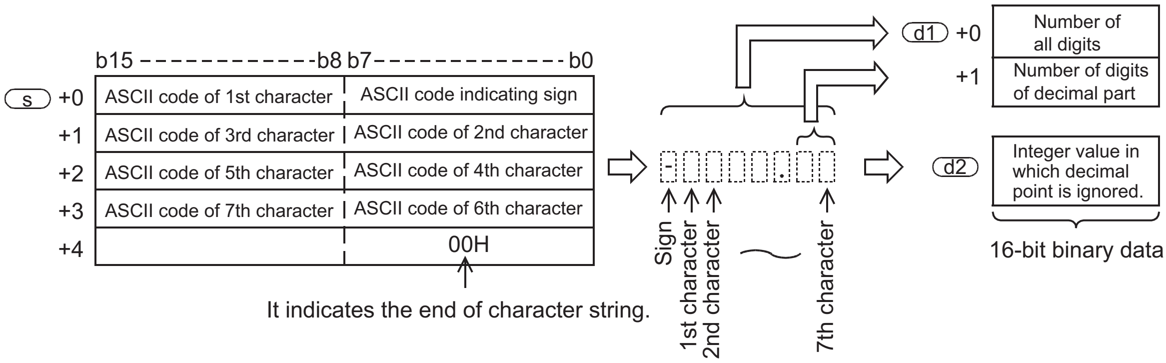
Function and operation explanation

1. 16-bit operation (VAL/VALP)

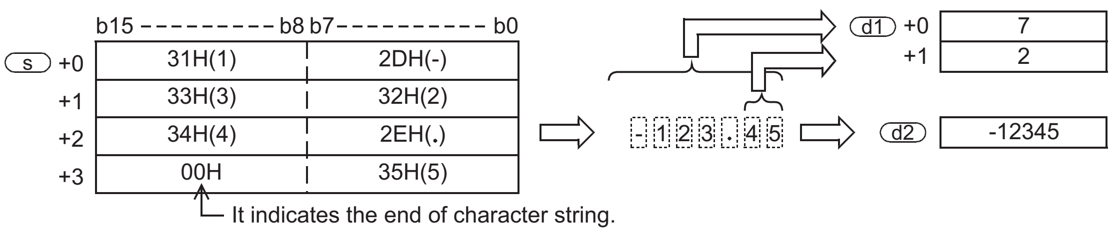
1) A character string stored in the device specified by (s) and later is converted into 16-bit binary data. The number of all digits of the binary data acquired by conversion is stored in the device specified by (d1), the number of digits of the decimal part is stored in the device specified by (d1) + 1, and the converted binary data is stored in the device specified by (d2).
In conversion from a character string into binary data, the data from the device specified by (s) to a device number storing "00H" is handled as a character string in byte units.



*1. This defines the head of the device that stores the character string to be converted into binary data.
*2. This defines the head of the device that stores the number of digits of the binary data converted.



For example, when a character string "-123.45" is specified in the device specified by (s) and later, the conversion result is stored in the devices specified by (d1) and (d2) as shown below.



2) Character string to be converted

a) Number of characters of character string and the numeric range when the decimal point is ignored.

	Description
Number of all characters (digits)	2 to 8
Number of characters (digits) of decimal part	0 to 5 and smaller than "number of all digits - 3"
Numeric range when decimal point is ignored	-32768 to 32767 Example) "123.45" → "12345"

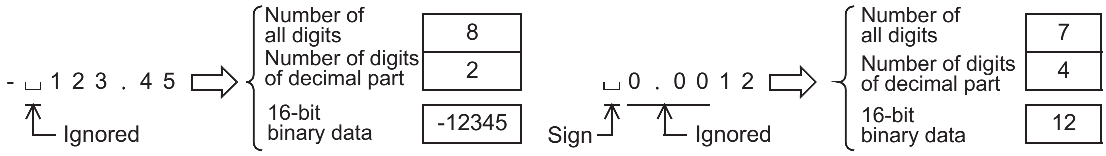
b) Character types used in characters to be converted

		Character type
Sign	Positive numeric value	"Space (20H)"
	Negative numeric value	"-(2DH)"
Decimal point		".(2EH)"
Number		"0(30H)" to 9(39H)"

3) The device specified by (d1) stores the number of all digits. The number of all digits indicates the number of all characters (including the number, sign and decimal point).

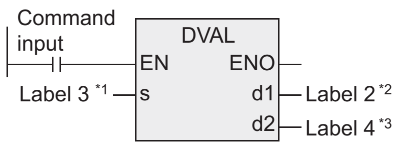
4) The device specified by (d1) + 1 stores the number of digits of the decimal part. The number of digits of the decimal part indicates the number of all characters after the decimal point "." (2EH).

5) The device specified by (d2) stores 16-bit binary data converted from a character string with the decimal point ignored.
In the character string located in the device specified by (s) and later, "space" (20H) and "0" (30H) characters between the sign and the first number other than "0" are ignored in the conversion to 16-bit binary data.



2. 32-bit operation (DVAL/DVALP)

1) A character string stored in the device specified by (s) and later is converted into 32-bit binary data. The number of all digits of the binary data acquired by conversion is stored in the device specified by (d1), the number of digits of the decimal part is stored in the device specified by (d1) + 1, and the binary data is stored in the device specified by (d2).
In conversion from a character string into binary data, the data from the device specified by (s) to a device number storing "00H" is handled as a character string in byte units.



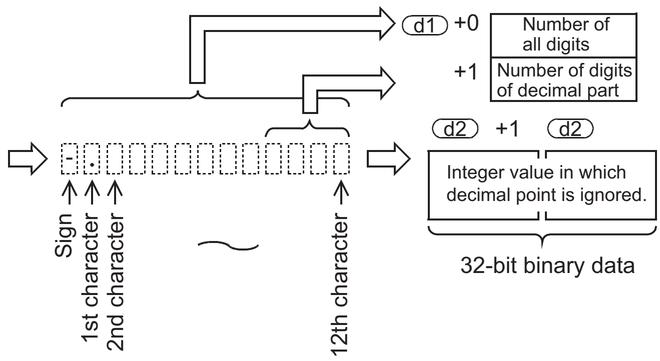
*1. This defines the head of the device that stores the character string to be converted into binary data.

*2. This defines the head of the device that stores the number of digits of the binary data converted.

*3. This defines the head of the device that stores the binary data converted.

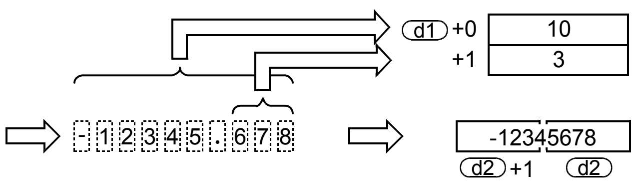
	b15 - - - b8	b7 - - - b0
(s) +0	ASCII code of 1st character	ASCII code indicating sign
+1	ASCII code of 3rd character	ASCII code of 2nd character
+2	ASCII code of 5th character	ASCII code of 4th character
+3	ASCII code of 7th character	ASCII code of 6th character
+4	ASCII code of 9th character	ASCII code of 8th character
+5	ASCII code of 11th character	ASCII code of 10th character
+6	00H	ASCII code of 12th character

It indicates the end of character string.



For example, when a character string "-12345.678" is specified in the device specified by (s) and later, the conversion result is stored in the devices specified by (d1) and (d2) as shown below.

	b15 - - - b8	b7 - - - b0
(s) +0	31H(1)	2DH(-)
+1	33H(3)	32H(2)
+2	35H(5)	34H(4)
+3	36H(6)	2EH(.)
+4	38H(8)	37H(7)
+5		00H



2) Character string to be converted

a) Number of characters of character string and the numeric range when the decimal point is ignored.

	Description
Number of all characters (digits)	2 to 13
Number of characters (digits) of decimal part	0 to 10 and smaller than "number of all digits - 3"
Numeric range when decimal point is ignored	-2,147,483,648 to 2,147,483,647 Example) "12345.678" to "12345678"

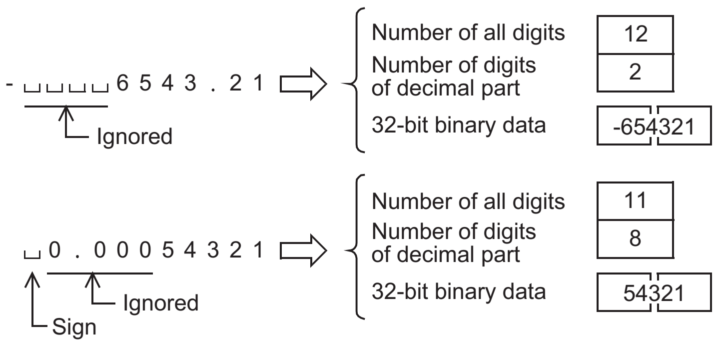
b) Character types used in characters to be converted

		Character type
Sign	Positive numeric value	"Space (20H)"
	Negative numeric value	"-(2DH)"
Decimal point		".(2EH)"
Number		"0(30H)" to "9(39H)"

3) The device specified by (d1) stores the number of all digits. The number of all digits indicates the number of all characters (including the number, sign and decimal point).

4) The device specified by (d1) + 1 stores the number of digits of the decimal part. The number of digits of the decimal part indicates the number of all characters after the decimal point "." (2EH).

5) The device specified by (d2) stores 32-bit binary data converted from a character string with the decimal point ignored.
For the character string located in the device specified by (s) and later, the "space" (20H) and "0" (30H) characters between the sign and the first number other than "0" are ignored in the conversion to 32-bit binary data.



Related instruction

Instruction	Description
DESTR	Converts binary floating point data into a character string (ASCII codes) with a specified number of digits.
DEVAL	Converts a character string (ASCII codes) into binary floating point data.
STR	Converts binary data into a character string (ASCII codes).

Cautions

1) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle array data or 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

2) The FX3UC PLC of V. 2.20 or later supports this instruction.

3) Sign data "space (20H)" or "- (2DH)" must be stored in the first byte (lower order 8 bits of the head device set in the device specified by (s)).
Only the ASCII code data "0 (30H)" to "9 (39H)", "space (20H)" and "decimal point (2EH)" can be stored from the second byte to the "00H" at the end of the character string of the device specified by (s).
If "- (2DH)" is stored in the second byte or later, an operation error occurs (error code: K6706).

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the number of characters of the character string to be converted (device specified by (S) and later) is outside the following ranges.
(Error code: K6706)

	Setting range
16-bit operation	2 to 8
32-bit operation	2 to 13

2) When the number of characters after the decimal point of the character string to be converted (device specified by (S) and later) is outside the following ranges.
(Error code: K6706)

	Setting range
16-bit operation	0 to 5
32-bit operation	0 to 10

3) When the relationship between the number of all characters in the character string to be converted (device specified by (S) and later) and the number of characters after the decimal point does not satisfy the following (error code: K6706).
(Number of all characters - 3) ≥ Number of characters after the decimal point
4) When the sign is set to any ASCII code other than "space" (20H) and "-" (2DH).
(Error code: K6706)
5) When a digit of a number is set to any ASCII code other than "0" (30H) to "9" (39H) or a decimal point "." (2EH).
(Error code: K6706)
6) When the decimal point "." (2EH) is set two or more times in the character string to be converted (device specified by (S) and later).
(Error code: K6706)
7) When the binary data acquired by conversion is outside the following ranges. (Error code: K6706)

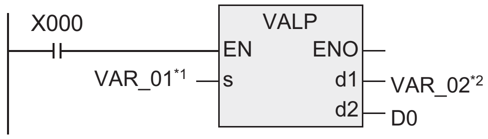
	Setting range
16-bit operation	-32768 to 32767
32-bit operation	-2,147,483,648 to 2,147,483,647

8) When "00H" is not present in the location from the device specified by (S) to the final device number.
(Error code: K6706)

Program examples

1) In the program below, the character string data stored in D20 to D22 is regarded as an integer value, converted into a binary value, and stored in D0 when X000 is set to ON.

[Structured ladder]

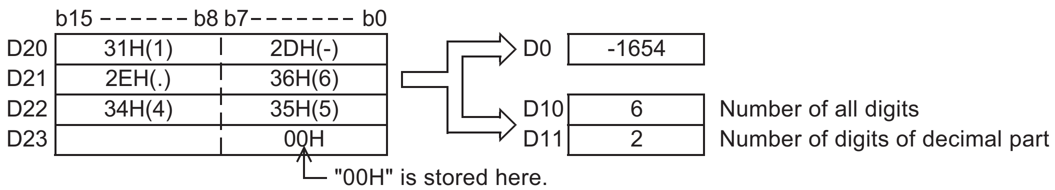


[ST]

```
VAR_02:=VALP(X000,VAR_01);
D0:=VALP(X000,VAR_01);
```

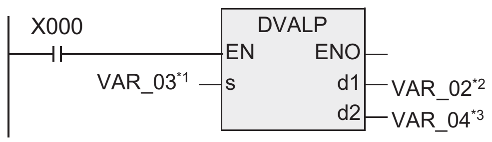
*1. VAR_01 is a global label and is defined as D20.

*2. VAR_02 is a global label and is defined as D10.



2) In the program below, the character string data stored in D20 to D24 is regarded as an integer value, converted into a binary value, and stored in D0 when X000 is set to ON.

[Structured ladder]



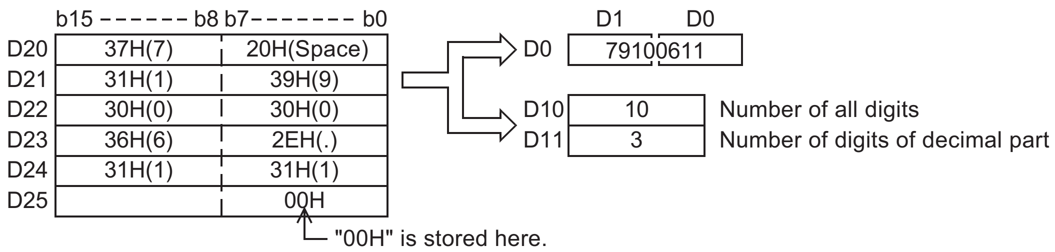
[ST]

```
VAR_02:=DVALP(X000,VAR_03);
VAR_04:=DVALP(X000,VAR_03);
```

*1. VAR_03 is a global label and is defined as D20.

*2. VAR_02 is a global label and is defined as D10.

*3. VAR_04 is a global label and is defined as D0.



7.20.3 $+

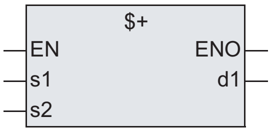
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

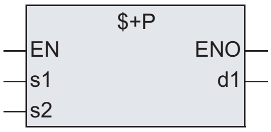
Outline

This instruction links a character string to another character string.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
$+	16 bits	Continuous	(see diagram 1 below)	$+(EN,s1,s2,d1);
$+P	16 bits	Pulse	(see diagram 2 below)	$+P(EN,s1,s2,d1);





2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Head device storing the link source data (character string) or directly specified character string	String
	s2	Head device storing the link data (character string) or directly specified character string	String
Output variable	ENO	Execution state	Bit
	d1	Head device storing the linked data (character string)	String

3. Applicable devices

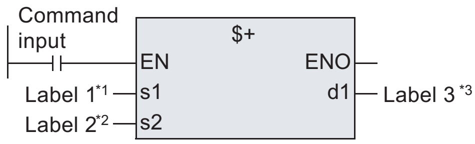
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	X	Y	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	●	●			●				●	
s2								●	●	●	●	●	●	●	●	●			●				●	
d1									●	●	●	●	●	●	●	●			●					

Function and operation explanation

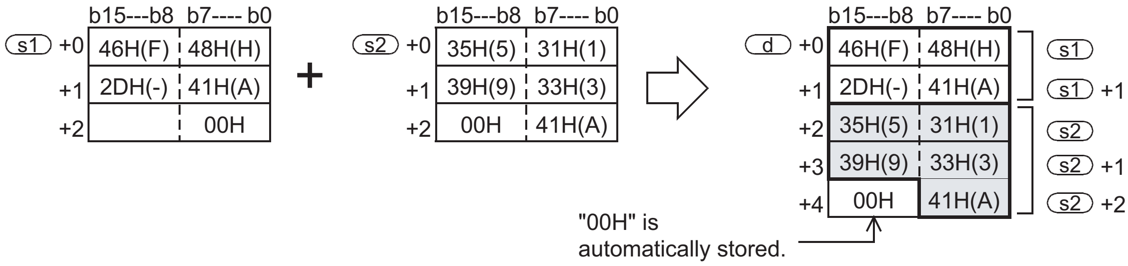
1. 16-bit operation ($+/$+P)

The character string data stored in the device specified by s2 and later is linked to the end of the character string data stored in the device specified by s1 and later, and the linked data is stored to devices starting from the device specified by d.

A character string specified by s1 and s2 indicates the data from the specified device to the first "00H" in units of byte.



*1. This defines the head of the device that stores the link source data (character string) or defines the directly specified character string.
*2 This defines the head of the device that stores the link data (character string) or defines the directly specified character string.
*3. This defines the head of the device that stores the linked data (character string).



1) In linking, "00H" indicating the end of a character string specified in s1 is ignored, and a character string specified in s2 is linked to the last character specified in s1.
When a character string is linked, "00H" is automatically added at the end.
   a) When the number of characters after linking is odd, "00H" is stored in the high-order byte of the device storing the last character.
   b) When the number of characters after linking is even, "0000H" is stored in the device after the last character.

Cautions

1) When handling character string data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle character string data.
Use a global label to specify a device.
2) When directly specifying a character string, up to 32 characters can be specified (input).
However, this limitation in the number of characters is not applied when a word device is specified in s1 or s2.
3) When the number of characters in both devices specified by s1 and s2 start from "00H" (that is, when the number of characters is "0"), "0000H" is stored in the device specified by d.

Error

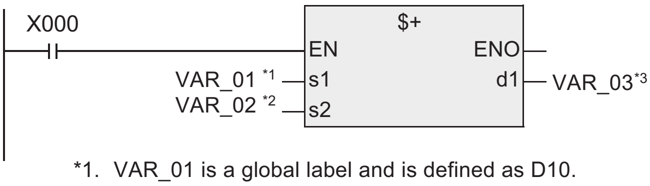
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the number of devices after a device number specified by d is smaller than the number of devices required to store all linked character strings (that is, when "00H" cannot be stored after all character strings and the last character).
(Error code: K6706)
2) When the same device is specified in s1, s2 and d as a device for storing a character string.
(Error code: K6706)
3) When "00H" is not set within the corresponding device range after the device specified by s1 or s2.
(Error code: K6706)

Program examples

In the program example shown below, a character string stored in D10 to D12 (abcde) is linked to the character string "ABCD", and the result is stored to D100 and later when X000 turns ON.

[Structured ladder]



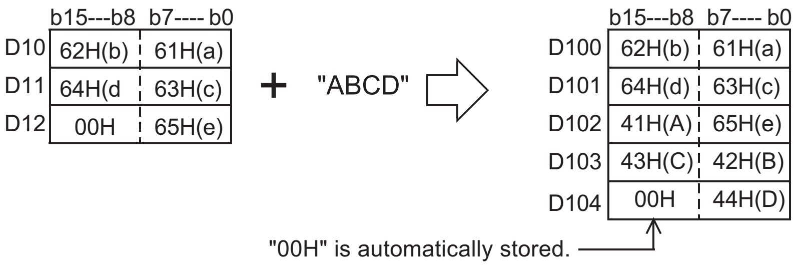
[ST]

```
VAR_03:=$+(X000,VAR_01,VAR_02);
```

*1. VAR_01 is a global label and is defined as D10.

*2. VAR_02 is a global label and is defined as "ABCD".

*3. VAR_03 is a global label and is defined as D100.



7.20.4 LEN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction detects the number of characters (bytes) of a specified character string.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	ST
LEN	16 bits	Continuous	LEN EN ENO s d	LEN(EN,s,d);
LENP	16 bits	Pulse	LENP EN ENO s d	LENP(EN,s,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device storing a character string whose number of characters is to be detected	String
Output variable	ENO	Execution state	Bit
	d	Device storing the detected character string length (number of bytes)	ANY16

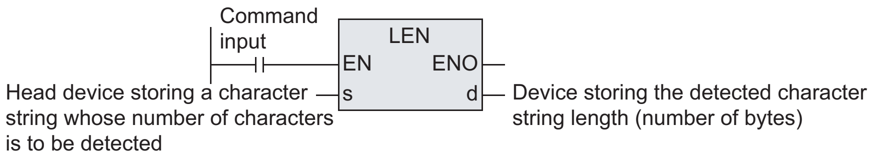
3. Applicable devices

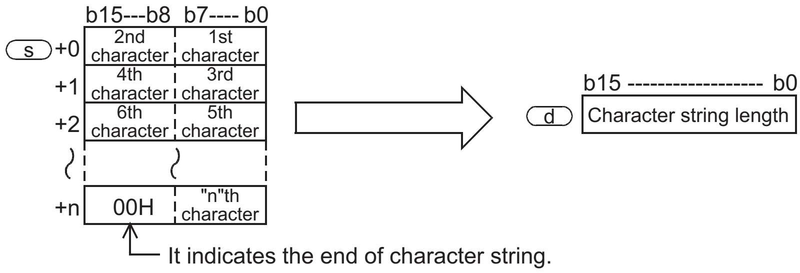
Operand type	Bit Devices: System user: X	Y	M	T	C	S	D□.b	Word Devices: Digit specification: KnX	KnY	KnM	KnS	System user: T	C	D	R	Special unit: U□\G□	Index: V	Z	Modifier	Others: Constant: K	H	Real Number: E	Character String: "□"	Pointer: P
s								●	●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●			●					

Function and operation explanation

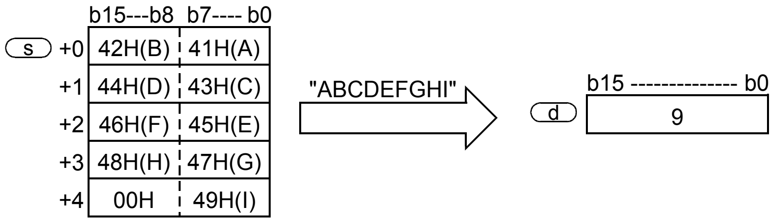
1. 16-bit operation (LEN/LENP)

The length of a character string stored in the device specified by (s) and later is detected, and stored to the device specified by (d). Data starting from the device specified by (s) until the first device storing "00H" is handled as a character string in units of byte.





For example, when "ABCDEFGHI" is stored in the device specified by (s) and later as shown below, K9 is stored in the device specified by (d).



Caution

1) This instruction can handle character codes other than ASCII codes, but the character string length is handled in byte units (8 bits). Accordingly, in the case of character codes in which two bytes express one character such as shift JIS codes, the length of one character is detected as "2".

Errors

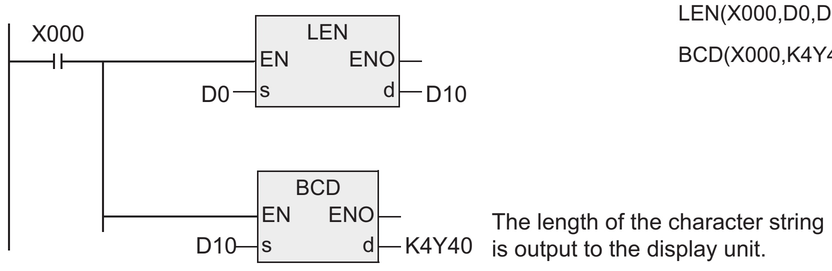
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "00H" is not set within the corresponding device range after a device specified by (s). (Error code: K6706)
2) When the detected number of characters is "32768" or more. (Error code: K6706)

Program examples

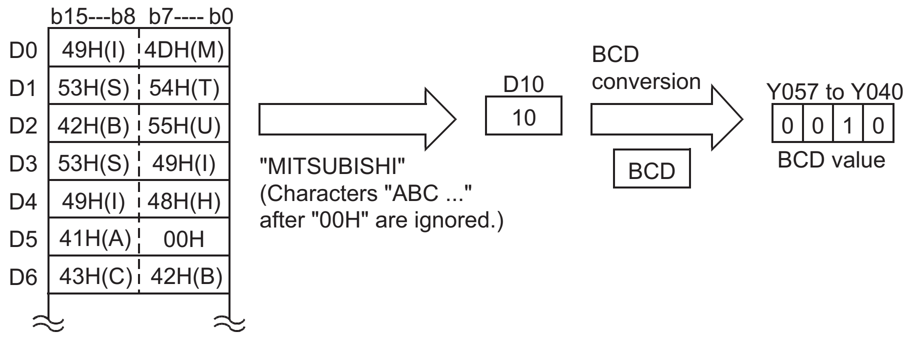
In the program example shown below, the length of a character string stored in D0 and later is output in 4-digit BCD to Y040 to Y057 when X000 turns ON.

[Structured ladder]



[ST]

```
LEN(X000,D0,D10);
BCD(X000,K4Y40);
```



7.20.5 RIGHT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction extracts a specified number of characters from the right end of a specified character string.

→ **For handling of character strings, refer to "FX Structured Programming Manual (Device & Common)."**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RIGHT	16 bits	Continuous	RIGHT EN ENO s d n	RIGHT(EN,s,n,d);
RIGHTP	16 bits	Pulse	RIGHTP EN ENO s d n	RIGHTP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device storing a character string	String
	n	Number of characters to be extracted	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device storing extracted character string	String

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●			●					
n														●	●					●	●			

Function and operation explanation

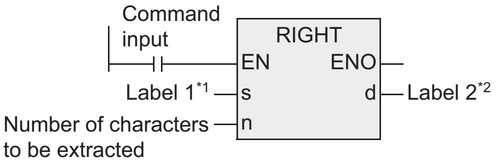
1. 16-bit operation (RIGHT/RIGHTP)

"n" characters are extracted from the right end (that is, from the end) of the character string data stored in the device specified by (s) and later, and stored to the device specified by (d) and later.

If the number of characters specified by "n" is "0", the NULL code (0000H) is stored to the device specified by (d).

When characters are extracted from a character string, "00H" is automatically added at the end of the extracted characters.

1) When the number of extracted characters is odd, "00H" is stored in the high-order byte of a device storing the last character.
2) When the number of extracted characters is even, "0000H" is stored in the device after the last character.



*1. This defines the head of the device that stores the number of characters.

*2. This defines the head of the device that stores the extracted character string.

(s)	b15 – b8	b7 – b0
+0	ASCII code for 2nd character	ASCII code for 1st character
+1	ASCII code for 4th character	ASCII code for 3rd character
	ASCII code for "(last character - n + 2)"th character	ASCII code for "(last character - n + 1)"th character
	ASCII code for "(last character - n + 4)"th character	ASCII code for "(last character - n + 3)"th character
	ASCII code for "(last character - 1)"th characte	ASCII code for "(last character - 2)"th character
	00H	ASCII code for last character

(d)	b15 – b8	b7 – b0
+0	ASCII code for "(last character - n + 2)"th character	ASCII code for "(last character - n + 1)"th character
+1	ASCII code for "(last character - n + 4)"th character	ASCII code for "(last character - n + 3)"th character
	ASCII code for "(last character - 1)"th character	ASCII code for "(last character - 2)"th character
	00H	ASCII code for last character

In the case of "n = 5"

(s)	b15 – b8	b7 – b0
+0	42H(B)	41H(A)
+1	44H(D)	43H(C)
+2	46H(F)	45H(E)
+3	32H(2)	31H(1)
+4	34H(4)	33H(3)
+5	00H	35H(5)

"ABCDEF12345"

ASCII code for 5th character

ASCII code for 1st character

(d)	b15 – b8	b7 – b0
+0	32H(2)	31H(1)
+1	34H(4)	33H(3)
+2	00H	35H(5)

"12345"

a) A character string stored in the device specified by (s) and later indicates data stored in devices from the specified device until "00H" is first detected in byte units.

Cautions

1) When handling character string data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle character string data.
   Use a global label to specify a device.
2) When handling character codes other than ASCII codes, note the following contents:
   a) The number of characters is handled in byte units (8 bits). Accordingly, in the case of character codes in which two bytes express one character such as shift JIS codes, the length of one character is detected as "2".
   b) When extracting characters from a character string including character codes in which two bytes express one character such as shift JIS codes, consider the number of characters to be extracted in units of character codes for one character.
      Note that the expected character code is not given if only one byte is extracted out of a 2-byte character code.

Error

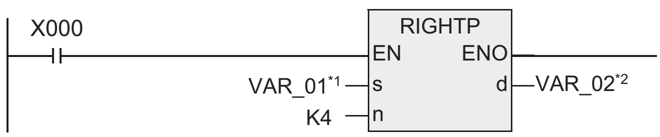
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "00H" is not set within the corresponding device range after a device specified by (s).
   (Error code: K6706)
2) When "n" exceeds the number of characters specified by (s). (Error code: K6706)
3) When the number of devices after a device number specified by (d) is smaller than the number of devices required to store extracted "n" characters (that is, when "00H" cannot be stored after all character strings and the last character).
   (Error code: K6706)
4) When "n" is a negative value. (Error code: K6706)

Program examples

In the program example shown below, 4 characters are extracted from the right end of the character string data stored in R0 and later, and stored to D0 and later when X000 turns ON.

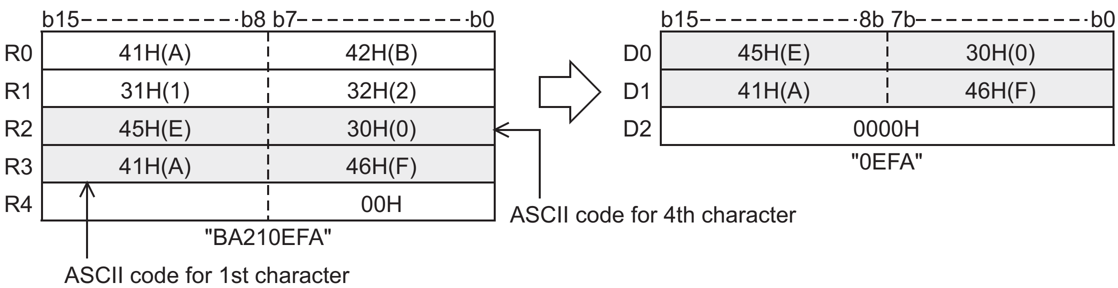
[Structured ladder]



[ST]

```
VAR_02:=RIGHTP(X000,K4);
```

*1. VAR_01 is a global label and is defined as R0.
*2. VAR_02 is a global label and is defined as D0.



7.20.6 LEFT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction extracts a specified number of characters from the left end of a specified character string.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
LEFT	16 bits	Continuous	LEFT EN ENO s d n	LEFT(EN,s,n,d);
LEFTP	16 bits	Pulse	LEFTP EN ENO s d n	LEFTP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device storing a character string	String
	n	Number of characters to be extracted	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device storing extracted character string	String

3. Applicable devices

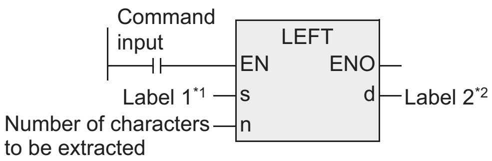
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	X	Y	M	K	H	E	"□"	P
s								●	●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●			●					
n														●	●					●	●			

Function and operation explanation

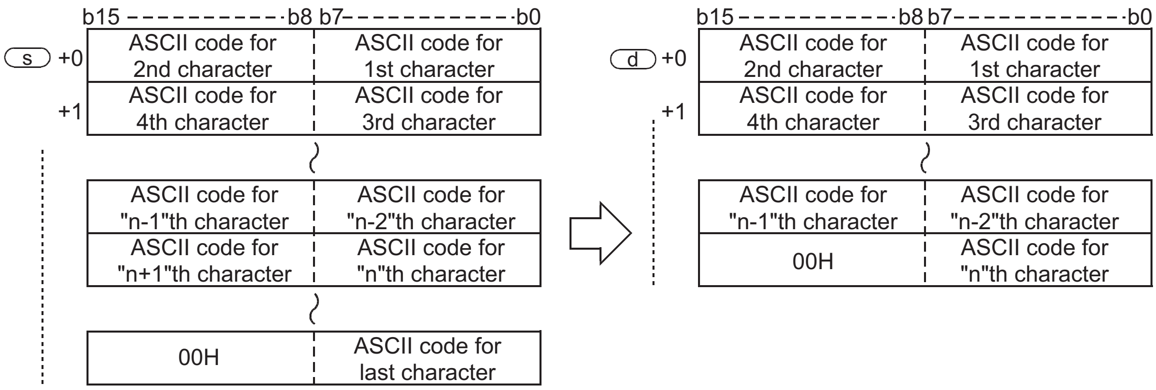
1. 16-bit operation (LEFT/LEFTP)

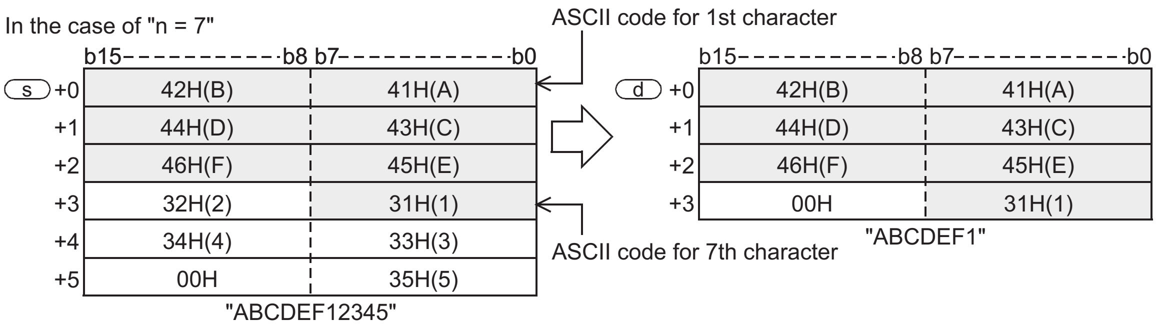
"n" characters are extracted from the left end (that is, from the head) of the character string data stored in the device specified by (s) and later and stored to the device specified by (d) and later.
If the number of characters specified by "n" is "0", the NULL code (0000H) is stored to the device specified by (d).
When characters are extracted from a character string, "00H" is automatically added at the end of the extracted characters.

1) When the number of extracted characters is odd, "00H" is stored in the high-order byte of a device storing the last character.
2) When the number of extracted characters is even, "0000H" is stored in the device after the last character.



*1. This defines the head of the device that stores the number of characters.
*2. This defines the head of the device that stores the extracted character string.





a) A character string stored in the device specified by (s) and later indicates data stored in devices from the specified device until "00H" is first detected in byte units.

Cautions

1) When handling character string data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle character string data.
   Use a global label to specify a device.
2) When handling character codes other than ASCII codes, note the following contents:
   a) The number of characters is handled in byte units (8 bits). Accordingly, in the case of character codes in which two bytes express one character such as shift JIS codes, the length of one character is detected as "2".
   b) When extracting characters from a character string including character codes in which two bytes express one character such as shift JIS codes, consider the number of characters to be extracted in units of character codes for one character.
      Note that the expected character code is not given if only one byte is extracted out of a 2-byte character code.

Error

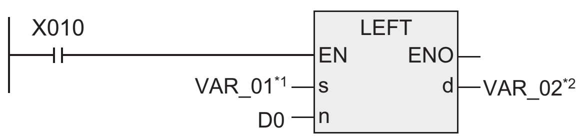
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "00H" is not set within the corresponding device range after a device specified by (s).
   (Error code: K6706)
2) When "n" exceeds the number of characters specified by (s). (Error code: K6706)
3) When the number of devices after a device number specified by (d) is smaller than the number of devices required to store extracted "n" characters (that is, when "00H" cannot be stored after all character strings and the last character). (Error code: K6706)
4) When "n" is a negative value. (Error code: K6706)

Program examples

In the program example shown below, the number of characters which is equivalent to the number stored in D0 is extracted from the left end of the character string data stored in D100 and later, and stored to R10 and later when X010 turns ON.

[Structured ladder]

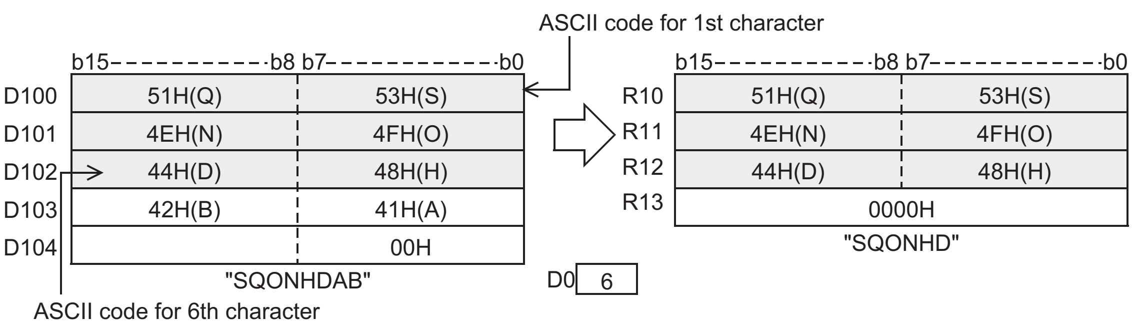


[ST]

```
VAR_02:=LEFT(X010,VAR_01,D0);
```

*1. VAR_01 is a global label and is defined as D100.

*2. VAR_02 is a global label and is defined as R10.



7.20.7 MIDR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction extracts a specified number of characters from arbitrary positions of a specified character string.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
MIDR	16 bits	Continuous	MIDR EN ENO s1 d s2	MIDR(EN,s1,s2,d);
MIDRP	16 bits	Pulse	MIDRP EN ENO s1 d s2	MIDRP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Head device storing a character string	String
	s2	Head device specifying the head position and number of characters to be extracted • s2 : Head character position • s2+1 : Number of characters	ARRAY [1..2] OF ANY16
Output variable	ENO	Execution state	Bit
	d	Head device storing extracted character string	String

3. Applicable devices

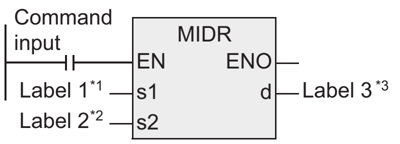
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●			●					
s2								●	●	●	●	●	●	●	●	●			●					

Function and operation explanation

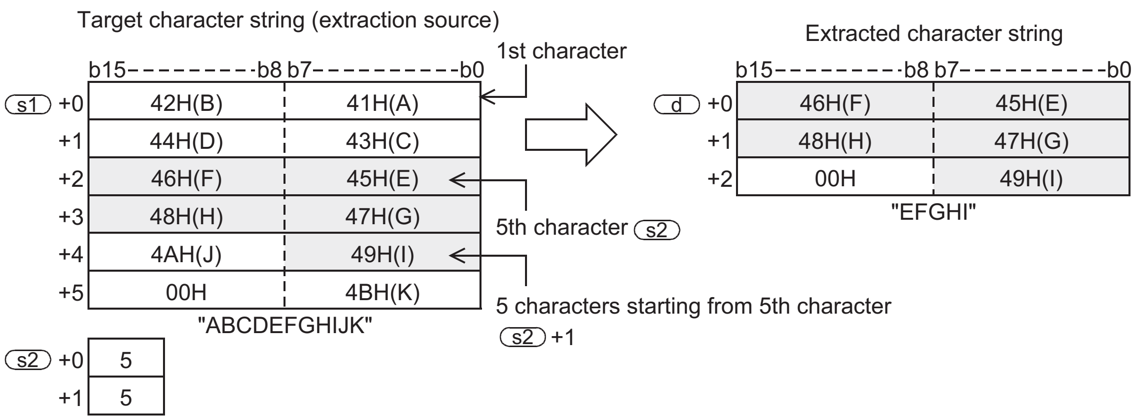
1. 16-bit operation (MIDR/MIDRP)

"s2 + 1" characters are extracted leftward from the position specified by s2 of the character string data stored in the device specified by s1 and later, and stored to the device specified by d and later.
When characters are extracted from a character string, "00H" is automatically added at the end of the extracted characters.

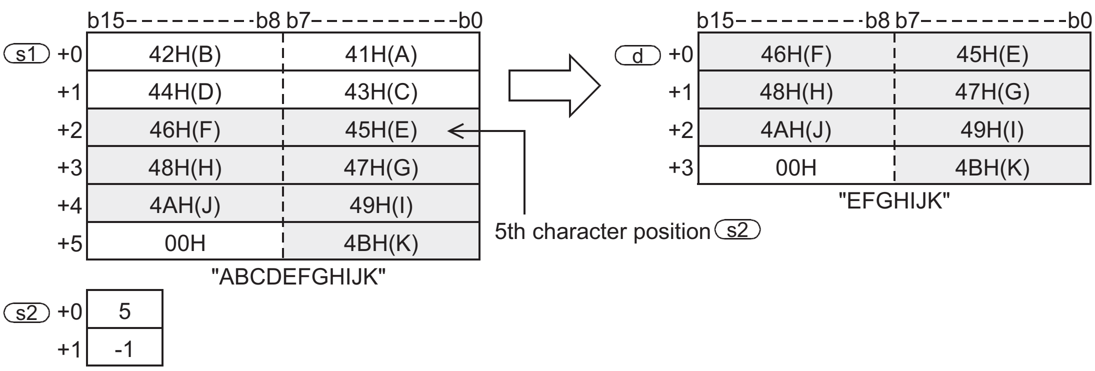
1) When the number of extracted characters of the device specified by "s2 +1" is odd, "00H" is stored in the high-order byte of a device storing the last character.
2) When the number of extracted characters of the device specified by "s2 + 1" is even, "0000H" is stored in the device after the last character.



*1. This defines the head of the device that stores the character string.
*2. This defines the head of the device that specifies the head character position and the number of characters to be extracted.
*3. This defines the head of the device that stores the extracted character string.



a) A character string specified by s1 indicates data stored in devices from the specified device until "00H" is first detected in units of byte.
b) When the number of characters to be extracted specified by "s2 + 1" is "0", the extraction processing is not executed.
c) When the number of characters to be extracted specified by "s2 + 1" is "-1", the entire character string specified by s1 is stored to the device specified by d and later.



Cautions

1) When handling array data or character string data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project.
   Use a label to handle array data or character string data.
   Use a global label to specify a device.
2) When handling character codes other than ASCII codes, note the following contents:
   a) The number of characters is handled in byte units (8 bits). Accordingly, in the case of character codes in which two bytes express one character such as shift JIS codes, the length of one character is detected as "2".
   b) When extracting characters from a character string including character codes in which two bytes express one character such as shift JIS codes, consider the number of characters to be extracted in units of character codes for one character.
      Note that the expected character code is not given if only one byte is extracted out of a 2-byte character code.

Error

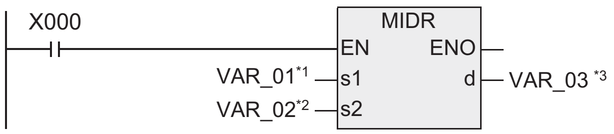
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "00H" is not set within the corresponding device range after a device specified by s1.
   (Error code: K6706)
2) When the value specified by s2 exceeds the number of characters specified by s1.
   (Error code: K6706)
3) When the number of characters specified by "s2 +1" from the position specified by the device specified by d exceeds the device range specified by d. (Error code: K6706)
4) When the number of devices after a device number specified by d is smaller than the number of devices required to store extracted characters as many as the number specified by "s2 +1" (that is, when "00H" cannot be stored after all character strings and the last character) (Error code: K6706)
5) When the contents of the device specified by s2 is a negative value. (Error code: K6706)
6) When the contents of the device specified by "s2 + 1" is "-2" or less. (Error code: K6706)
7) When the contents of the device specified by "s2 + 1" is a number larger than the number of characters specified by s1.
   (Error code: K6706)

Program examples

In the program example shown below, four characters are extracted from the third character from the left end of the character string data stored in D10 and later, and then stored to D0 and later when X000 turns ON.

[Structured ladder]



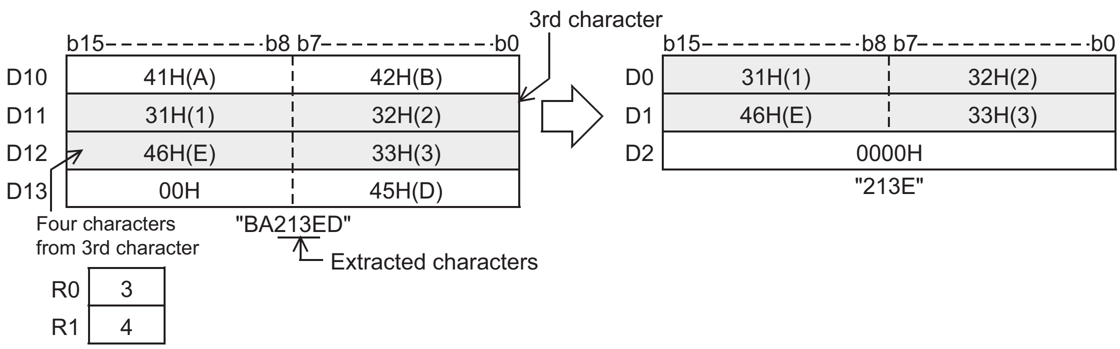
[ST]

```
VAR_03:=MIDR(X000,VAR_01,VAR_02);
```

*1. VAR_01 is a global label and is defined as D10.

*2. VAR_02 is a global label and is defined as R0.

*3. VAR_03 is a global label and is defined as D0.



7.20.8 MIDW

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction replaces the characters in arbitrary positions inside designated character string with a specified character string.

→ **For handling of character strings, refer to "FX Structured Programming Manual (Device & Common)."**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
MIDW	16 bits	Continuous	MIDW block: EN, s1, s2 → ENO, d	MIDW(EN,s1,s2,d);
MIDWP	16 bits	Pulse	MIDWP block: EN, s1, s2 → ENO, d	MIDWP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Head device storing a character string used in overwriting	String
	s2	Head device specifying the head position and number of characters to be overwritten • s2 :Head character position to be overwritten • s2+1 :Number of characters to be overwritten	ARRAY [1..2] OF ANY16
Output variable	ENO	Execution state	Bit
	d	Head device storing a character string overwritte	String

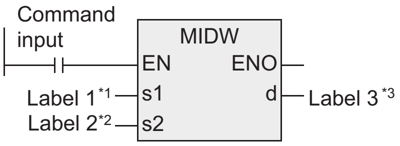
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●			●					
s2								●	●	●	●	●	●	●	●	●			●					

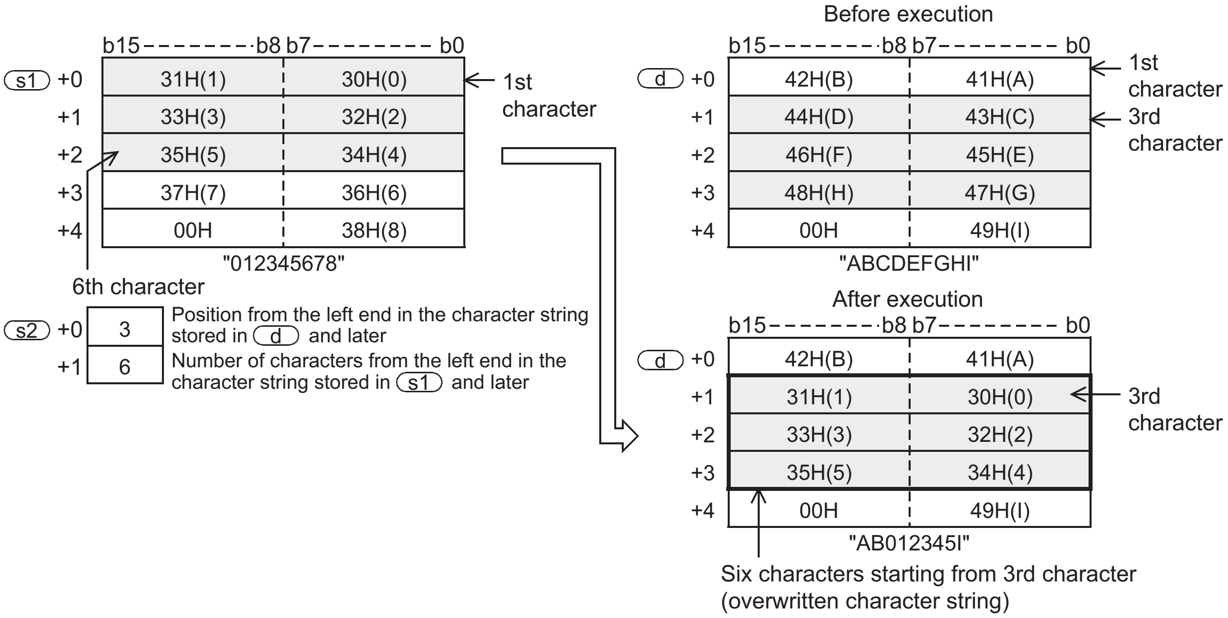
Function and operation explanation

1. 16-bit operation (MIDW/MIDWP)

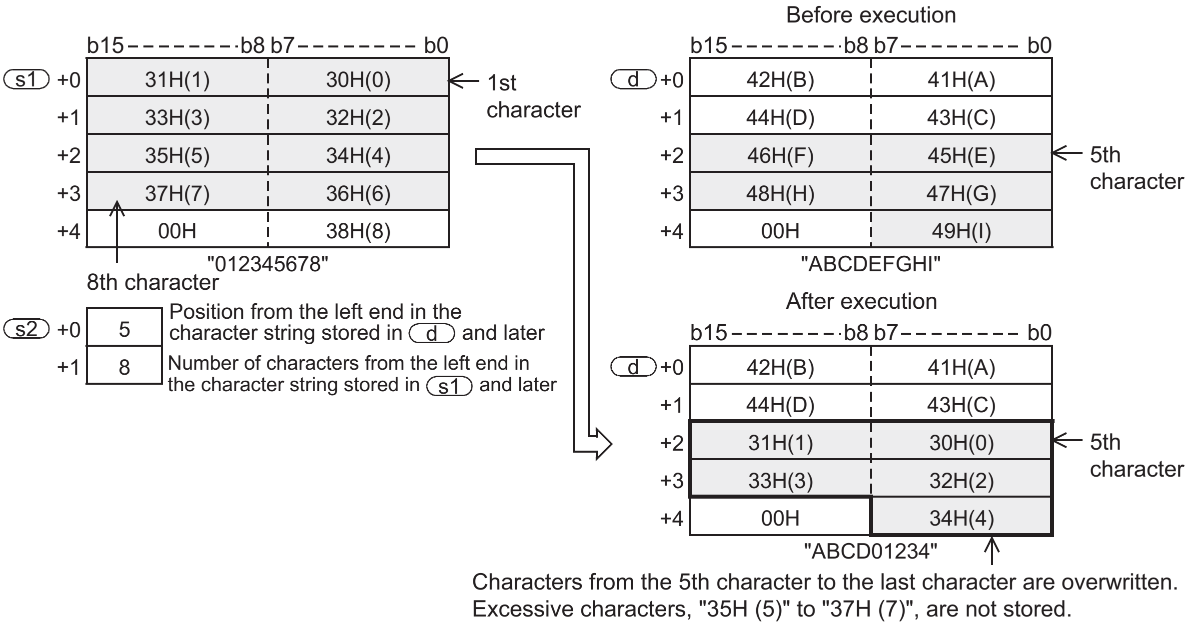
Character data specified by "s2 + 1" are extracted from the left end (that is, the head) of the character string data stored in the device specified by s1 and later, and stored to the position specified by s2 and later of the character string data stored in the device specified by d and later.



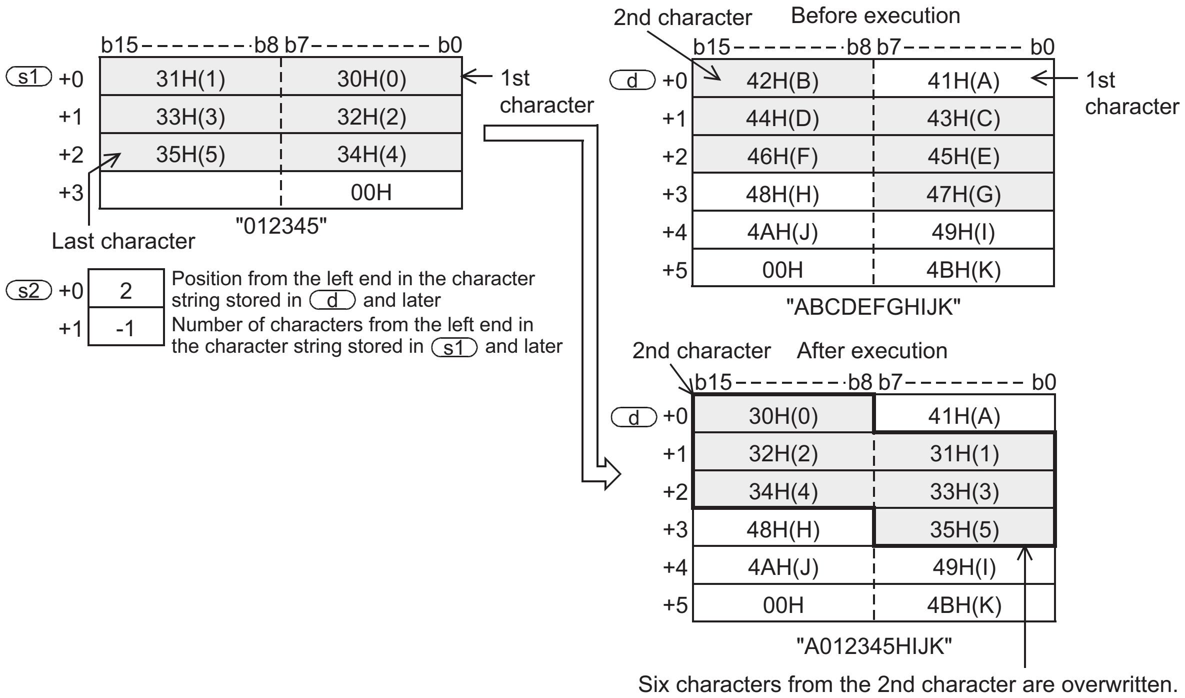
*1. This defines the head of the device that stores the character string used in overwriting.
*2. This defines the head of the device that defines the head position and the number of the characters to be overwritten
*3. This defines the head of the device that stores the character string overwritten.



1) The character string specified by s1 or d indicates data stored in devices from the specified device until "00H" is first detected in byte units.
2) When the number of characters to be overwritten specified by "s2 + 1" is "0", the overwriting processing is not executed.
3) When the number of characters to be overwritten specified by "s2 + 1" exceeds the last character of the character string stored in the device specified by d and later, data is stored up to the last character.



4) When "s2 + 1" (the number of characters to be overwritten) is "-1", the entire character string stored in s1 and later is stored to the device specified by d and later.



Cautions

1) When handling array data or character string data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project.
Use a label to handle array data or character string data.
Use a global label to specify a device.
2) When handling character codes other than ASCII codes, note the following contents:
   a) The number of characters is handled in byte units (8 bits). Accordingly, in the case of character codes in which two bytes express one character such as shift JIS codes, the length of one character is detected as "2".
   b) When extracting characters from a character string including character codes in which two bytes express one character such as shift JIS codes, consider the number of characters to be extracted in units of character codes for one character.
   Note that the expected character code is not given if only one byte is extracted out of a 2-byte character code.

Error

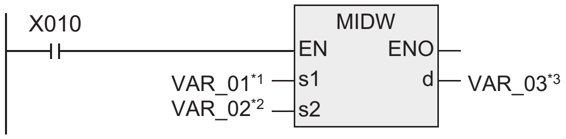
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "00H" is not set within the corresponding device range after a device specified by s1 or d.
(Error code: K6706)
2) When the value of the device specified by s2 exceeds the number of characters specified by d.
(Error code: K6706)
3) When the value of the device specified by "s2 + 1" exceeds the number of characters specified by s1.
(Error code: K6706)
4) When the value of the device specified by s2 is a negative value (Error code: K6706)
5) When the value of the device specified by "s2 + 1" is "-2" or less

Program examples

In the program example shown below, four characters are extracted from the character string data stored in D0 and later, and stored to the third character (from the left end) and later for the character string data stored in D100 and later when X010 turns ON.

[Structured ladder]



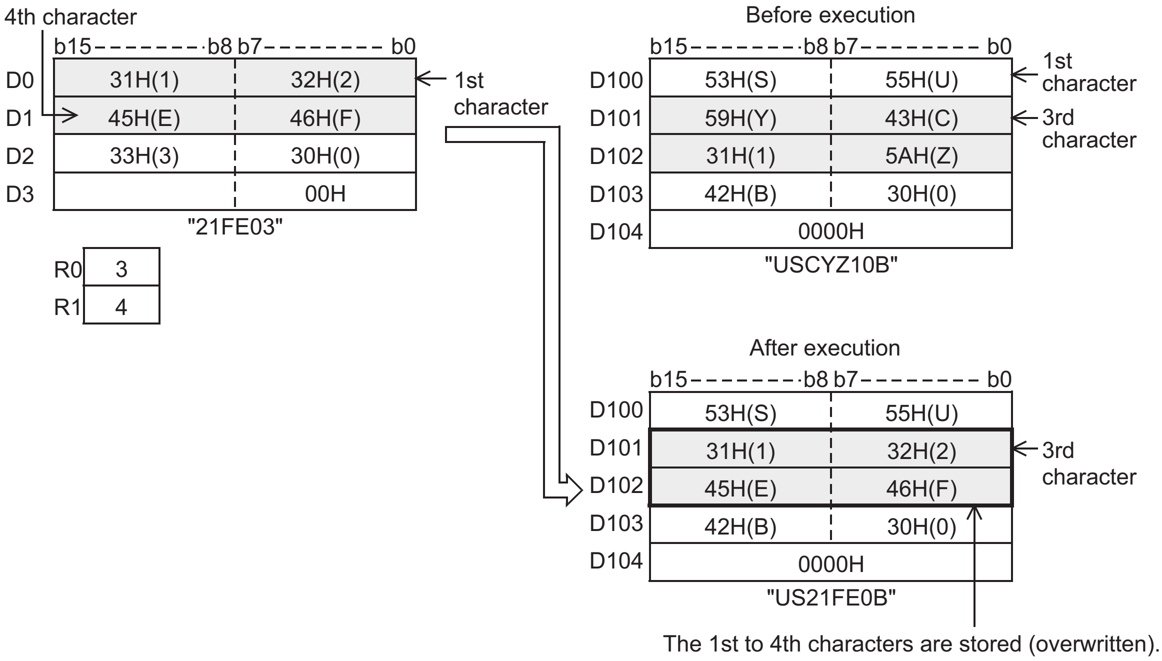
[ST]

```
VAR_03:=MIDW(X010,VAR_01,VAR_02);
```

*1. VAR_01 is a global label and is defined as D0.

*2. VAR_02 is a global label and is defined as R0.

*3. VAR_03 is a global label and is defined as D100.



7.20.9 INSTR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction searches a specified character string within another character string.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
INSTR	16 bits	Continuous	INSTR (EN, s1, s2, n → ENO, d)	INSTR(EN,s1,s2,n,d);
INSTRP	16 bits	Pulse	INSTRP (EN, s1, s2, n → ENO, d)	INSTRP(EN,s1,s2,n,d);

```
      INSTR
 —EN       ENO—
 —s1         d—
 —s2
 —n
```

```
      INSTRP
 —EN       ENO—
 —s1         d—
 —s2
 —n
```

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Head device storing a character string to search for	String
	s2	Head device storing a character string to be searched	String
	n	Search start position	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device storing search result	ANY16

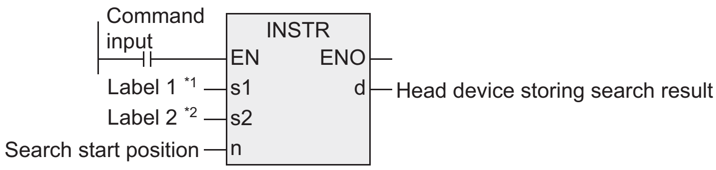
3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit specification				System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1												●	●	●	●				●				●	
s2												●	●	●	●				●					
d												●	●	●	●				●					
n														●	●					●	●			

Function and operation explanation

1. 16-bit operation (INSTR/INSTRP)

1) The character string stored in the device specified by (s1) and later is searched for within the character string of the device specified by (s2) and later. The search begins at the "n"th character from the left end (head character) of the device specified by (s2) and the search result is stored in the device specified by (d).
The search result provides the first matching character (located from the left end (head character)) in the device specified by (s2).



*1. This defines the head of the device that stores the character string to search for.
*2. This defines the head of the device that stores the character string to be searched.

Character string to be searched

	b15 - - - b8	b7 - - - b0	
(s2) +0	42H(B)	41H(A)	
+1	44H(D)	43H(C)	← Search is started from the 3rd character. (n=3)
+2	46H(F)	45H(E)	← 5th character from the head character
+3	48H(H)	47H(G)	
+4	4AH(J)	49H(I)	
+5	00H	4BH(K)	

"ABCDEFGHIJK"

Character string to search for

	b15 - - - b8	b7 - - - b0
(s1) +0	46H(F)	45H(E)
+1	48H(H)	47H(G)
+2		00H

"EFGH"

⇩

(d) 5 — Stores the position where the first detected character is located from the head character in the character string data stored in (s2).

2) When the searched character string is not detected, "0" is stored in the device specified by (d).
3) When the search start position "n" is a negative number or "0", search processing is not executed.
4) A character string can be directly specified in the character string to search for specified by (s1).

Character string to be searched

	b15 - - - b8	b7 - - - b0	
(s2) +0	32H(2)	31H(1)	
+1	34H(4)	33H(3)	← Search is started from the 3rd character. (n=3)
+2	42H(B)	41H(A)	← 5th character from he head character
+3	36H(6)	35H(5)	
+4	42H(B)	41H(A)	
+5		00H	

"1234AB56AB"

Character string to search for

"AB"

⇩

D100 5

Cautions

1) When handling character string data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle character string data.
Use a global label to specify a device.
2) The FX3UC PLC of V 2.20 or later supports this instruction.

Error

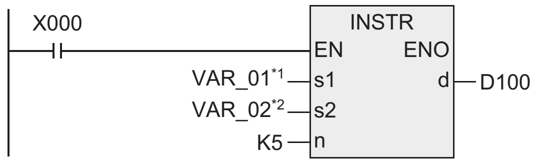
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the search start position "n" exceeds the number of characters stored in (s2)
   (Error code: K6706)
2) When "00H (NULL)" is not located within the corresponding device range starting from the device specified by (s1).
   (Error code: K6706)
3) When "00H (NULL)" is not located within the corresponding device range starting from the device specified by (s2).
   (Error code: K6706)

Program examples

1) In the program example below, the character string "C123" (D0 and later) is searched from the fifth character from the left end (head character) of the character string "CI2312CIM" (R0 and later) when X000 is set to ON. The search result is stored in D100.

[Structured ladder]

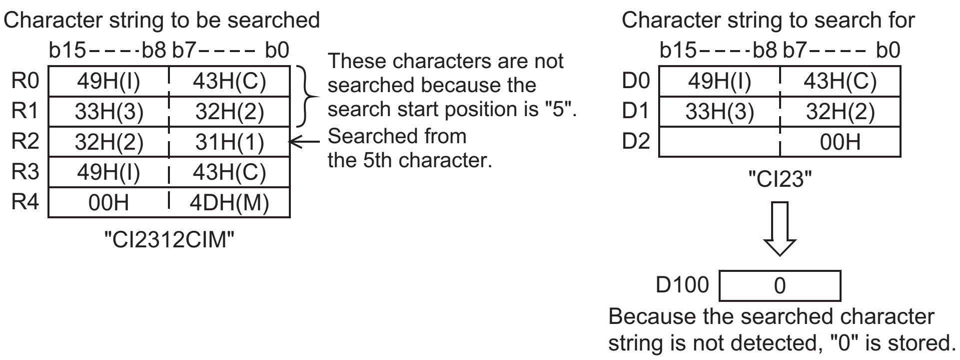


[ST]

```
D100:=INSTR(X000,VAR_01,VAR_02,K5);
```

*1. VAR_01 is a global label and is defined as D0.

*2. VAR_03 is a global label and is defined as D100.



7.20.10 $MOV

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction transfers character string data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
$MOV	16 bits	Continuous	$MOV EN ENO s d	$MOV(EN,s,d);
$MOVP	16 bits	Pulse	$MOVP EN ENO s d	$MOVP(EN,s,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Directly specified character string (up to 32 characters) or head device storing character string which is handled as the transfer source	String
Output variable	ENO	Execution state	Bit
	d	Head device storing transferred character string	String

3. Applicable devices

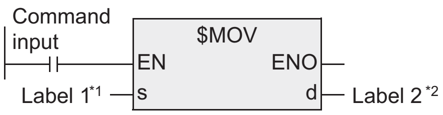
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s								●	●	●	●	●	●	●	●	●			●				●	
d									●	●	●	●	●	●	●	●			●					

1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

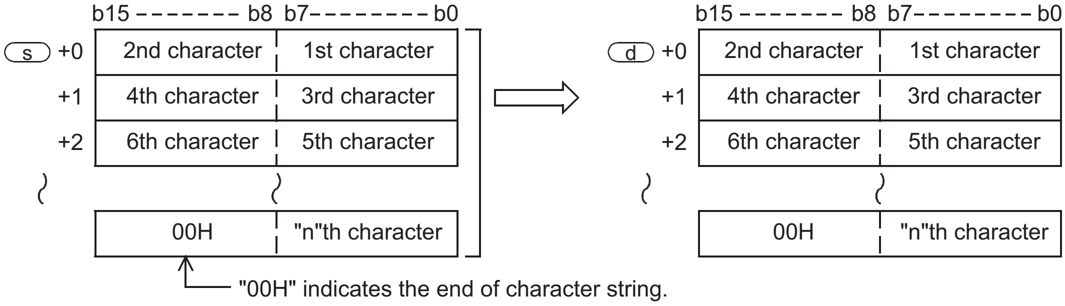
1. 16-bit operation ($MOV/$MOVP)

The character string data stored in the device specified by (s) and later is transferred to the device specified by (d) and later.
From the device number specified by (s) to a device after that which stores "00H" in its high-order or low-order byte are transferred at one time.

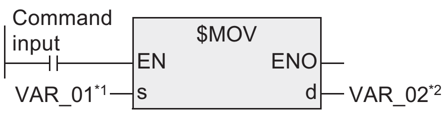


*1. This defines the directly specified character string or head device that stores the character string which is handled as transfer source.

*2. This defines the head of the device that stores the transferred character string.

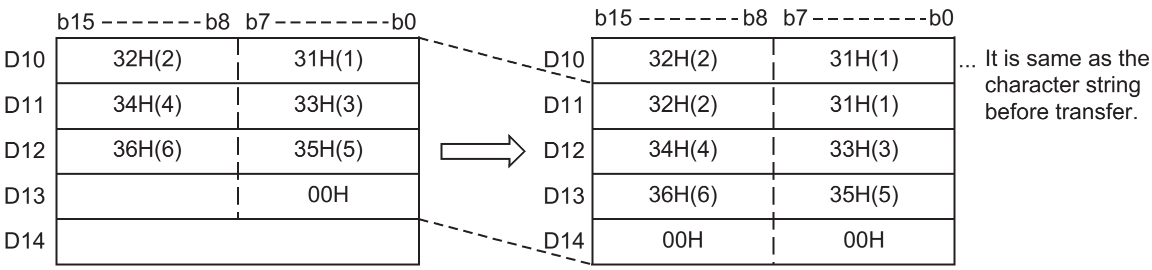


Even if the device range storing the character string data to be transferred overlaps the device range storing the transferred character string data, transfer is executed.
For example, when a character string stored in D10 to D13 is transferred to D11 to D14, the transfer is executed as shown below.



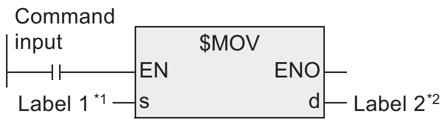
*1. VAR_01 is a global label and is defined as D10.

*2. VAR_02 is a global label and is defined as D11.



Cautions

1) When handling character string data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle character string data.
   Use a global label to specify a device.
2) When "00H" is stored in the low-order byte of the device specified by "(s2) + n", "00H" is stored to both the high-order byte and low-order byte of the device specified by "(d) + n".



*1. This defines the directly specified character string or head device that stores the character string which is handled as transfer source.

*2. This defines the head of the device that stores the transferred character string.

(s)	b15 ------- b8	b7-------- b0
+0	42H(B)	41H(A)
+1	44H(D)	43H(C)
+2	45H(E)	00H

The high-order byte is not transferred.

⇨

(d)	b15 ------- b8	b7-------- b0
+0	42H(B)	41H(A)
+1	44H(D)	43H(C)
+2	00H	00H

"00H" is automatically stored in the high-order byte.

Error

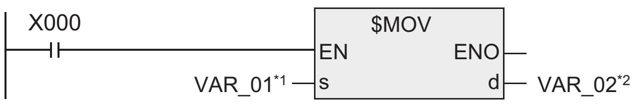
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When "00H" does not exist in the range specified from device specified by (s) to the last device (Error code: K6706)
2) When the specified character string cannot be stored in devices from the device specified by (d) to the last device. (Error code: K6706)

Program examples

In the program example shown below, character string data stored in D10 to D12 is transferred to D20 through D22 when X000 is set to ON.

[Structured ladder]



[ST]

```
VAR_02:=$MOV(X000,VAR_01);
```

*1. VAR_01 is a global label and is defined as D10.

*2. VAR_02 is a global label and is defined as D20.

	b15 -------b8	b7 --------b0
D10	4DH(M)	2AH(*)
D11	45H(E)	45H(E)
D12		00H

⇨

	b15 -------b8	b7 --------b0
D20	4DH(M)	2AH(*)
D21	45H(E)	45H(E)
D22	00H	00H

7.21 Data Operation 3

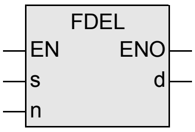
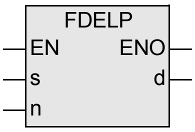
7.21.1 FDEL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction deletes an arbitrary piece of data from a data table.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladde	ST
FDEL	16 bits	Continuous		FDEL(EN,s,n,d);
FDELP	16 bits	Pulse		FDELP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Device storing deleted data	ANY16
	n	Position of deleted data in table	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device in data table	ANY16

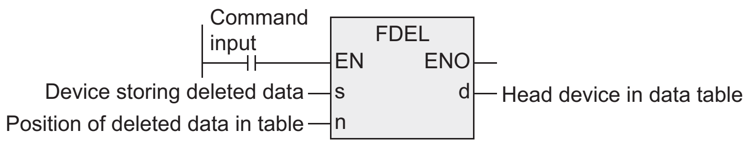
3. Applicable devices

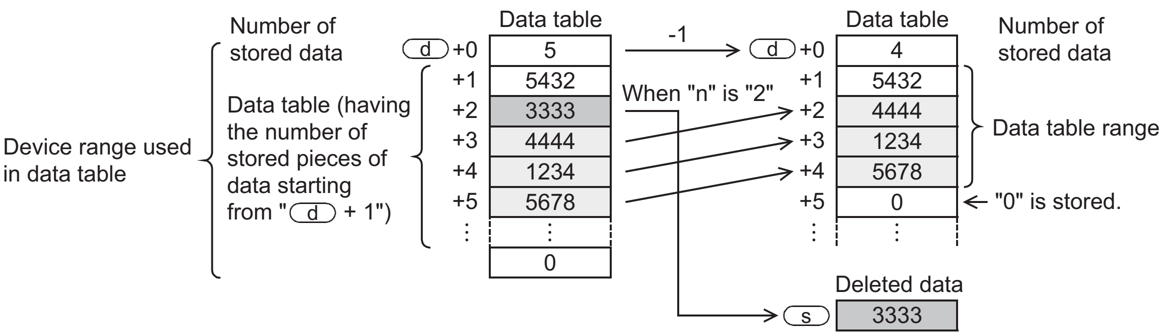
Operand type	Bit Devices							Word Devices													Others					
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s												●	●	●	●				●					
d												●	●	●	●				●					
n														●	●					●	●			

Function and operation explanation

1. 16-bit operation (FDEL/FDELP)

"n"th data is deleted from a data table (stored in the device specified by (d) and later), and the deleted data is stored in the device specified by (s). "n + 1"th data and later in the data table are shifted forward one by one, and the number of stored data is subtracted by "1".





Cautions

1) The device range used in a data table should be controlled by the user.
   The data table has the number of pieces of data, which is stored in (d), starting from the next device ((d) + 1) after the device storing the number of pieces of data (d).
   → **Refer to the program example below.**
2) The FX3UC PLC of V 2.20 or later supports this instruction.

Related instruction

Instruction	Description
FINS	Inserts data into an arbitrary position in a data table

Error

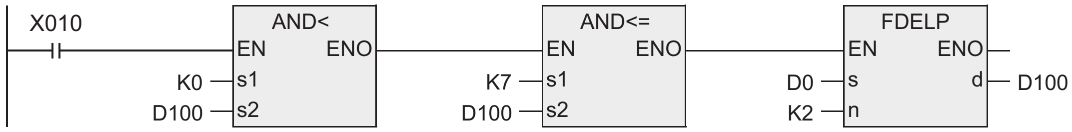
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the table position "n" from the device specified by (d) exceeds the "amount of data stored plus one"
   (Error code: K6706)
2) When the value "n" exceeds the range of the number of pieces of data specified by (d).
   (Error code: K6706)
3) When the FDEL instruction is executed under the condition "n ≤ 0" (Error code: K6706)
4) When the amount of data stored specified by (d) is "0". (Error code: K6706)
5) When the data table range exceeds the corresponding device range (Error code: K6706)

Program examples

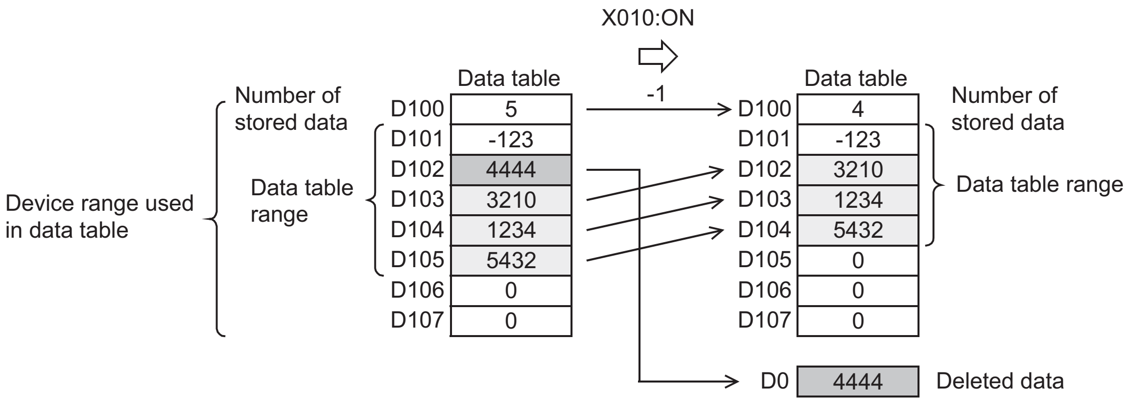
In the program example shown below, the second data is deleted from the data table stored in D100 to D105, and the deleted data is stored in D0 when X010 is set to ON.
When the amount of data stored is "0", however, the FDEL instruction is not executed.
(The device range used in the data table is D100 to D107.)

[Structured ladder]



[ST]

```
IF((IF(AND<(X010,K0,D100)THEN AND<=(TRUE,K7,D100)))
THEN FDELP(TRUE,D0,K2,D100);
```



7.21.2 FINS

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction inserts data into an arbitrary position in a data table.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
FINS	16 bits	Continuous	FINS block: EN, s, n inputs; ENO, d outputs	FINS(EN,s,n,d);
FINSP	16 bits	Pulse	FINSP block: EN, s, n inputs; ENO, d outputs	FINSP(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Device storing inserted data	ANY16
	(n)	Data insertion position in table	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Head device in data table	ANY16

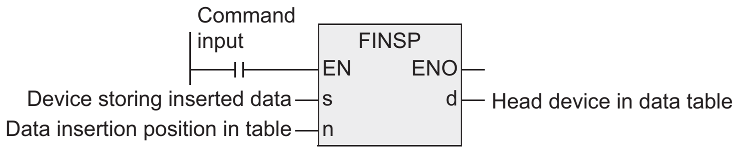
3. Applicable devices

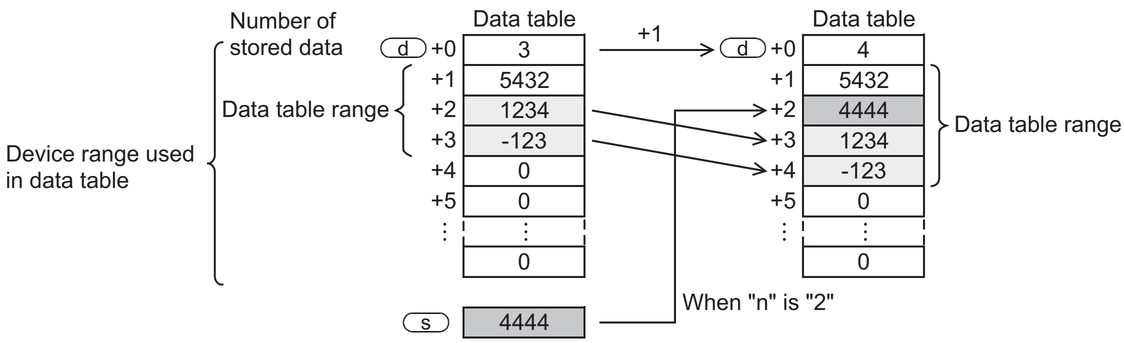
Operand type	Bit Devices: System user							Word Devices: Digit specification				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)												●	●	●	●				●	●	●			
(d)												●	●	●	●				●					
(n)														●	●					●	●			

Function and operation explanation

1. 16-bit operation (FINS/FINSP)

1) 16-bit data of the device specified by (s) is inserted in "n"th position in a data table (stored in the device specified by (d) and later).
"n"th data and later in the data table are shifted backward one by one, and the number of stored data is added by "1".





Cautions

1) The device range used in a data table should be controlled by the user.
The data table has the number of pieces of data, which is stored in (d), starting from the device after the device that indicates the number of stored data (d).
→ **Refer to the program example below.**
2) The FX3UC PLC of V 2.20 or later supports this instruction.

Related instruction

Instruction	Description
FDEL	Deletes an arbitrary data from a data table.

Error

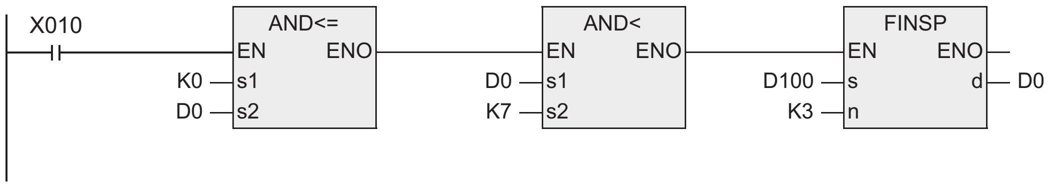
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the table position "n" from the device specified by (d) exceeds the "amount of data stored plus one"
(Error code: K6706)
2) When the value "n" exceeds the device range of the data table specified by (d)
(Error code: K6706)
3) When the FDEL instruction is executed under the condition "n ≤ 0" (Error code: K6706)
4) When the data table range exceeds the corresponding device range (Error code: K6706)

Program examples

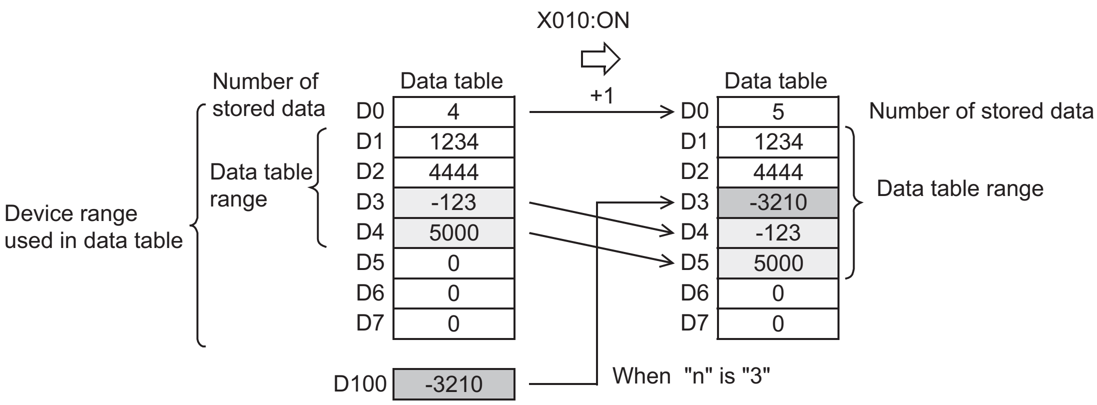
In the program example shown below, data stored in D100 is inserted into the third position of the data table stored in D0 to D4 when X010 is set to ON.
When the amount of data stored exceeds "7", however, the FINS instruction is not executed.
(The device range used in the data table is D0 to D7.)

[Structured ladder]



[ST]

```
IF(( IF(AND<=(X010,K0,D0)THEN AND<(TRUE,D0,K7)))THEN FINSP(TRUE,D100,K3,D0);
```



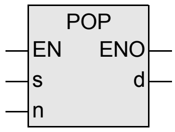
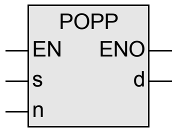
7.21.3 POP

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction reads the last data written by the shift write (SFWR) instruction for the first-in first-out and first-in last-out control

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
POP	16 bits	Continuous		POP(EN,s,n,d);
POPP	16 bits	Pulse		POPP(EN,s,n,d);

2. Set data

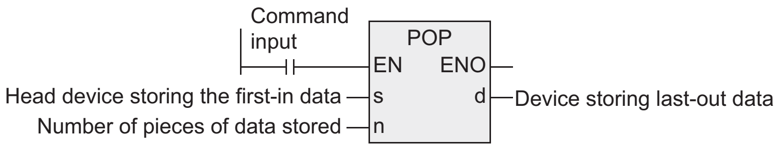
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device storing first-in data (including pointer data)	ANY16
	n	Number of pieces of data stored (Add "1" because pointer data is also included.)	ANY16
Output variable	ENO	Execution state	Bit
	d	Device storing last-out data	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s									●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●	●	●	●					
n																				●	●			

Function and operation explanation

1. 16-bit operation (POP/POPP)

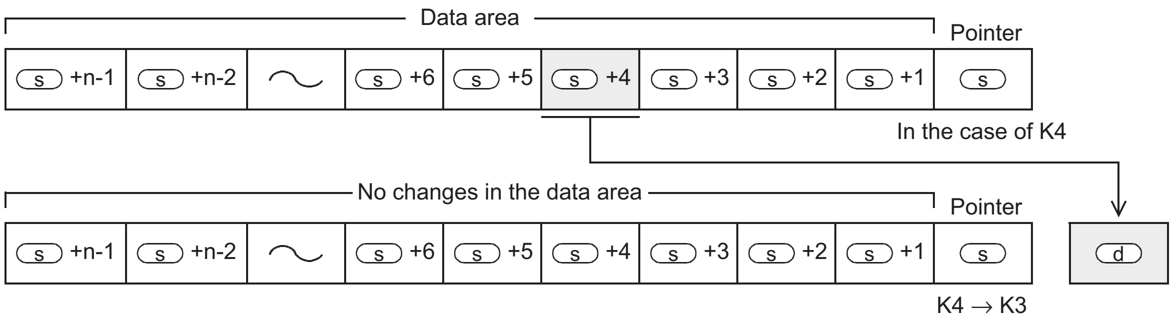


Data for FILO control

	Description
(s)	Pointer data (amount of data stored)
(s)+1	Data area (First-in data written by shift write (SFWR) instruction)
(s)+2	
(s)+3	
⋮	
(s)+n-3	
(s)+n-2	
(s)+n-1	

1) Every time the instruction is executed for the word devices specified by "(s) to (s) + n-1", a device "(s) + Pointer data" is read to the device specified by (d). (The last data entry written by the shift write (SFWR) instruction for first-in first-out control is read to the device specified by (d).) Specify "n" in the range from "2" to "512".

2) Subtract "1" from the value of the pointer data of the device specified by (s).



Related device

→ for the zero flag use method, refer to Section 1.3.4.

Device	Name	Description
M8020	Zero flag	Turns ON when the instruction is executed while the pointer is "0".

Related instruction

Instruction	Description
SFWR	Shift write [for FIFO/FILO control]
SFRD	Shift read [for FIFO control]

Cautions

1) When this instruction is programmed in the continuous operation type, the instruction is executed in every operation cycle. As a result, an expected operation may not be achieved.
Usually, program this instruction in the "pulse operation type", or let this instruction be executed by a "pulsed command contact.
2) When the current value of the pointer specified by (s) is "0", the zero flag M8020 turns ON and the instruction is not executed.
Check in advance using a comparison instruction whether the current value of the device specified by (s) satisfies "1 $\leq$ current value $\leq$ (n-1)", and then execute this instruction.
3) When the current value of the pointer specified by (s) is "1", "0" is written to the pointer and the zero flag M8020 turns ON.

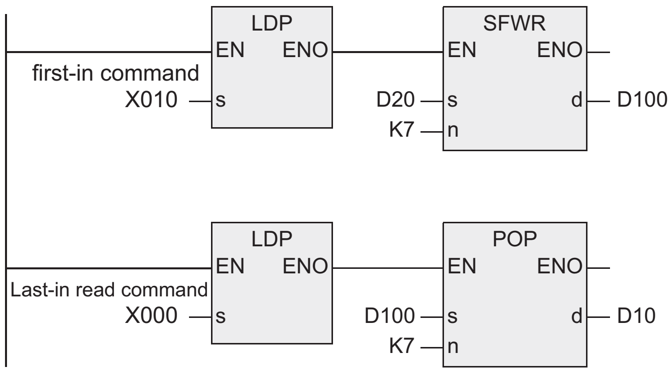
Error

1) An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.
   a) When the pointer is larger than "n-1". (Error code: K6706)
   b) When the pointer is smaller than "0". (Error code: K6706)

Program examples

In the program example shown below, among value stored in D20 input first to D101 to D106, the last value input is stored to D10, and "1" is subtracted from the number of stored data (pointer D100) every time X000 turns ON.

[Structured ladder]

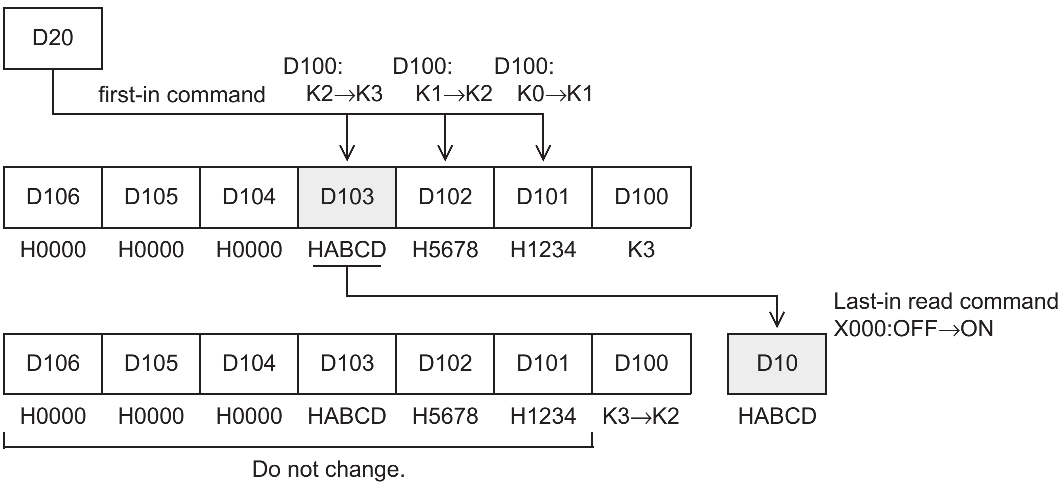


[ST]

```
IF(LDP(TRUE,X010))THEN SFWR(TRUE,D20,K7,D100);
IF(LDP(TRUE,X000))THEN POP(TRUE,D100,K7,D10);
```

When the first-in data is as shown in the table below

Pointer	D100	K3
Data	D101	H1234
	D102	H5678
	D103	HABCD
	D104	H0000
	D105	H0000
	D106	H0000



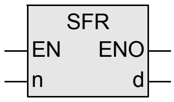
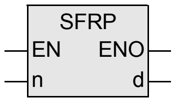
7.21.4 SFR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction shifts 16 bits stored in a word device rightward by "n" bits.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
SFR	16 bits	Continuous		SFR(EN,n,d);
SFRP	16 bits	Pulse		SFRP(EN,n,d);

2. Set data

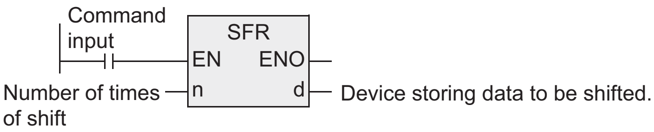
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(n)	Number of times of shift (0 ≤ n ≤ 15)	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Device storing data to be shifted.	ANY16

3. Applicable devices

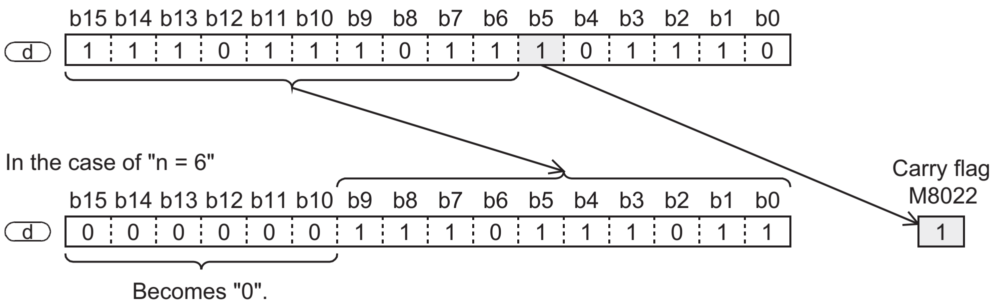
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(d)									●	●	●	●	●	●	●	●	●	●	●					
(n)								●	●	●	●	●	●	●	●	●	●	●		●	●			

Function and operation explanation

1. 16-bit operation (SFR/SFRP)

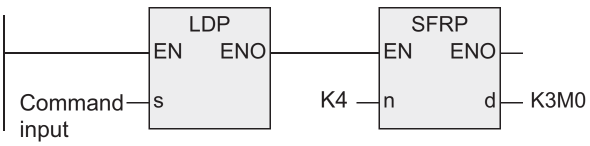


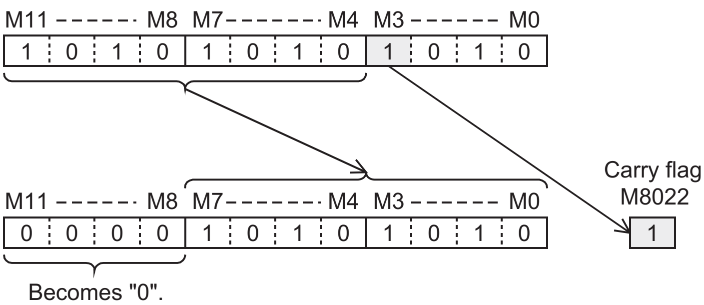
1) 16 bits stored in a word device specified by (d) are shifted rightward by "n" bits.
   Specify a value in the range from "0" to "15" as "n".
   If "16" or larger value is specified as "n", 16 bits are shifted rightward by the remainder of "(n) divided by (16)".
   For example, when "n" is set to "18", 16 bits are shifted rightward by 2 bits as "2" remains when "18" is divided by "16".
2) The ON (1) or OFF (0) status of the "n"th bit (bit "n-1") in the word device specified by (d) is transferred to the carry flag M8022.
3) "0" is set to "n" bits from the most significant bit.



When a bit device is specified by digit specification

(4 × K□) bits are shifted according to the data bit specification.)





Related device

→ For the carry flag use method, refer to Section 1.3.4.

Device	Name	Description
M8022	Carry flag	Shifts the ON/OFF status of bit "n-1"

Error

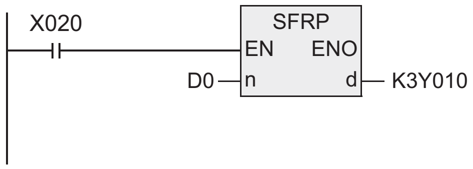
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When a negative value is set to "n" (Error code: 6706)

Program examples

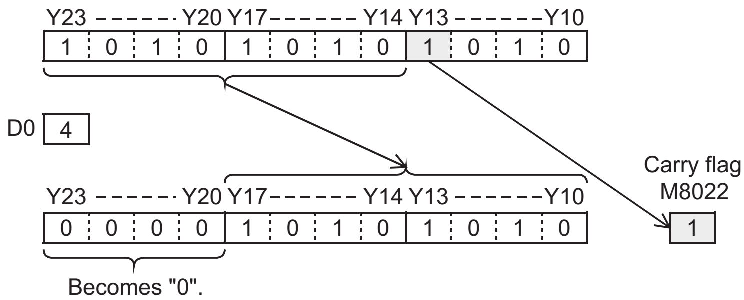
In the program example shown below, the contents of Y010 to Y023 are shifted rightward by the number of bits specified by D0 when X020 turns ON.

[Structured ladder]



[ST]

```
SFR(X020,D0,Y3Y010);
```



7.21.5 SFL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction shifts 16 bits stored in a word device leftward by "n" bits.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
SFL	16 bits	Continuous	SFL EN ENO n d	SFL(EN,n,d);
SFLP	16 bits	Pulse	SFLP EN ENO n d	SFLP(EN,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	n	Number of times of shift	ANY16
Output variable	EN	Execution state	Bit
	d	Device storing data to be shifted.	ANY16

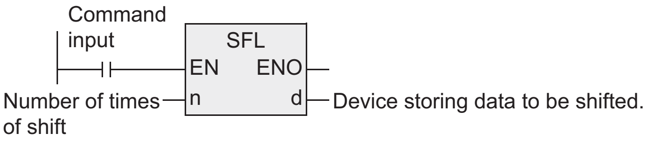
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
d									●	●	●	●	●	●	●	●	●	●	●					
n								●	●	●	●	●	●	●	●	●	●	●		●	●			

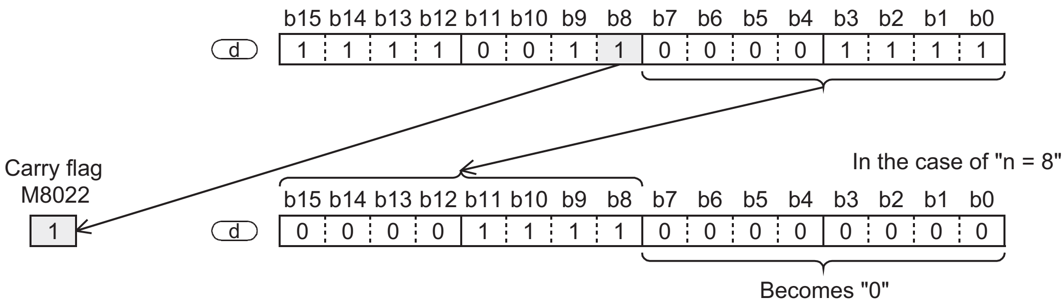
1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

1. 16-bit operation (SFL/SFLP)

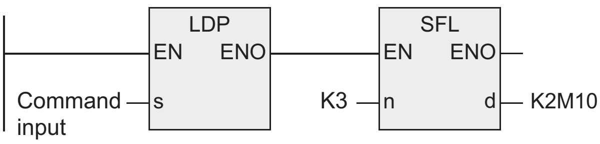


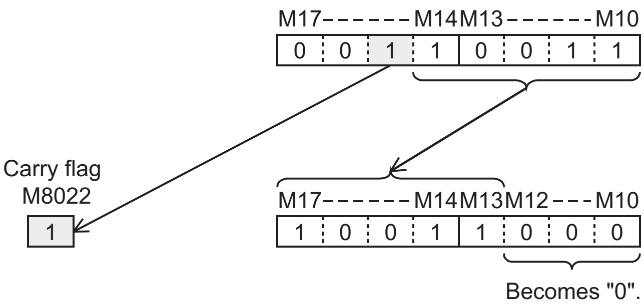
1) 16 bits stored in a word device specified by (d) are shifted leftward by "n" bits.
   Specify a value in the range from "0" to "15" as "n".
   If "16" or larger value is specified as "n", 16 bits are shifted leftward by the remainder of "(n) divided by (16)".
   For example, when "n" is set to "18", 16 bits are shifted leftward by 2 bits as "2" remains when "18" is divided by "16".
2) The ON (1) or OFF (0) status of the "n + 1"th bit (bit "n") in the word device specified by (d) is transferred to the carry flag M8022.
3) "0" is set to "n" bits from the least significant bit.



When a bit device is specified by digit specification

(4 × K□) bits are shifted according to the data bit specification.)





Related device

→ **For the carry flag use method, refer to Section 1.3.4.**

Device	Name	Description
M8022	Carry flag	Shifts the ON/OFF status of bit "n"

Error

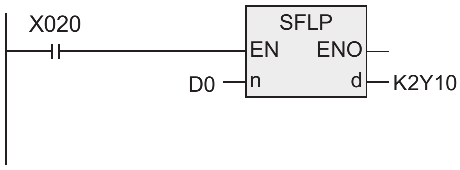
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When a negative value is set to "n" (Error code: 6706)

Program examples

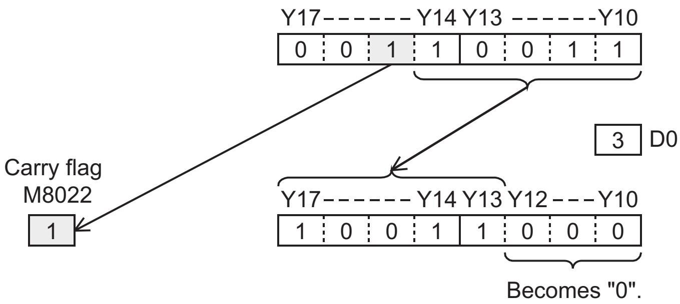
In the program example shown below, the contents of Y010 to Y017 are shifted leftward by the number of bits specified by D0 when X020 turns ON.

[Structured ladder]



[ST]

```
SFLP(X020,D0,K2Y10);
```



7.22 Data Comparison

7.22.1 LD=, LD>, LD<, LD<>, LD<=, LD>=

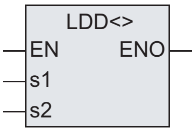
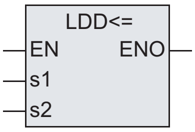
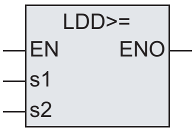
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

These instructions compare numeric values, and set a contact to ON when the condition agrees so that an operation started.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
LD=	16 bits	Continuous	LD= EN ENO s1 s2	LD=(EN,s1,s2);
LD>	16 bits	Continuous	LD> EN ENO s1 s2	LD>(EN,s1,s2);
LD<	16 bits	Continuous	LD< EN ENO s1 s2	LD<(EN,s1,s2);
LD<>	16 bits	Continuous	LD<> EN ENO s1 s2	LD<>(EN,s1,s2);
LD<=	16 bits	Continuous	LD<= EN ENO s1 s2	LD<=(EN,s1,s2);
LD>=	16 bits	Continuous	LD>= EN ENO s1 s2	LD>=(EN,s1,s2);
LDD=	32 bits	Continuous	LDD= EN ENO s1 s2	LDD=(EN,s1,s2);
LDD>	32 bits	Continuous	LDD> EN ENO s1 s2	LDD>(EN,s1,s2);
LDD<	32 bits	Continuous	LDD< EN ENO s1 s2	LDD<(EN,s1,s2);

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
LDD<>	32 bits	Continuous		LDD<>(EN,s1,s2);
LDD<=	32 bits	Continuous		LDD<=(EN,s1,s2);
LDD>=	32 bits	Continuous		LDD>=(EN,s1,s2);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s1)	Device storing comparison data	ANY16	ANY32
	(s2)	Device storing comparison data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(s2)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			

▲: Refer to "Cautions"

Function and operation explanation

These data comparison instructions are connected to bus lines.

The contents of the devices specified by (s1) are compared with the contents of the device specified by (s2) in the binary format, and a contact becomes conductive (ON) or non-conductive (OFF) depending on the comparison result.

16-bit instruction	32-bit instruction	ON condition	OFF condition
LD=	LDD=	(s1) = (s2)	(s1) ≠ (s2)
LD>	LDD>	(s1) > (s2)	(s1) ≤ (s2)
LD<	LDD<	(s1) < (s2)	(s1) ≥ (s2)
LD<>	LDD<>	(s1) ≠ (s2)	(s1) = (s2)
LD<=	LDD<=	(s1) ≤ (s2)	(s1) > (s2)
LD>=	LDD>=	(s1) ≥ (s2)	(s1) < (s2)

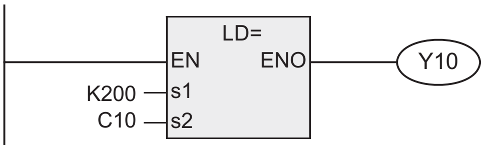
Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data. A 32-bit counter can be specified directly as it is a 32-bit long device. Use a global label to specify a device.
2) Negative values
   When the most significant bit is "1" in the data stored in the device specified by (s1) or (s2), it is regarded as a negative value in comparison.
   a) In the 16-bit operation: bit 15
   b) In the 32-bit operation: bit 31
3) When using 32-bit counters (including 32-bit high speed counters)
   Be sure to execute the 32-bit operation (such as LDD=, LDD> and LDD<) when comparing 32-bit counters.
   If a 32-bit counter is specified in the 16-bit operation (such as LD=, LD> and LD<), a program error or operation error will occur.
4) Some restrictions to applicable devices
   ▲1: Applicable only to the FX3U, FX3UC and FX3G PLCs.
   ▲2: Applicable only to the FX3U and FX3UC PLCs.

Program examples

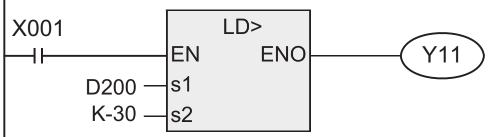
[Structured ladder] [ST]

When the current value of the counter C10 is "200", Y010 is driven.



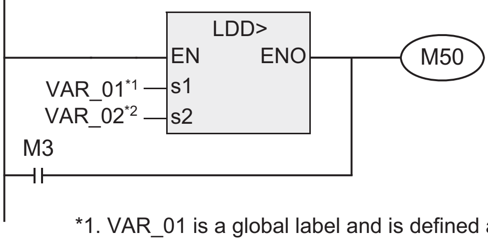
Y10:=LD=(TRUE,K200,C10);

When the contents of D200 are "-29" or more and X001 is ON, Y011 is set.



Y11:=LD>(X001,D200,K-30);

When the contents of the counter C200 are less than "K678,493" or when M3 turns ON, M50 is driven.



M50:=LDD>(TRUE,VAR_01,VAR_02) OR M3;

*1. VAR_01 is a global label and is defined as K678493.

*2. VAR_02 is a global label and is defined as C200.

7.22.2 AND=, AND>, AND<, AND<>, AND<=, AND>=

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

These instructions compare numeric values, and set a contact to ON when the condition agrees.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
AND=	16 bits	Continuous	AND= EN ENO s1 s2	AND=(EN,s1,s2);
AND>	16 bits	Continuous	AND> EN ENO s1 s2	AND>(EN,s1,s2);
AND<	16 bits	Continuous	AND< EN ENO s1 s2	AND<(EN,s1,s2);
AND<>	16 bits	Continuous	AND<> EN ENO s1 s2	AND<>(EN,s1,s2);
AND<=	16 bits	Continuous	AND<= EN ENO s1 s2	AND<=(EN,s1,s2);
AND>=	16 bits	Continuous	AND>= EN ENO s1 s2	AND>=(EN,s1,s2);
ANDD=	32 bits	Continuous	ANDD= EN ENO s1 s2	ANDD=(EN,s1,s2);
ANDD>	32 bits	Continuous	ANDD> EN ENO s1 s2	ANDD>(EN,s1,s2);
ANDD<	32 bits	Continuous	ANDD< EN ENO s1 s2	ANDD<(EN,s1,s2);

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ANDD<>	32 bits	Continuous	ANDD<> EN ENO s1 s2	ANDD<>(EN,s1,s2);
ANDD<=	32 bits	Continuous	ANDD<= EN ENO s1 s2	ANDD<=(EN,s1,s2);
ANDD>=	32 bits	Continuous	ANDD>= EN ENO s1 s2	ANDD>=(EN,s1,s2);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Device storing comparison data	ANY16	ANY32
	s2	Device storing comparison data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	▲1	▲2	●	●	●					
s2								●	●	●	●	●	●	●	▲1	▲2	●	●	●					

▲: Refer to "Cautions"

Function and operation explanation

These data comparison instructions are connected to other contacts in series.

The contents of the device specified by (s1) are compared with the contents of the device specified by (s2) in binary format, and a contact becomes conductive (ON) or non-conductive (OFF) depending on the comparison result.

16-bit instruction	32-bit instruction	ON condition	OFF condition
AND=	ANDD=	(s1) = (s2)	(s1) ≠ (s2)
AND>	ANDD>	(s1) > (s2)	(s1) ≤ (s2)
AND<	ANDD<	(s1) < (s2)	(s1) ≥ (s2)
AND<>	ANDD<>	(s1) ≠ (s2)	(s1) = (s2)
AND<=	ANDD<=	(s1) ≤ (s2)	(s1) > (s2)
AND>=	ANDD>=	(s1) ≥ (s2)	(s1) < (s2)

Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data. A 32-bit counter can be specified directly as it is a 32-bit long device. Use a global label to specify a device.

2) Negative values
   When the most significant bit is "1" in the data stored in the device specified by (s1) or (s2), it is regarded as a negative value in comparison.
   a) In the 16-bit operation: bit 15
   b) In the 32-bit operation: bit 31

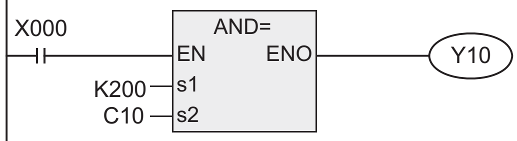
3) When using 32-bit counters (including 32-bit high speed counters)
   Be sure to execute the 32-bit operation (such as ANDD=, ANDD> and ANDD<) when comparing 32-bit counters.
   If a 32-bit counter is specified in the 16-bit operation (such as AND=, AND> and AND<), a program error or operation error will occur.

4) Some restrictions to applicable devices
   ▲1: Applicable only to the FX3U, FX3UC and FX3G PLCs.
   ▲2: Applicable only to the FX3U and FX3UC PLCs.

Program examples

[Structured ladder]

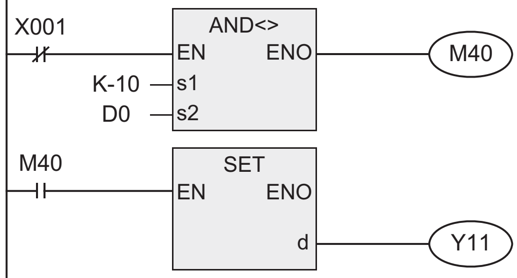
When X000 is ON and the current value of the counter C10 is "200" Y010 is driven.



[ST]

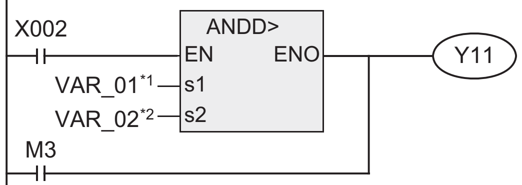
Y10:=AND=(X000,K200,C10);

When X001 is OFF and the contents of the data register D0 are not "-10", Y011 is set.



AND<>(NOT X001,K-10,D0,M40);
Y11:=SET(M40);

When X002 is ON, and the contents of the data registers D11 and D10 are less than "K678,493", or when M3 turns ON, M50 is driven.



Y11:=ANDD>(X002,VAR_01,VAR_02) OR M3;

*1. VAR_01 is a global label and is defined as K678493.
*2. VAR_02 is a global label and is defined as D10.

7.22.3 OR=, OR>, OR<, OR<>, OR<=, OR>=

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	○	○	○	×	×	×

Outline

These instructions compare numeric values, and set a contact to ON when the condition agrees.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
OR=	16 bits	Continuous	OR= EN ENO s1 s2	OR=(EN,s1,s2);
OR>	16 bits	Continuous	OR> EN ENO s1 s2	OR>(EN,s1,s2);
OR<	16 bits	Continuous	OR< EN ENO s1 s2	OR<(EN,s1,s2);
OR<>	16 bits	Continuous	OR<> EN ENO s1 s2	OR<>(EN,s1,s2);
OR<=	16 bits	Continuous	OR<= EN ENO s1 s2	OR<=(EN,s1,s2);
OR>=	16 bits	Continuous	OR>= EN ENO s1 s2	OR>=(EN,s1,s2);
ORD=	32 bits	Continuous	ORD= EN ENO s1 s2	ORD=(EN,s1,s2);

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
ORD>	32 bits	Continuous	ORD> (EN ENO, s1, s2)	ORD>(EN,s1,s2);
ORD<	32 bits	Continuous	ORD< (EN ENO, s1, s2)	ORD<(EN,s1,s2);
OR<>	32 bits	Continuous	OR<> (EN ENO, s1, s2)	OR<>(EN,s1,s2);
ORD<=	32 bits	Continuous	ORD<= (EN ENO, s1, s2)	ORD<=(EN,s1,s2);
ORD>=	32 bits	Continuous	LD= (EN ENO, s1, s2)	ORD>=(EN,s1,s2);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s1)	Device storing comparison data	ANY16	ANY32
	(s2)	Device storing comparison data	ANY16	ANY32
Output variable	ENO	Execution state	Bit	

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			
(s2)								●	●	●	●	●	●	●	▲1	▲2	●	●	●	●	●			

▲: Refer to "Cautions"

Function and operation explanation

These data comparison instructions are connected to other contacts in parallel.

The contents of the device specified by s1 is compared with the contents of the device specified by s2 in binary format, and a contact becomes conductive (ON) or non-conductive (OFF) depending on the comparison result.

16-bit instruction	32-bit instruction	ON condition	OFF condition
OR=	ORD=	s1 = s2	s1 ≠ s2
OR>	ORD>	s1 > s2	s1 ≤ s2
OR<	ORD<	s1 < s2	s1 ≥ s2
OR<>	OR<>	s1 ≠ s2	s1 = s2
OR<=	ORD<=	s1 ≤ s2	s1 > s2
OR>=	ORD>=	s1 ≥ s2	s1 < s2

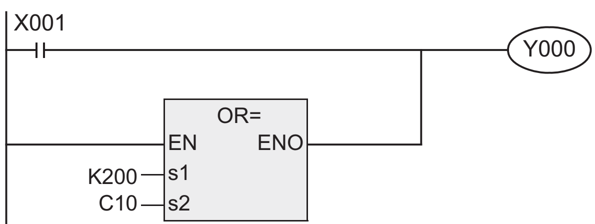
Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data. A 32-bit counter can be specified directly as it is a 32-bit long device. Use a global label to specify a device.
2) Negative values
   When the most significant bit is "1" in the data stored in the device specified by s1 or s2, it is regarded as a negative value in comparison.
   a) In the 16-bit operation: bit 15
   b) In the 32-bit operation: bit 31
3) When using 32-bit counters (including 32-bit high speed counters)
   Be sure to execute the 32-bit operation (such as ORD=, ORD> and ORD<) when comparing 32-bit counters.
   If a 32-bit counter is specified in the 16-bit operation (such as OR=, OR> and OR<), a program error or operation error will occur.
4) Some restrictions to applicable devices
   ▲1: Applicable only to the FX3U, FX3UC and FX3G PLCs.
   ▲2: Applicable only to the FX3U and FX3UC PLCs.

Program examples

[Structured ladder]

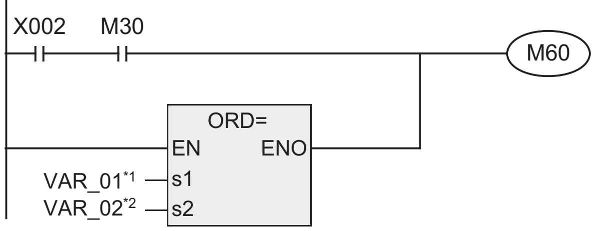
When X001 turns ON or when the current value of the counter C10 is "200", Y000 is driven.



[ST]

Y000:=X001 OR OR=(TRUE,K200,C10);

When X002 and M30 turn ON or when the contents of the data registers D101 and D100 are more than "K100,000", M60 is driven.



M60:=(X002 AND M30)OR ORD=(TRUE,VAR-01,VAR-02);

*1. VAR_01 is a global label and is defined as D100.
*2. VAR_02 is a global label and is defined as K100000.

7.23 Data Table Operation

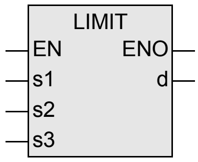
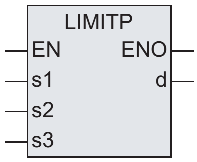
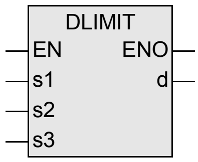
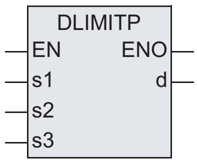
7.23.1 LIMIT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction provides the upper limit value and lower limit value for an input numeric value, and control the output value using these limit values.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
LIMIT	16 bits	Continuous		LIMIT(EN,s1,s2,s3,d);
LIMITP	16 bits	Pulse		LIMITP(EN,s1,s2,s3,d);
DLIMIT	32 bits	Continuous		DLIMIT(EN,s1,s2,s3,d);
DLIMITP	32 bits	Pulse		DLIMITP(EN,s1,s2,s3,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Lower limit value (minimum output value)	ANY16	ANY32
	s2	Upper limit value (maximum output value)	ANY16	ANY32
	s3	Input value controlled by the upper and lower limit values	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head device storing the output value controlled by the upper and lower limit values	ANY16	ANY32

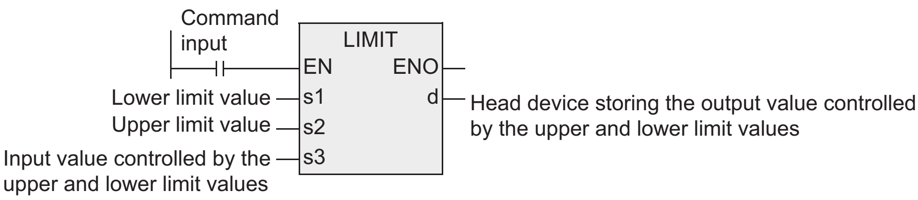
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	●	●			●	●	●			
s2								●	●	●	●	●	●	●	●	●			●	●	●			
s3								●	●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●			●					

Function and operation explanation

1. 16-bit operation (LIMIT/LIMITP)

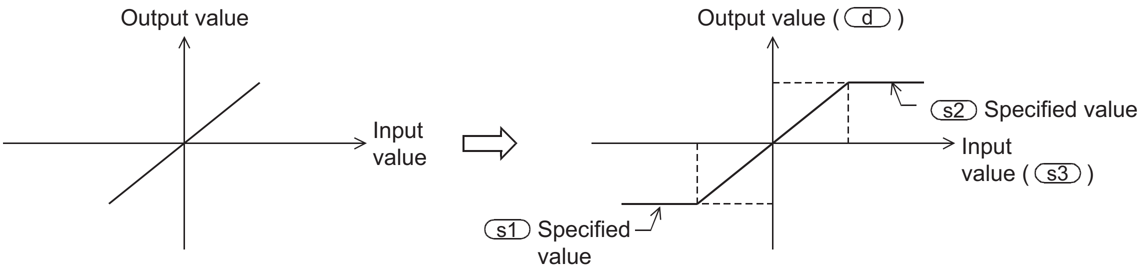
Depending on how the input value (16-bit binary value) of the device specified by (s3) compares to the upper and lower limit range between the devices specified by (s1) and (s2), the output value stored in the device specified by (d) is controlled.
The output value is controlled as shown below.



1) In the case of " (s1) lower limit value > (s3) input value " .... (s1) lower limit value → (d) output value

2) In the case of " (s2) upper limit value < (s3) input value " .... (s2) upper limit value → (d) output value

3) In the case of " (s1) lower limit value ≤ (s3) input value ≤ (s2) upper limit value " ... (s3) input value → (d) output value

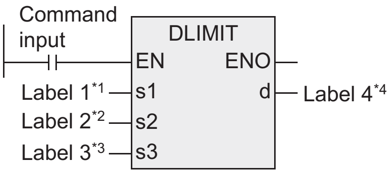


4) When controlling the output value using only the upper limit value, set "-32768" to the lower limit value of the device specified by (s1).

5) When controlling the output value using only the lower limit value, set "32767" to the upper limit value of the device specified by (s2).

2. 32-bit operation (DLIMIT/DLIMITP)

Depending on how the input value (32-bit binary value) of the device specified by (s3) compares to the range between the upper and lower limits specified by (s1) and (s2), the output value to be stored in the device specified by (d) is controlled.



*1. This defines the lower limit value.

*2. This defines the upper limit value.

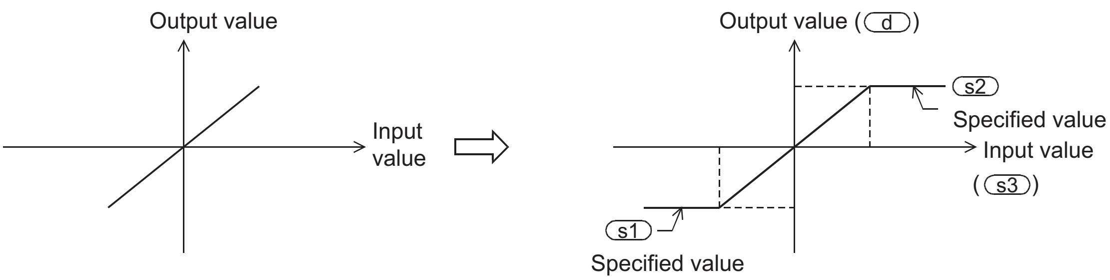
*3. This defines the control input value depending on the upper and lower limit control.

*4. This defines the head of the device that stores the output value depending on the result from the upper and lower limit control.

1) In the case of " (s1) lower limit value > (s3) input value "........ (s1) lower limit value → (d) output value

2) In the case of " (s2) upper limit value < (s3) input value "........ (s2) upper limit value → (d) output value

3) In the case of " (s1) lower limit value ≤ (s3) input value ≤ (s2) upper limit value "...... (s3) input value → (d) output value



4) When controlling the output value using only the upper limit value, set "-2,147,483,648" to the lower limit value specified in (s1).

5) When controlling the output value using only the lower limit value, set "2,147,483,647" to the upper limit value specified in (s2).

Caution

When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

Error

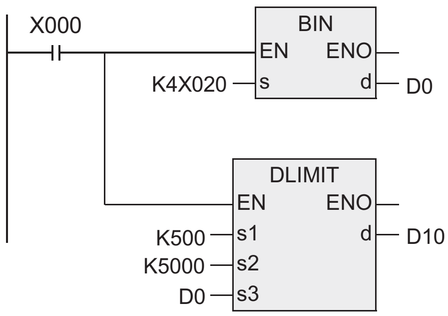
An operation error is caused when the instruction is executed in the setting status shown below. The error flag M8067 turns ON, and the error code (K6706) is stored in D8067.
"(Contents of the device specified by (s1)) > (contents of the device specified by (s2))

Program example

1. Program example 1

In the program example shown below, the BCD data set in X020 to X037 is controlled by the limit values "500" to "5000", and the controlled value is output to D1 when X000 turns ON.

[Structured ladder]

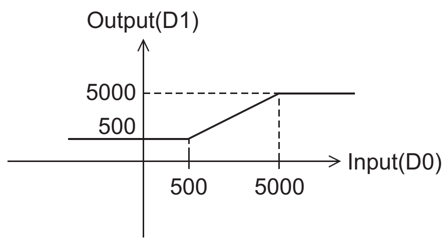


[ST]

```
BIN(X000,K4X020,D0);
D10:=DLIMIT(X000,K500,K5000,D0);
```

Operation

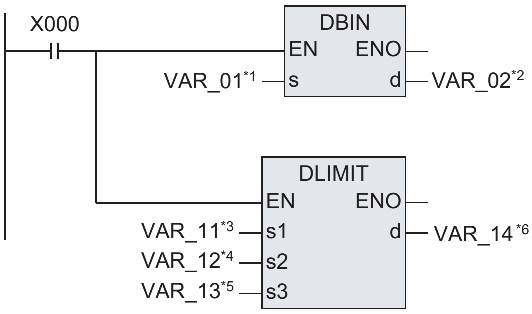
1) In the case of "D0 < 500", "500" is output to D1.
2) In the case of "500 ≤ D0 ≤ 5000", the value of D0 is output to D1.
3) In the case of "5000 < D0", "5000" is output to D1.



2. Program example 2

In the program example shown below, the BCD data set in X020 to X057 is controlled by the limit values "10000" and "1,000,000", and the controlled value is output to D11 and D10 when X000 turns ON.

[Structured ladder]



[ST]

```
DBIN(X000,VAR_01,VAR_02):
VAR_14:=DLIMIT(X000,VAR_11,VAR_12,VAR_13);
```

*1. VAR_01 is a global label and is defined as K8X020.

*2. VAR_02 is a global label and is defined as D0.

*3. VAR_11 is a global label and is defined as K10000.

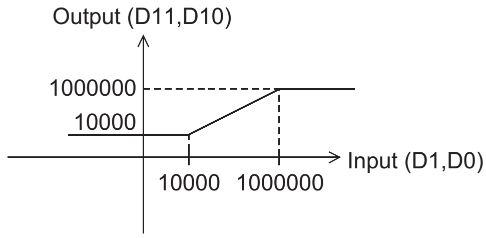
*4. VAR_12 is a global label and is defined as K1000000.

*5. VAR_13 is a global label and is defined as D0.

*6. VAR_14 is a global label and is defined as D10.

Operation 1

1) In the case of "(D1, D0) < 10000", "10000" is set to (D11, D10).
2) In the case of "10000 ≤ (D1, D0) ≤ 1,000,000", the value of (D1, D0) is output to (D11, D10).
3) In the case of "1,000,000 < (D1, D0)", "1,000,000" is output to (D11, D10)



1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

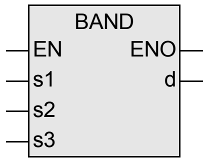
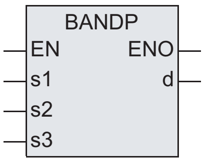
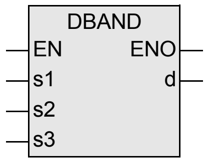
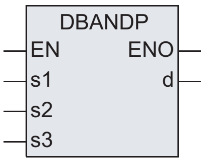
7.23.2 BAND

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction provides the upper limit value and lower limit value of the dead band for an input numeric value, and controls the output value using these limit values.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BAND	16 bits	Continuous		BAND(EN,s1,s2,s3,d);
BANDP	16 bits	Pulse		BANDP(EN,s1,s2,s3,d);
DBAND	32 bits	Continuous		DBAND(EN,s1,s2,s3,d);
DBANDP	32 bits	Pulse		DBANDP(EN,s1,s2,s3,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Lower limit value of the dead band (no-output band)	ANY16	ANY32
	s2	Upper limit value of the dead band (no-output band)	ANY16	ANY32
	s3	Input value controlled by the dead band	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Device storing the output value controlled by the dead band.	ANY16	ANY32

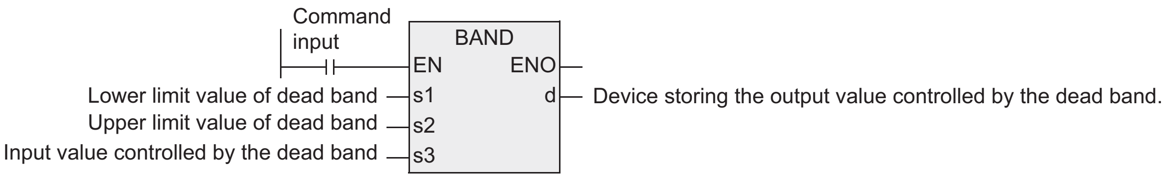
3. Applicable devices

Operand type	Bit Devices							Word Devices													Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	●	●			●	●	●			
s2								●	●	●	●	●	●	●	●	●			●	●	●			
s3								●	●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●			●					

Function and operation explanation

1. 16-bit operation (BAND/BANDP)

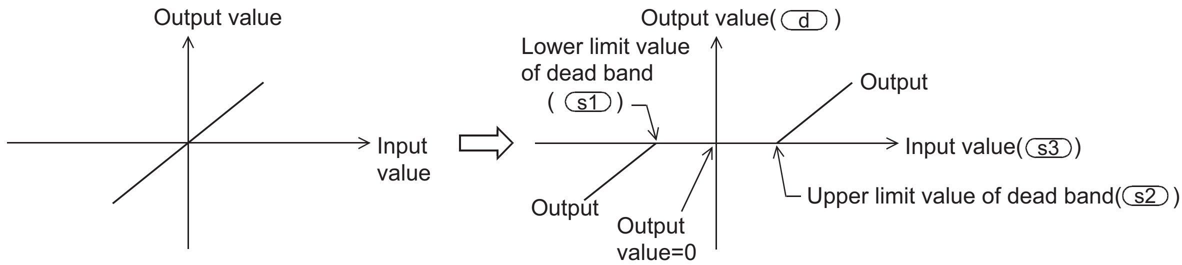
Depending on how the input value (16-bit binary value) of the device specified by (s3) compares to the upper and lower limit dead band range between the devices specified by (s1) and (s2), the output value to be stored in the device specified by (d) is controlled.
The output value is controlled as shown below.



1) In the case of " (s1) lower limit value > (s3) input value " .... (s3) input value - (s1) lower limit value → (d) output value

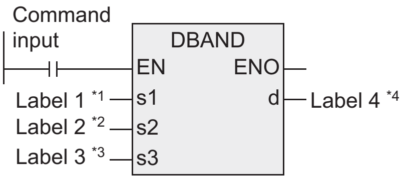
2) In the case of " (s2) upper limit value < (s3) input value " .... (s3) input value - (s2) upper limit value → (d) output value

3) In the case of " (s1) lower limit value ≤ (s3) input value ≤ (s2) upper limit value .... 0 → (d) output value



2. 32-bit operation (DBAND/DBANDP)

Depending on how the input value (32-bit binary value) specified by (s3) compares to the upper and lower limit dead band range between the devices specified by (s1) and (s2), the output value to be stored in the device specified by (d) is controlled.
The output value is controlled as shown below.



*1. This defines the lower limit value of the dead band.

*2. This defines the upper limit value of the dead band.

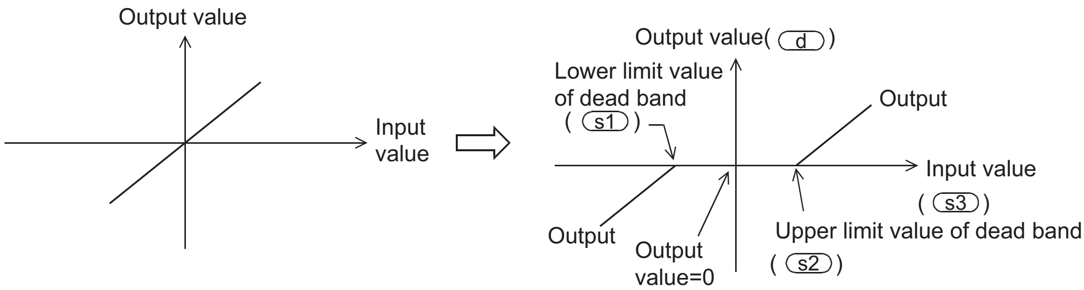
*3. This defines the input value controlled by the dead band.

*4. This defines the device storing the output value controlled by the dead band.

1) In the case of " (s1) lower limit value > (s3) input value "............................ (s3) input value - (s1) lower limit value → (d) output value

2) In the case of " (s2) upper limit value < (s3) input value "............................ (s3) input value - (s2) upper limit value → (d) output value

3) In the case of " (s1) lower limit value ≤ (s3) input value ≤ (s2) upper limit value "..0 → (d) output value



Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

2) When the output value overflows, it is handled as follows:

a) In the 16-bit operation
The output value is a 16-bit binary value with sign. Accordingly, if the operation result is outside the range from -32768 to 32767, it is handled as follows:

Lower limit value of dead band (s1) =10
Input value (s3) =-32,768
⇒
Output value =-32,768-10
=8000H-AH
=7FF6H
=32,758

b) In the 32-bit operation
The output value is a 32-bit binary value with sign. Accordingly, if the operation result is outside the range from -2,147,483,648 to 2,147,483,647, it is handled as follows:

Lower limit value of dead band (s1) =1000
Input value (s3) =-2,147,483,648
⇒
Output value=-2,147,483,648-1000
=8000000H-000003E8H
=7FFFFC18H
=2,147,482,648

Error

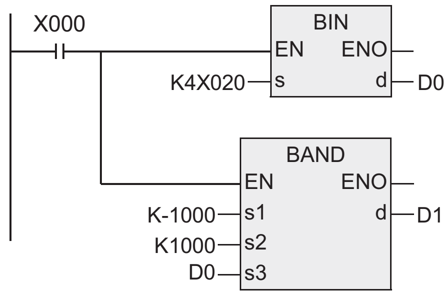
An operation error is caused when the instruction is executed in the setting status shown below. The error flag M8067 turns ON, and the error code (K6706) is stored in D8067.
(Contents of the device specified by (s1)) > (contents of the device specified by (s2))

Program example

1. Program example 1

In the program example shown below, the BCD data set in X020 to X037 is controlled by the dead band from "-1000" to "1000", and a controlled value is output to D1 when X000 turns ON.

[Structured ladder]

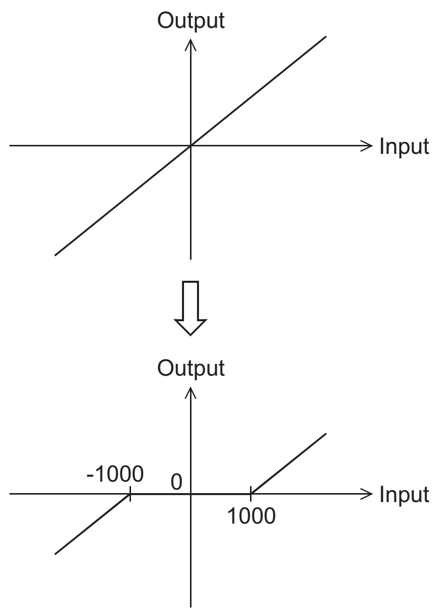


[ST]

```
BIN(X000,K4X020,D0);
D1:=BAND(X000,K-1000,K1000,D0);
```

Operation

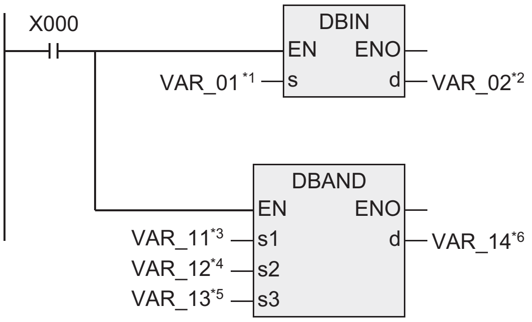
1) In the case of "D0 < (-1000)", "D0 - (-1000)" is set to D1.
2) In the case of "(-1000 ≤ D0 ≤ 1000", "0" is output to D1.
3) In the case of "1000 < D0", "D0 - 1000" is output to D1.



2. Program example 2

In the program example shown below, the BCD data set in X020 to X057 is controlled by the dead band from "-10000" to "10000", and a controlled value is output to D11 and D10 when X000 turns ON.

[Command input]



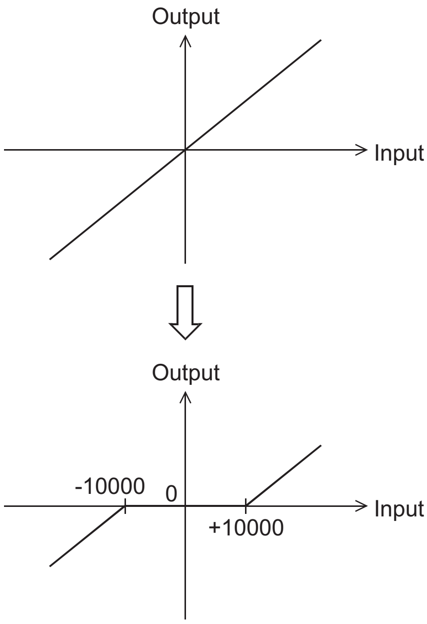
[ST]

```
DBIN(X000,VAR_01,VAR_02);
VAR_14:=DBAND(X000,VAR_11,VAR_12,VAR_13);
```

*1. VAR_01 is a global label and is defined as K8X020.
*2. VAR_02 is a global label and is defined as D0.
*3. VAR_11 is a global label and is defined as K-10000.
*4. VAR_12 is a global label and is defined as K10000.
*5. VAR_13 is a global label and is defined as D0.
*6. VAR_14 is a global label and is defined as D10.

Operation

1) In the case of "(D1, D0) < (-10000)", "(D1, D0) - (-10000)" is set to (D11, D10).
2) In the case of "(-10000 ≤ (D1, D0) ≤ 10000", "0" is output to (D11, D10).
3) In the case of "10000 < (D1, D0)", "(D1, D0) - 10000" is output to (D11, D10).



7.23.3 ZONE

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

Depending on whether the input value is positive or negative, the output value is controlled by the bias value specified.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
ZONE	16 bits	Continuous	ZONE EN ENO s1 d s2 s3	ZONE(EN,s1,s2,s3,d);
ZONEP	16 bits	Pulse	ZONEP EN ENO s1 d s2 s3	ZONEP(EN,s1,s2,s3,d);
DZONE	32 bits	Continuous	DZONE EN ENO s1 d s2 s3	DZONE(EN,s1,s2,s3,d);
DZONEP	32 bits	Pulse	DZONEP EN ENO s1 d s2 s3	DZONEP(EN,s1,s2,s3,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Negative bias value to be added to the input value	ANY16	ANY32
	s2	Positive bias value to be added to the input value	ANY16	ANY32
	s3	Input value controlled by the zone	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Head device storing the output value controlled by the zone	ANY16	ANY32

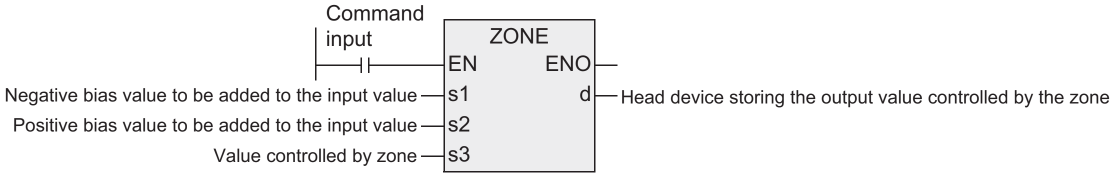
3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit specification				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	●	●			●	●	●			
s2								●	●	●	●	●	●	●	●	●			●	●	●			
s3								●	●	●	●	●	●	●	●	●			●					
d									●	●	●	●	●	●	●	●			●					

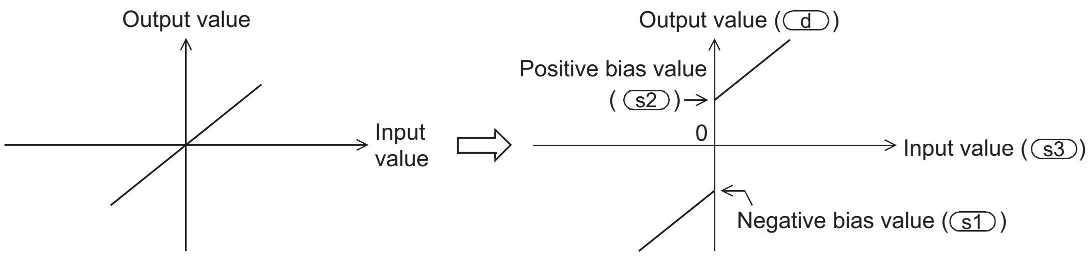
Function and operation explanation

1. 16-bit operation (ZONE/ZONEP)

The bias value specified by (s1) or (s2) is added to the input value specified by (s3), and output to the device specified by (d).
The bias value is added as shown below.



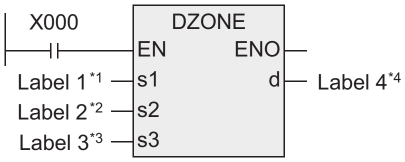
1) In the case of " (s3) [input value] < 0 " ..... (s3) [input value] + (s1) [negative bias value] → (d) [output value]
2) In the case of " (s3) [input value] = 0 " .... 0 → (d) [output value]
3) In the case of " (s3) [input value] > 0 " ..... (s3) [input value] + (s1) [positive bias value] → (d) [output value]



2. 32-bit operation (DZONE/DZONEP)

The bias value specified by (s1) or (s2) is added to the input value of the device specified by (s3), and output to the device specified by (d).

The bias value is added as shown below:



*1. This defines the negative bias value to be added to the input value.

*2. This defines the positive bias value to be added to the input value.

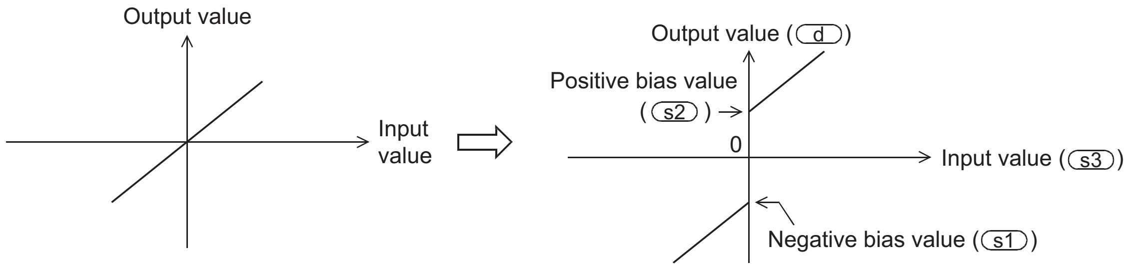
*3. This defines the input value controlled by the zone.

*4. This defines the head device that stores the output value controlled by the zone.

1) In the case of " (s3) input value < 0 ".................. (s3) input value + (s1) negative bias value → (d) output value

2) In the case of " (s3) input value = 0 "... 0 → (d) output value

3) In the case of " (s3) input value > 0 ".................. (s3) input value + (s2) positive bias value → (d) output value



Cautions

1) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device.
2) When the output value overflows, it is handled as follows:
   a) In the 16-bit operation
      The output value is a 16-bit binary value with sign. Accordingly, if the operation result is outside the range from -32768 to 32767, it is handled as follows:

      Negative bias value (s1) =-100
      Input value (s3) =-32,768
      ⇨
      Output value =-32,768+(-100)
      =8000H+FF9CH
      =7F9CH
      =32,668

   b) In the 32-bit operation
      The output value is a 32-bit binary value with sign. Accordingly, if the operation result is outside the range from -2,147,483,648 to 2,147,483,647, it is handled as follows:

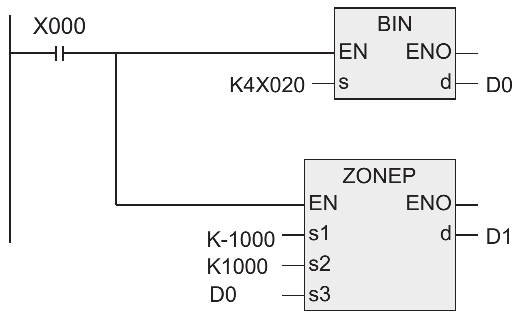
      Negative bias value (s1) =-1000
      Input value (s3) =-2,147,483,648
      ⇨
      Output value =-2,147,483,648+(-1000)
      =80000000H+FFFFFC18H
      =7FFFFC18
      =2,147,482,648

Program example

1. Program example 1

In the program example shown below, the BCD data set in X020 to X037 is controlled by the zone from "-1000" to "1000", and the controlled value is output to D1 when X000 turns ON.

[Structured ladder]

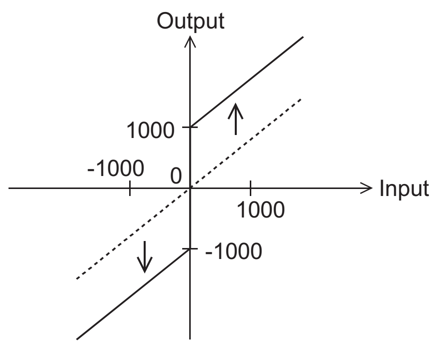


[ST]

```
BIN(X000,K4X020,D0);
D1:=ZONEP(X000,K-1000,K1000,D0);
```

Operation

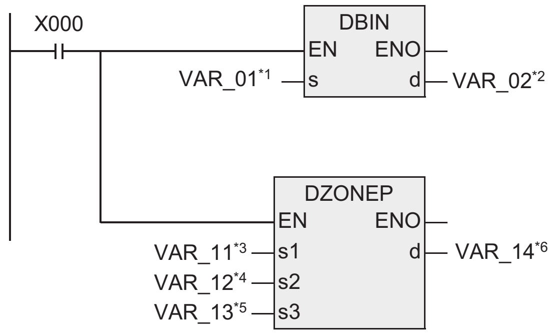
1) In the case of "D0 < 0", "D0 + (-1000)" is output to D1.
2) In the case of "D = 0", "0" is output to D1.
3) In the case of "0 < D0", "D0 + 1000" is output to D1.



2. Program example 2

In the program example below, the BCD data set in X020 to X057 is controlled by the zone from "-10000" to "10000", and the controlled value is output to D11 and D10 when X000 turns ON.

[Structured ladder]



[ST]

```
DBIN(X000,VAR_01,VAR_02);

VAR_14:=DZONEP(X000,VAR_11,VAR_12,VAR_13);
```

*1. VAR_01 is a global label and is defined as K8X020.

*2. VAR_02 is a global label and is defined as D0.

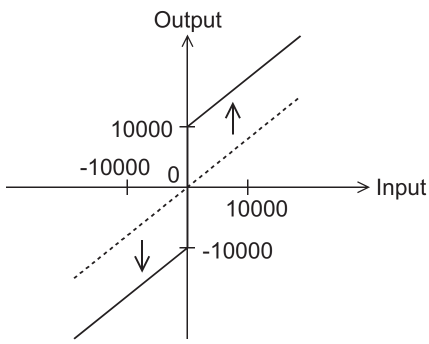
*3. VAR_11 is a global label and is defined as K-10000.

*4. VAR_12 is a global label and is defined as K10000.

*5. VAR_13 is a global label and is defined as D0.

Operation

1) In the case of "(D1, D0) < 0", "(D1, D0) + (-10000)" is output to (D11, D10).
2) In the case of "(D1, D0) = 0", "0" is output to (D11, D10).
3) In the case of "0 < (D1, D0)", "(D1, D0) + 10000" is output to (D11, D10)



1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

7.23.4 SCL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction executes scaling of the input value using a specified data table, and outputs the result.
SCL2 is also available with a different data table configuration for scaling.

→ **For SCL2 instruction, refer to Section 7.23.7.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
SCL	16 bits	Continuous	SCL EN ENO s1 d s2	SCL(EN,s1,s2,d);
SCLP	16 bits	Pulse	SCLP EN ENO s1 d s2	SCLP(EN,s1,s2,d);
DSCL	32 bits	Continuous	DSCL EN ENO s1 d s2	DSCL(EN,s1,s2,d);
DSCLP	32 bits	Pulse	DSCLP EN ENO s1 d s2	DSCLP(EN,s1,s2,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Input value used in scaling or device storing the input value	ANY16	ANY32
	s2	Head device storing the conversion table used in scaling	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Device storing the output value controlled by scaling	ANY16	ANY32

3. Applicable devices

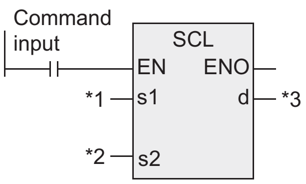
Operand type	Bit Devices: System user							Word Devices: Digit specification				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	●	●			●	●	●			
s2														●	●				●					
d									●	●	●	●	●	●	●	●			●					

Function and operation explanation

1. 16-bit operation (SCL/SCLP)

The input value of the device specified by s1 is processed by scaling for the specified conversion characteristics, and stored to a device number specified by d . Conversion for scaling is executed based on the data table stored in a device specified in s2 and later.
If the output data is not an integer, however, the number in the first decimal place is rounded.

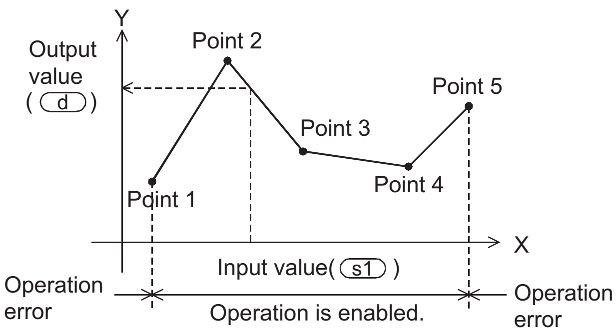
→ **For the method to set the conversion table for scaling, refer to the next page.**



*1. Input value used in scaling or device storing the input value

*2. Head device storing the conversion table used in scaling

*3. Device storing the output value controlled by scaling

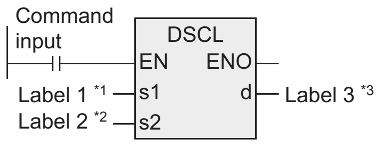


Conversion setting data table for scaling

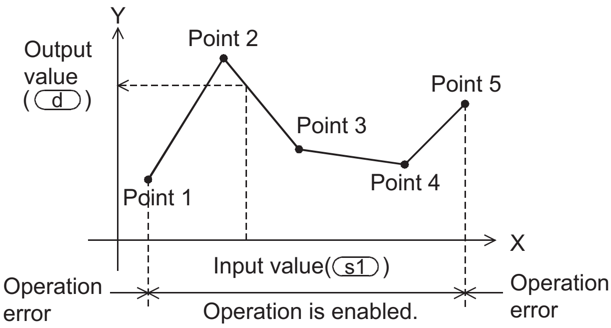
Set item		Device assignment in setting data table
Number of coordinate points ("5" in the case shown in the left figure)		s2
Point 1	X coordinate	s2 +1
	Y coordinate	s2 +2
Point 2	X coordinate	s2 +3
	Y coordinate	s2 +4
Point 3	X coordinate	s2 +5
	Y coordinate	s2 +6
Point 4	X coordinate	s2 +7
	Y coordinate	s2 +8
Point 5	X coordinate	s2 +9
	Y coordinate	s2 +10

2. 32-bit operation (DSCL/DSCLP)

The input value specified by (s1) is processed by scaling for the specified conversion table, and stored to a device number specified in (d). Conversion for scaling is executed based on the data table stored in a device specified in (s2) and later.
If the output data is not an integer, however, the number in the first decimal place is rounded.



*1. This defines the input value used in scaling or device storing the input value.
*2. This defines the head device storing the conversion table used in scaling.
*3. This defines the device storing the output value controlled by scaling.



Conversion setting data table for scaling

Set item		Device assignment in setting data table
Number of coordinate points ("5" in the case shown in the left figure)		[(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 3	X coordinate	[(s2)+11, (s2)+10]
	Y coordinate	[(s2)+13, (s2)+12]
Point 4	X coordinate	[(s2)+15, (s2)+14]
	Y coordinate	[(s2)+17, (s2)+16]
Point 5	X coordinate	[(s2)+19, (s2)+18]
	Y coordinate	[(s2)+21, (s2)+20]

3. Setting the conversion table for scaling

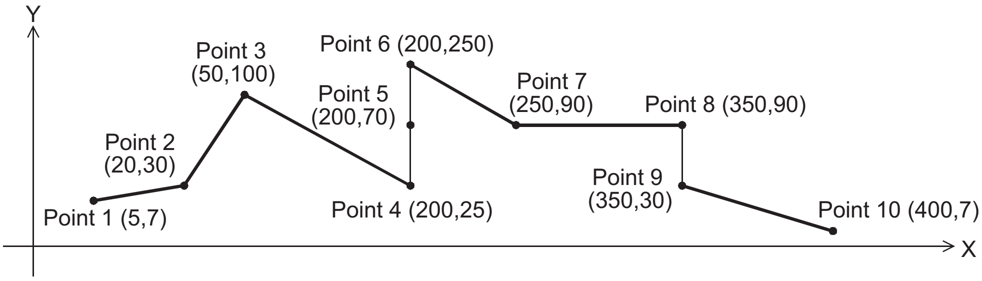
The conversion table for scaling is set based on the data table stored in a device specified in (s2) and later.
The data table has the following configuration:

→ **For a setting example, refer to the next page.**

Set item		Device assignment in setting data table	
		16-bit operation	32-bit operation
Number of coordinate points		(s2)	[(s2)+1, (s2)]
Point 1	X coordinate	(s2)+1	[(s2)+3, (s2)+2]
	Y coordinate	(s2)+2	[(s2)+5, (s2)+4]
Point 2	X coordinate	(s2)+3	[(s2)+7, (s2)+6]
	Y coordinate	(s2)+4	[(s2)+9, (s2)+8]
⋮	⋮	⋮	⋮
Point n (last)	X coordinate	(s2)+2n-1	[(s2)+4n-1, (s2)+4n-2]
	Y coordinate	(s2)+2n	[(s2)+4n+1, (s2)+4n]

4. Setting example of the conversion table for scaling

A setting example for the 16-bit operation is shown below.
For the 32-bit operation, set each item using a 32-bit binary value.
In the case of the conversion characteristics for scaling shown in the figure below, set the following data table.



Setting the conversion setting data table for scaling

Set item		Setting device and setting contents			Remarks
		When R0 is specified in (s2)		Setting contents	
Number of coordinate points		(s2)	R0	K10	
Point 1	X coordinate	(s2)+1	R1	K5	
	Y coordinate	(s2)+2	R2	K7	
Point 2	X coordinate	(s2)+3	R3	K20	
	Y coordinate	(s2)+4	R4	K30	
Point 3	X coordinate	(s2)+5	R5	K50	
	Y coordinate	(s2)+6	R6	K100	
Point 4	X coordinate	(s2)+7	R7	K200	When coordinates are specified using three points in this way, the output value can be set to an intermediate value. In this example, the output value (intermediate value) is specified by the Y coordinate of the point 5. If the x coordinate is the same at three points or more, the value at the second point is also output.
	Y coordinate	(s2)+8	R8	K25	
Point 5	X coordinate	(s2)+9	R9	K200	
	Y coordinate	(s2)+10	R10	K70	
Point 6	X coordinate	(s2)+11	R11	K200	
	Y coordinate	(s2)+12	R12	K250	
Point 7	X coordinate	(s2)+13	R13	K250	
	Y coordinate	(s2)+14	R14	K90	
Point 8	X coordinate	(s2)+15	R15	K350	When coordinates are specified using two points in this way, the output value is the Y coordinate at the next point. In this example, the output value is specified by the Y coordinate of the point 9.
	Y coordinate	(s2)+16	R16	K90	
Point 9	X coordinate	(s2)+17	R17	K350	
	Y coordinate	(s2)+18	R18	K30	
Point 10	X coordinate	(s2)+19	R19	K400	
	Y coordinate	(s2)+20	R20	K7	

Cautions

When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

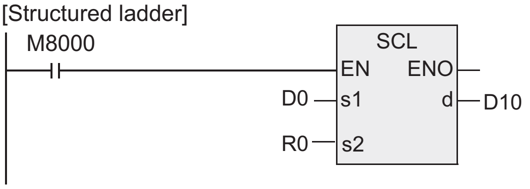
1) When the Xn data is not set in the ascending order in the data table (error code: K6706)
The data table is searched from the low-order side of device numbers in the data table in the operation. Accordingly, even if only some Xn data is set in the ascending order in the data table, the instruction is executed without operation error up to the area of the data table in which the Xn data is set in the ascending order.
2) When the device specified by (s1) is outside the data table range (error code: K6706)
3) When the value exceeds the 32-bit data range in the middle of operation (error code: K6706)
In this case, check whether the distance between points is not "65535" or more.
If the distance is "65535" or more, reduce the distance between points.

Program example

In the program example shown below, the value input to D0 is processed by scaling based on the conversion table for scaling set in R0 and later, and output to D10.

Program

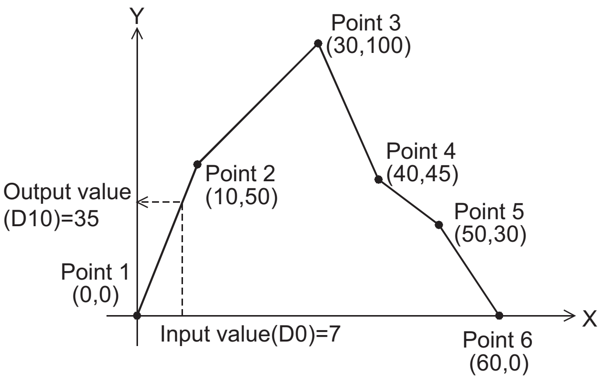
[Structured ladder]



[ST]

SCL (M8000, D0, R0, D10)

Operation



Conversion setting data table for scaling

Set item		Device	Setting contents
Number of coordinate points		R0	K6
Point 1	X coordinate	R1	K0
	Y coordinate	R2	K0
Point 2	X coordinate	R3	K10
	Y coordinate	R4	K50
Point 3	X coordinate	R5	K30
	Y coordinate	R6	K100
Point 4	X coordinate	R7	K40
	Y coordinate	R8	K45
Point 5	X coordinate	R9	K50
	Y coordinate	R10	K30
Point 6	X coordinate	R11	K60
	Y coordinate	R12	K0

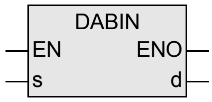
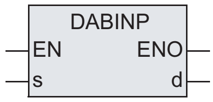
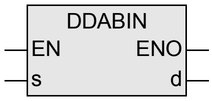
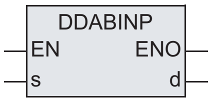
7.23.5 DABIN

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction converts numeric data expressed in decimal ASCII codes (30H to 39H) into binary data.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DABIN	16 bits	Continuous		DABIN(EN,s,d);
DABINP	16 bits	Pulse		DABINP(EN,s,d);
DDABIN	32 bits	Continuous		DDABIN(EN,s,d);
DDABINP	32 bits	Pulse		DDABINP(EN,s,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Head device storing data (ASCII codes) to be converted into binary data	String	String
Output variable	ENO	Execution state	Bit	
	(d)	Device storing conversion result	ANY16	ANY32

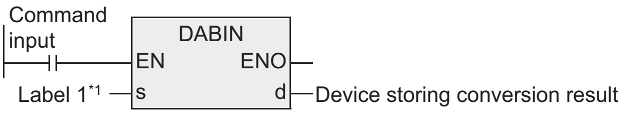
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)												●	●	●	●				●					
(d)									●	●	●	●	●	●	●	●	●	●	●					

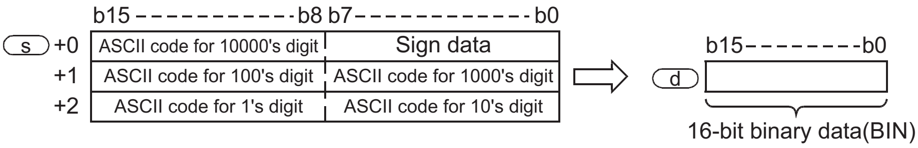
Function and operation explanation

1. 16-bit operation (DABIN/DABINP)

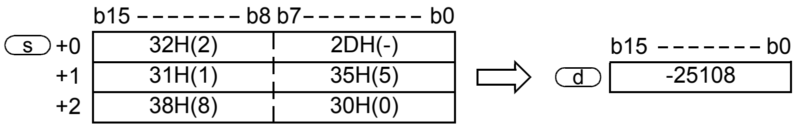
1) Data expressed in decimal ASCII codes (30H to 39H) and stored in the device specified by (s) is converted into 16-bit binary data, and stored in the device specified by (d).



*1. This defines the head device that stores data (ASCII codes) to be converted into binary data.



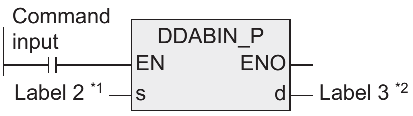
For example, when the device specified by (s) stores ASCII codes expressing "-25108", 16-bit binary data is stored in the device specified by (d) as follows:



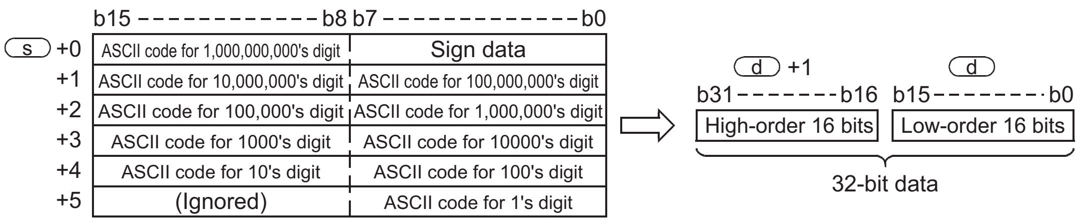
2) The numeric range of data stored in the device specified by (s) is from "-32768" to "32767".
3) As "sign data", "20H (space)" is set when the data to be converted is positive, and "2DH (-)" is set when the data to be converted is negative.
4) An ASCII code for each digit is within the range from 30H to 39H.
5) When an ASCII code for each digit is "20H (space)" or "00H (NULL)", it is handled as "30H".

2. 32-bit operation (DDABIN/DDABINP)

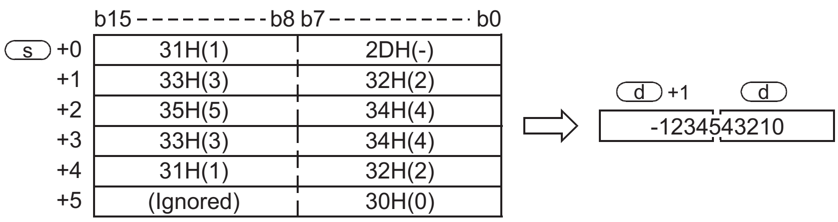
1) Data expressed in decimal ASCII codes (30H to 39H) and stored in the device specified by (s) is converted into 32-bit binary data, and stored in the device specified by (d).



*1. This defines the head device that stores data (ASCII codes) to be converted into binary data.
*2. This defines the device storing conversion result.



For example, when the device specified by (s) stores ASCII codes expressing "-1,234,543,210", 32-bit binary data is stored in the device specified by (d) as follows:



2) The numeric range of data stored in the device specified by (s) is from "-2,147,483,648" to "2,147,483,647".
The high-order byte of (s) + 5 is ignored.
3) As "sign data", "20H (space)" is set when the data to be converted is positive, and "2DH (-)" is set when the data to be converted is negative.
4) An ASCII code for each digit is within the range from 30H to 39H.
5) When an ASCII code for each digit is "20H (space)" or "00H (NULL)", it is handled as "30H".

Related instructions

Instruction	Description
ASCI	Converts hexadecimal codes into ASCII codes.
HEX	Converts ASCII codes into hexadecimal codes.
STR	Converts binary data into a character string (ASCII codes).
VAL	Converts a character string (ASCII codes) into binary data.
BINDA	Converts binary data into decimal ASCII codes (30H to 39H).

Cautions

1) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle array data or 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device
2) The FX3UC PLC of V 2.20 or later supports the DABIN instruction.

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the sign data is any value other than "20H (space)" or "2DH (-)".
   (Error code: K6706)
2) When an ASCII code for each digit stored in s to s + 2 (5) is any value other than "30H" to "39H", "20H (space)", or "00H (NULL)".
   (Error code: K6706)
3) When the numeric range of s to s + 2 (5) is outside the following range. (Error code: K6706)

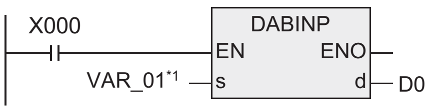
	Setting range
16-bit operation	-32768 to 32767
32-bit operation	-2,147,483,648 to 2,147,483,647

4) When the device specified by s exceeds the device range. (Error code: K6706)

Program example

In the program example below, the sign and decimal ASCII codes in five digits stored in D20 to D22 are converted into a binary value and stored in D0 when X000 turns ON.

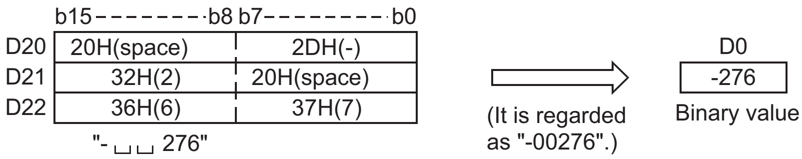
[Structured ladder]



[ST]

```
DABINP(X000, VAR_01, D0);
```

*1. VAR_01 is a global label and is defined as D20.



7.23.6 BINDA

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction converts binary data into decimal ASCII codes (30H to 39H).

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
BINDA	16 bits	Continuous	BINDA EN ENO s d	BINDA(EN,s,d);
BINDAP	16 bits	Pulse	BINDAP EN ENO s d	BINDAP(EN,s,d);
DBINDA	32 bits	Continuous	DBINDA EN ENO s d	DBINDA(EN,s,d);
DBINDAP	32 bits	Pulse	DBINDAP EN ENO s d	DBINDAP(EN,s,d);

2. Set data

Variable		Description	Data type	
			16-bit operation	32-bit operation
Input variable	EN	Execution condition	Bit	
	(s)	Device storing binary data to be converted into ASCII codes	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	(d)	Head device storing conversion result	String	String

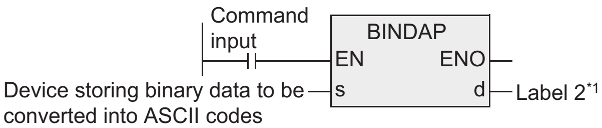
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others					
	System user							Digit specification				Special unit				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)								●	●	●	●	●	●	●	●	●	●	●	●	●	●			
(d)												●	●	●	●				●					

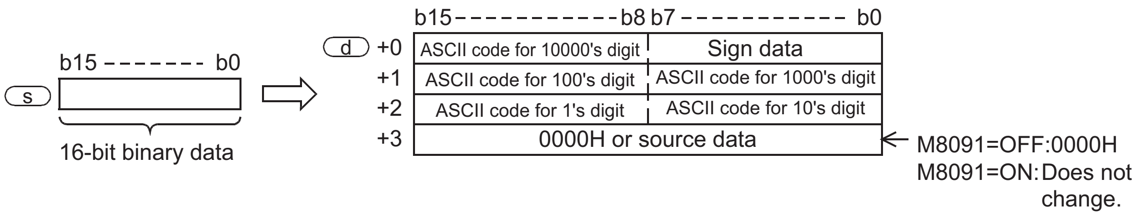
Function and operation explanation

1. 16-bit operation (BINDA/BINDAP)

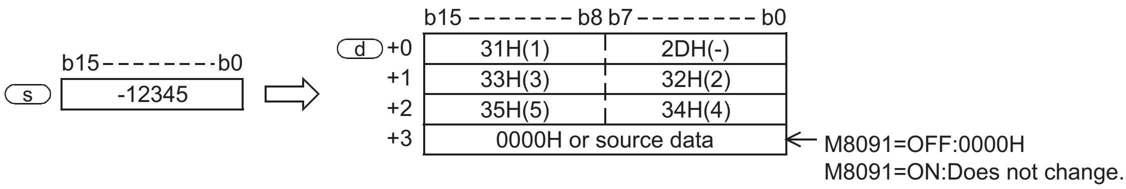
1) Each digit of 16-bit binary data stored in the device specified by (s) is converted into an ASCII code (30H to 39H), and stored in the device specified by (d) and later.



*1. This defines the head device that stores the conversion result.



For example, when the device specified by (s) stores "-12345", the conversion result is stored in the device specified by (d) and later as follows:

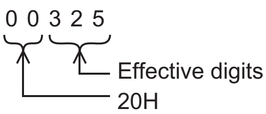


2) The numeric range of 16-bit binary data stored in the device specified by (s) is from "-32768" to "32767".

3) The conversion result stored in the device specified by (d) is as follows:

a) As "sign data", "20H (space)" is set when the 16-bit binary data stored in the device specified by (s) is positive, and "2DH (-)" is set when 16-bit binary data stored in the device specified by (s) is negative.

b) "20H (space)" is stored for "0" on the left side of the effective digits (zero suppression).



c) (d) + 3 is set as follows depending on the ON/OFF status of M8091.

ON/OFF status	Contents of processing
M8091=OFF	(d) +3 is set to "0000H (NULL)".
M8091=ON	(d) +3 does not change

2. 32-bit operation (DBINDA/DBINDAP)

1) Each digit of 32-bit binary data stored in the device specified by (s) is converted into an ASCII code (30H to 39H), and stored in the device specified by (d) and later.

*1. This defines the device that stores binary data to be converted into ASCII codes.
*2. This defines the head device that stores the conversion result.

(s)+1: High-order 16 bits; (s): Low-order 16 bits — 32-bit data (BIN) ⇨

	b15 ------ b8	b7 ------ b0
(d)+0	ASCII code for 1,000,000,000's digit	Sign data
+1	ASCII code for 10,000,000's digit	ASCII code for 100,000,000's digit
+2	ASCII code for 100,000's digit	ASCII code for 1,000,000's digit
+3	ASCII code for 1000's digit	ASCII code for 10000's digit
+4	ASCII code for 10's digit	ASCII code for 100's digit
+5	00H or 20H	ASCII code for 1's digit

00H or 20H: M8091=OFF:00H, M8091=ON:20H

For example, when the device specified by (s) stores "-12,345,678", the conversion result is stored in the device specified by (d) and later as follows:

(s)+1, (s): -12345678 ⇨

	b15 ------ b8	b7 ------ b0
(d)+0	20H (space)	2DH(-)
+1	31H(1)	20H (space)
+2	33H(3)	32H(2)
+3	35H(5)	34H(4)
+4	37H(7)	36H(6)
+5	00H or 20H	38H(8)

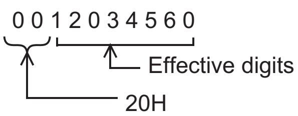
00H or 20H: M8091=OFF:00H, M8091=ON:20H

2) The numeric range of 32-bit binary data stored in the device specified bay (s) is from "-2,147,483,648" to "2,147,483,647".

3) The conversion result stored in the device specified by (d) is as follows:

a) As "sign data", "20H (space)" is set when the 32-bit binary data stored in the device specified by (s) is positive, and "2DH (-)" is set when 32-bit binary data stored in the device specified by (s) is negative.

b) "20H (space)" is stored for "0" on the left side of the effective digits (zero suppression).



c) The high-order byte of (d) + 5 is set as follows depending on the ON/OFF status of M8091.

ON/OFF status	Contents of processing
M8091=OFF	The high-order byte of (d) + 5 is set to "00H (NULL)".
M8091=ON	The high-order byte of (d) + 5 is set to "20H (space)".

Related devices

Device	Name	Description
M8091	Output character quantity selector signal	• For 16-bit operation - When M8091 is OFF, (d) + 3 is set to "0000H (NULL)". - When M8091 is ON, (d) + 3 does not change. • For 32-bit operation - When M8091 is OFF, the high-order byte of (d) + 5 is set to "00H (NULL)". - When M8091 is ON, the high-order byte of (d) + 5 is set to "20H (space)".

Related instructions

Instruction	Description
ASCI	Converts hexadecimal codes into ASCII codes.
HEX	Converts ASCII codes into hexadecimal codes.
DESTR	Converts binary floating point data into a character string data (ASCII code) with the specified number of digits.
DEVAL	Converts character string data (ASCII code) into binary floating point data.
DABIN	Converts numeric value data expressed in decimal ASCII codes (30H to 39H) into binary data.

Cautions

1) When handling array data or 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle array data or 32-bit data.
   A 32-bit counter can be specified directly as it is a 32-bit long device.
   Use a global label to specify a device
2) The FX3UC PLC of V 2.20 or later supports the DABIN instruction.
3) Occupied device points
   The table below shows the occupied device points of the device specified by (d) for 16-bit operation (BINDA/BINDAP) when M8091 is ON or OFF and 32-bit operation (DBINDA/DBINDAP).

		Occupied points of (d)
16-bit operation	M8091=ON	3
	M8091=OFF	4
32-bit operation		6

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the occupied device points of the ASCII code character string in the device specified by (d) exceed the corresponding device range. (Error code: K6706)

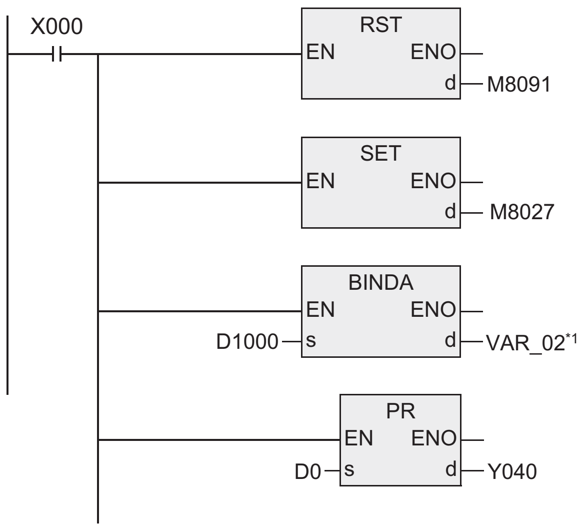
Program example

In the program example below, 16-bit binary data stored in D1000 is converted into decimal ASCII codes when X000 is set to ON, and the ASCII codes converted by PR (FNC77) instruction are output one by one in the time division method to Y040 to Y051.

By setting to OFF the output character selector signal M8091 and setting to ON PR mode flag M8027, ASCII codes up to "00H" are output.

→ **For PR mode flag and PR instruction, refer to Section 7.8.8.**

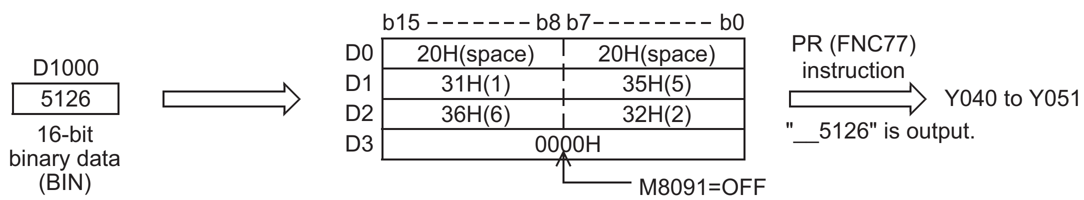
[Structured ladder]



[ST]

```
RST(X000,M8091);
SET(X000,M8027);
BINDA(X000,D1000,VAR_02);
Y040:=PR(X000,D0);
```

*1. VAR_02 is a global label and is defined as D0.



7.23.7 SCL2

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction executes scaling of the input value using a specified data table, and outputs the result.
SCL instruction is also available with a different data table configuration for scaling.

→ **For SCL instruction, refer to Section 7.23.4.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
SCL2	16 bits	Continuous	SCL2 EN ENO s1 d s2	SCL2(EN,s1,s2,d);
SCL2P	16 bits	Pulse	SCL2P EN ENO s1 d s2	SCL2P(EN,s1,s2,d);
DSCL2	32 bits	Continuous	DSCL2 EN ENO s1 d s2	DSCL2(EN,s1,s2,d);
DSCL2P	32 bits	Pulse	DSCL2P EN ENO s1 d s2	DSCL2P(EN,s1,s2,d);

2. Set data

Variable		Description	Data type: 16-bit operation	Data type: 32-bit operation
Input variable	EN	Execution condition	Bit	
	s1	Input value used in scaling or device storing the input value	ANY16	ANY32
	s2	Head device storing the conversion table used in scaling	ANY16	ANY32
Output variable	ENO	Execution state	Bit	
	d	Device storing the output value controlled by scaling	ANY16	ANY32

3. Applicable devices

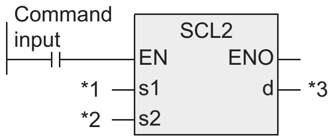
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1								●	●	●	●	●	●	●	●	●			●	●	●			
s2														●	●				●					
d									●	●	●	●	●	●	●	●			●					

Function and operation explanation

1. 16-bit operation (SCL2/SCL2P)

The input value specified in (s1) is processed by scaling for the specified conversion characteristics, and stored to a device number specified in (d). Conversion for scaling is executed based on the data table stored in a device specified in (s2) and later.
If the output data is not an integer, however, the number in the first decimal place is rounded.

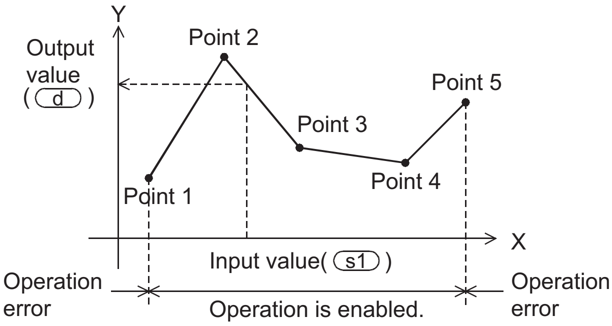
→ **For the method to set the conversion table for scaling, refer to the next page.**



*1. Input value used in scaling or device storing the input value

*2. Head device storing the conversion table used in scaling

*3. Device storing the output value controlled by scaling

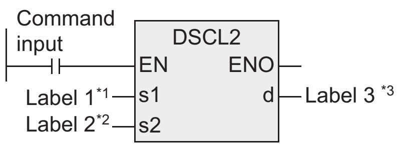


Conversion setting data table for scaling

Set item		Device assignment in setting data table
Number of coordinate points ("5" in the case shown in the left figure)		(s2)
X coordinate	Point 1	(s2)+1
	Point 2	(s2)+2
	Point 3	(s2)+3
	Point 4	(s2)+4
	Point 5	(s2)+5
Y coordinate	Point 1	(s2)+6
	Point 2	(s2)+7
	Point 3	(s2)+8
	Point 4	(s2)+9
	Point 5	(s2)+10

2. 32-bit operation (DSCL2/DSCL2P)

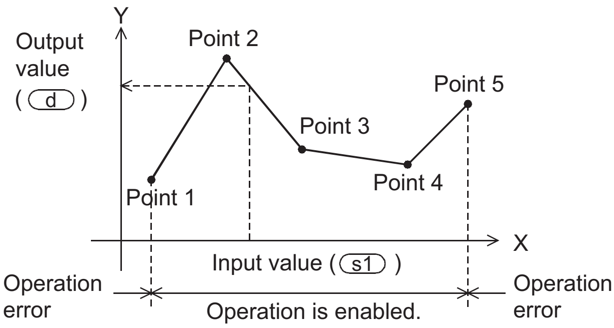
The input value specified by s1 is processed by scaling for the specified conversion table, and stored to a device number specified in d . Conversion for scaling is executed based on the data table stored in a device specified in s2 and later.
If the output data is not an integer, however, the number in the first decimal place is rounded.



*1. This defines the input value used in scaling or device storing the input value.

*2. This defines the head device storing the conversion table used in scaling.

*3. This defines the device storing the output value controlled by scaling.



Conversion setting data table for scaling

Set item		Conversion setting data table for scaling
Number of coordinate points ("5" in the case shown in the left figure)		[s2 +1, s2]
X coordinate	Point 1	[s2 +3, s2 +2]
X coordinate	Point 2	[s2 +5, s2 +4]
X coordinate	Point 3	[s2 +7, s2 +6]
X coordinate	Point 4	[s2 +9, s2 +8]
X coordinate	Point 5	[s2 +11, s2 +10]
Y coordinate	Point 1	[s2 +13, s2 +12]
Y coordinate	Point 2	[s2 +15, s2 +14]
Y coordinate	Point 3	[s2 +17, s2 +16]
Y coordinate	Point 4	[s2 +19, s2 +18]
Y coordinate	Point 5	[s2 +21, s2 +20]

3. Setting the conversion table for scaling

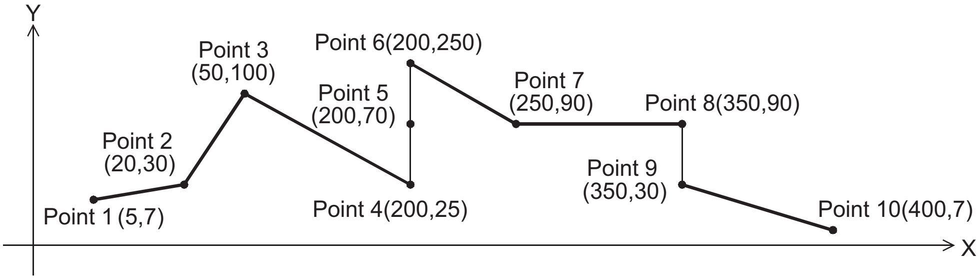
The conversion table for scaling is set based on the data table stored in a device specified in (s2) and later.
The data table has the following configuration:

→ **For a setting example, refer to the next page.**

Set item		Device assignment in setting data table	
		16-bit operation	32-bit operation
Number of coordinate points		(s2)	[(s2)+1, (s2)]
X coordinate	Point 1	(s2)+1	[(s2)+3, (s2)+2]
	Point 2	(s2)+2	[(s2)+5, (s2)+4]
	⋮	⋮	⋮
	Point n (last)	(s2)+n	[(s2)+2n+1, (s2)+2n]
Y coordinate	Point 1	(s2)+n+1	[(s2)+2n+3, (s2)+2n+2]
	Point 2	(s2)+n+2	[(s2)+2n+5, (s2)+2n+4]
	⋮	⋮	⋮
	Point n (last)	(s2)+2n	[(s2)+4n+1, (s2)+4n]

Setting example of the conversion table for scaling

A setting example for the 16-bit operation is shown below.
For the 32-bit operation, set each item using a 32-bit binary value.
In the case of the conversion characteristics for scaling shown in the figure below, set the following data table.



Setting the conversion setting data table for scaling

Set item		Setting device and setting contents: When R0 is specified in (s2)		Setting contents	Remarks
Number of coordinate points		(s2)	R0	K10	
X coordinate	Point 1	(s2)+1	R1	K5	
	Point 2	(s2)+2	R2	K20	
	Point 3	(s2)+3	R3	K50	
	Point 4	(s2)+4	R4	K200	Refer to *1.
	Point 5	(s2)+5	R5	K200	
	Point 6	(s2)+6	R6	K200	
	Point 7	(s2)+7	R7	K250	
	Point 8	(s2)+8	R8	K350	Refer to *2.
	Point 9	(s2)+9	R9	K350	
	Point 10	(s2)+10	R10	K400	
Y coordinate	Point 1	(s2)+11	R11	K7	
	Point 2	(s2)+12	R12	K30	
	Point 3	(s2)+13	R13	K100	
	Point 4	(s2)+14	R14	K25	Refer to *1.
	Point 5	(s2)+15	R15	K70	
	Point 6	(s2)+16	R16	K250	
	Point 7	(s2)+17	R17	K90	
	Point 8	(s2)+18	R18	K90	Refer to *2.
	Point 9	(s2)+19	R19	K30	
	Point 10	(s2)+20	R20	K7	

*1. When coordinates are specified using three points as shown in the points 4, 5 and 6, the output value can be set to an intermediate value.
In this example, the output value (intermediate value) is specified by the Y coordinate of the point 5.
If the X coordinate is same at three points or more, the value at the second point is output also.

*2. When coordinates are specified using two points as shown in the points 8 and 9, the output value is the Y coordinate at the next point.
In this example, the output value is specified by the Y coordinate of the point 9.

Cautions

When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

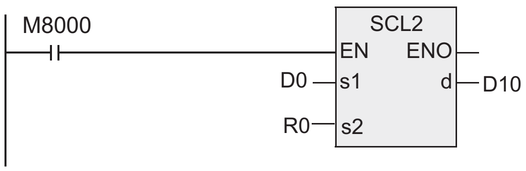
1) When the Xn data is not set in the ascending order in the data table (error code: K6706)
The data table is searched from the low-order side of device numbers in the data table in the operation. Accordingly, even if only some Xn data is set in the ascending order in the data table, the instruction is executed without operation error up to the area of the data table in which the Xn data is set in the ascending order.
2) When the device specified by (s1) is outside the data table range (error code: K6706)
3) When the value exceeds the 32-bit data range in the middle of operation (error code: K6706)
In this case, check whether the distance between points is not "65535" or more.
If the distance is "65535" or more, reduce the distance between points.

Program example

In the program example shown below, the value input to D0 is processed by scaling based on the conversion table for scaling set in R0 and later, and output to D10.

Program

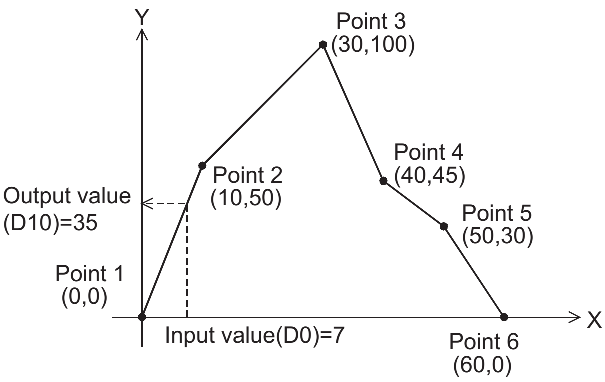
[Structured ladder]



[ST]

```
D10:=SCL2(M8000,D0,R0);
```

Operation



Conversion setting data table for scaling

Set item		Device	Setting contents
Number of coordinate points		R0	K6
X coordinate	Point 1	R1	K0
	Point 2	R2	K10
	Point 3	R3	K30
	Point 4	R4	K40
	Point 5	R5	K50
	Point 6	R6	K60
Y coordinate	Point 1	R7	K0
	Point 2	R8	K50
	Point 3	R9	K100
	Point 4	R10	K45
	Point 5	R11	K30
	Point 6	R12	K0

7.24 External Device Communication (Inverter Communication)

7.24.1 IVCK

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	×	×	×	×	×	×

Outline

This instruction reads the operation status of an inverter to a PLC using the computer link operation function of the inverter. Applicable inverters vary depending on the version.
This instruction corresponds to the EXTR (K10) instruction in the FX2N and FX2NC series.

→ **For detailed explanation of the instruction, refer to the Data Communication Edition manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
IVCK	16 bits	Continuous	IVCK block: inputs EN, s1, s2, n; outputs ENO, d	IVCK(EN,s1,s2,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Inverter station number	ANY16
	s2	Inverter instruction code	ANY16
	n	Channel to be used (K1:ch1,K2:ch2*1)	ANY16
Output variable	ENO	Execution state	Bit
	d	Device storing the read value	ANY16

*1. "ch2" is not available for use for the FX3G PLCs of 14- and 24-point types.

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1	▲2			●	●	●			
s2														●	▲1	▲2			●	●	●			
d									●	●	●			●	▲1	▲2			●					
n																				●	●			

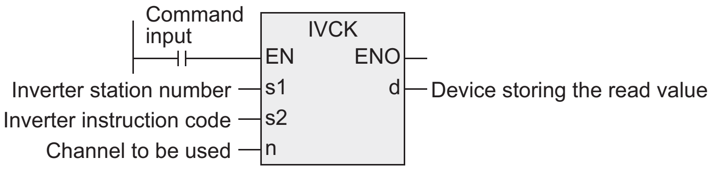
▲: Refer to "Cautions".

Function and operation explanation

→ **For detailed explanation of the instruction, refer to the Data Communication Edition manual.**

1. 16-bit operation (IVCK)

The operation status corresponding to the instruction code*2 specified in the device specified by (s2) of an inverter*1 connected to communication port n whose station number is specified in the device specified by (s1) is read and transferred to the device specified by (d).



*1. Mitsubishi Electric's FREQROL - F700, A700, E700, D700, V500, F500, A500, E500 and S500 series general purpose inverters

*2. Refer to the instruction code list.
Refer to the pages in the inverter manual on which the computer link function is explained in detail.

2. Instruction codes of inverters

The table below shows the inverter instruction codes specified in (s2) along with their functions.
For instruction codes, refer to the pages in the inverter manual where the computer link function is explained in detail.

Instruction code specified in (s2)	Read contents	Corresponding inverter								
		F700	A700	E700	D700	V500	F500	A500	E500	S500
H7B	Operation mode	✓	✓	✓	✓	✓	✓	✓	✓	✓
H6F	Output frequency (number of rotations)	✓	✓	✓	✓	✓	✓	✓	✓	✓
H70	Output current	✓	✓	✓	✓	✓	✓	✓	✓	✓
H71	Output voltage	✓	✓	✓	✓	✓	✓	✓	✓	-
H72	Special monitor	✓	✓	✓	✓	✓	✓	✓	-	-
H73	Special monitor selection number	✓	✓	✓	✓	✓	✓	✓	-	-
H74	Abnormal contents	✓	✓	✓	✓	✓	✓	✓	✓	✓
H75	Abnormal contents	✓	✓	✓	✓	✓	✓	✓	✓	✓
H76	Abnormal contents	✓	✓	✓	✓	✓	✓	✓	✓	-
H77	Abnormal contents	✓	✓	✓	✓	✓	✓	✓	✓	-
H79	Inverter status monitor (extension)	✓	✓	✓	✓	-	-	-	-	-
H7A	Inverter status monitor	✓	✓	✓	✓	✓	✓	✓	✓	✓
H6E	Set frequency (read from E2PROM)	✓	✓	✓	✓	✓	✓	✓	✓	✓
H6D	Set frequency (read from RAM)	✓	✓	✓	✓	✓	✓	✓	✓	✓

3. Related devices

→ **For the instruction execution complete flag use method, refer to Section 1.3.4.**

Number ch1	Number ch2	Description
M8029		Instruction execution complete
M8063	M8438	Serial communication error
M8151	M8156	Inverter communicating*1
M8152	M8157	Inverter communication error*1
M8153	M8158	Inverter communication error latch*1
M8154	M8159	IVBWR instruction error*1

Number ch1	Number ch2	Description
D8063	D8438	Error code of serial communication error
D8150	D8155	Response wait time in inverter communication
D8151	D8156	Step number in inverter communication*2
D8152	D8157	Error code of inverter communication error*1
D8153	D8158	Latch of inverter communication error occurrence step*2
D8154	D8159	IVBWR instruction error parameter number*2

*1. Cleared when the PLC mode switches from STOP to RUN.

*2. Initial value: -1

Cautions

→ **For other cautions, refer to the Data Communication Edition manual.**

1) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
2) It is not permitted to use the RS and RS2 instructions and an inverter communication instruction (IVCK, IVDR, IVRD, IVWR and IVBWR) for the same port.
3) Two or more inverter communication instructions (IVCK, IVDR, IVRD, IVWR and IVBWR) can be driven for the same port at the same time.
4) Some restrictions to applicable devices
   ▲1: Applicable to the FX3U, FX3UC and FX3G PLCs only.
   ▲2: Applicable to the FX3U and FX3UC PLCs only.

PLC applicable version

The table below shows PLC versions applicable to each inverter.

PLC	FREQROL-V500/F500/A500/E500/S500	FREQROL-F700/A700	FREQROL-E700/D700
FX3G	Ver.1.10 or later		
FX3U	Ver.2.20 or later		Ver.2.32 or later
FX3UC	Ver.1.00 or later	Ver.2.20 or later	Ver.2.32 or later

7.24.2 IVDR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	×	×	×	×	×	×

Outline

This instruction writes a inverter operation required control value to the PLC using the computer link operation function of the inverter.
This instruction corresponds to the EXTR (K11) instruction in the FX2N and FX2NC series PLCs.

→ **For detailed explanation of the instruction, refer to the Data Communication Edition manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
IVDR	16 bits	Continuous	IVDR EN ENO s1 s2 s3 n	IVDR(EN,s1,s2,s3,n);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s1)	Inverter station number	ANY16
	(s2)	Inverter instruction code	ANY16
	(s3)	Set value to be written to the inverter parameter or device storing the data to be set.	ANY16
	(n)	Channel to be used (K1:ch1,K2:ch2*1)	ANY16
Output variable	ENO	Execution state	Bit

*1. "ch2" is not available for use for the FX3G PLCs of 14- and 24-point types.

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)														●	▲1	▲2			●	●	●			
(s2)														●	▲1	▲2			●	●	●			
(s3)								●	●	●	●			●	▲1	▲2			●	●	●			
(n)																				●	●			

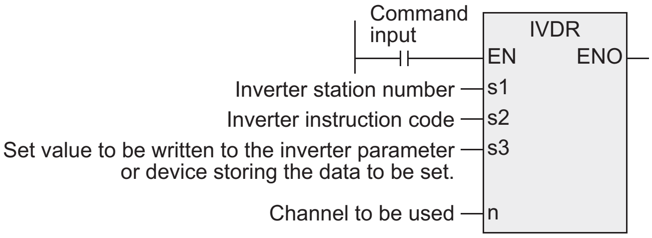
▲: Refer to "Cautions".

Function and operation explanation

→ For detailed explanation of the instruction, refer to the Data Communication Edition manual.

1. 16-bit operation (IVDR)

The control value specified in the device specified by (s3) is written to the instruction code[*2] in the device specified by (s2) of an inverter[*1] connected to a communication port n whose station number is specified in the device specified by (s1).



*1. Mitsubishi Electric's FREQROL - F700, A700, E700, D700, V500, F500, A500, E500 and S500 series general purpose inverters

*2. Refer to the instruction code list.
Refer to the pages in the inverter manual on which the computer link function is explained in detail.

2. Instruction codes of inverters

The table below shows the inverter instruction codes specified in (s2) along with their functions.
For instruction codes, refer to the pages in the inverter manual where the computer link function is explained in detail.

Hexadecimal instruction code of inverter specified in (s2)	Written contents	Corresponding inverter								
		F700	A700	E700	D700	V500	F500	A500	E500	S500
HFB	Operation mode	✓	✓	✓	✓	✓	✓	✓	✓	✓
HF3	Special monitor selection number	✓	✓	✓	✓	✓	✓	✓	-	-
HF9	Operation command (extension)	✓	✓	✓	✓	-	-	-	-	-
HFA	Operation command	✓	✓	✓	✓	✓	✓	✓	✓	✓
HEE	Set frequency (written to EEPROM)	✓	✓	✓	✓	✓	✓	✓	✓	✓
HED	Set frequency (written to RAM)	✓	✓	✓	✓	✓	✓	✓	✓	✓
HFD	Inverter reset	✓	✓	✓	✓	✓	✓	✓	✓	✓
HF4	Abnormal contents all clear	✓	✓	✓	✓	-	✓	✓	✓	✓
HFC	Parameter all clear	✓	✓	✓	✓	✓	✓	✓	✓	✓
HFC	User clear	-	-	-	-	-	✓	✓	-	-

3. Related devices

→ For the instruction execution complete flag use method, refer to Section 1.3.4.

Number		Description
ch1	ch2	
M8029		Instruction execution complete
M8063	M8438	Serial communication error
M8151	M8156	Inverter communicating[*1]
M8152	M8157	Inverter communication error[*1]
M8153	M8158	Inverter communication error latch[*1]
M8154	M8159	IVBWR instruction error[*1]

Number		Description
ch1	ch2	
D8063	D8438	Error code of serial communication error
D8150	D8155	Response wait time in inverter communication
D8151	D8156	Step number in inverter communication[*2]
D8152	D8157	Error code of inverter communication error[*1]
D8153	D8158	Latch of inverter communication error occurrence step[*2]
D8154	D8159	IVBWR instruction error parameter number[*2]

*1. Cleared when the PLC mode switches from STOP to RUN.

*2. Initial value: -1

Cautions

→ **For other cautions, refer to the Data Communication Edition manual.**

1) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
2) It is not permitted to use the RS and RS2 instructions and an inverter communication instruction (IVCK, IVDR, IVRD, IVWR and IVBWR) for the same port.
3) Two or more inverter communication instructions (IVCK, IVDR, IVRD, IVWR and IVBWR) can be driven for the same port at the same time.
4) Some restrictions to applicable devices
▲1: Applicable to the FX3U, FX3UC and FX3G PLCs only.
▲2: Applicable to the FX3U and FX3UC PLCs only.

PLC applicable version

The table below shows PLC versions applicable to each inverter.

PLC	FREQROL-V500/F500/A500/E500/S500	FREQROL-F700/A700	FREQROL-E700/D700
FX3G	Ver.1.10 or later		
FX3U	Ver.2.20 or later		Ver.2.32 or later
FX3UC	Ver.1.00 or later	Ver.2.20 or later	Ver.2.32 or later

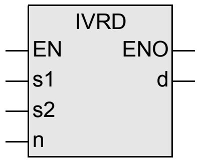
7.24.3 IVRD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	×	×	×	×	×	×

Outline

This instruction reads an inverter parameter to the PLC using the computer link operation function of the inverter.
This instruction corresponds to the EXTR (K12) instruction in the FX2N and FX2NC series PLCs.
→ **For detailed explanation of the instruction, refer to the Data Communication Edition manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
IVRD	16 bits	Continuous		IVRD(EN,s1,s2,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Inverter station number	ANY16
	s2	Inverter parameter number	ANY16
	n	Channel to be used (K1:ch1,K2:ch2*1)	ANY16
Output variable	ENO	Execution state	Bit
	d	Device storing the read value	ANY16

*1. "ch2" is not available for use for the FX3G PLCs of 14- and 24-point types.

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1	▲2			●	●	●			
s2														●	▲1	▲2			●	●	●			
d														●	▲1	▲2			●					
n																				●	●			

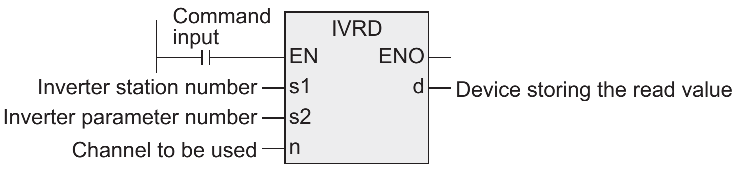
▲: Refer to "Cautions".

Function and operation explanation

→ For detailed explanation of the instruction, refer to the Data Communication Edition manual.

1. 16-bit operation (IVRD)

The value of the parameter in the device specified by s2 is read from an inverter*1 connected to a communication port n whose station number is in the device specified by s1, and output to the device specified by d.



*1. Mitsubishi Electric's FREQROL - F700, A700, E700, D700, V500, F500, A500, E500 and S500 series general purpose inverters

2. Related devices

→ For the instruction execution complete flag use method, refer to Section 1.3.4.

Number ch1	Number ch2	Description
M8029		Instruction execution complete
M8063	M8438	Serial communication error
M8151	M8156	Inverter communicating*1
M8152	M8157	Inverter communication error*1
M8153	M8158	Inverter communication error latch*1
M8154	M8159	IVBWR instruction error*1

Number ch1	Number ch2	Description
D8063	D8438	Error code of serial communication error
D8150	D8155	Response wait time in inverter communication
D8151	D8156	Step number in inverter communication*2
D8152	D8157	Error code of inverter communication error*1
D8153	D8158	Latch of inverter communication error occurrence step*2
D8154	D8159	IVBWR instruction error parameter number*2

*1. Cleared when the PLC mode switches from STOP to RUN.

*2. Initial value: -1

Cautions

→ For other cautions, refer to the Data Communication Edition manual.

1) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
2) It is not permitted to use the RS and RS2 instructions and an inverter communication instruction (IVCK, IVDR, IVRD, IVWR and IVBWR) for the same port.
3) Two or more inverter communication instructions (IVCK, IVDR, IVRD, IVWR and IVBWR) can be driven for the same port at the same time.
4) Some restrictions to applicable devices
▲1: Applicable to the FX3U, FX3UC and FX3G PLCs only.
▲2: Applicable to the FX3U and FX3UC PLCs only.

PLC applicable version

The table below shows PLC versions applicable to each inverter.

PLC	FREQROL-V500/F500/A500/E500/S500	FREQROL-F700/A700	FREQROL-E700/D700
FX3G	Ver.1.10 or later		
FX3U	Ver.2.20 or later		Ver.2.32 or later
FX3UC	Ver.1.00 or later	Ver.2.20 or later	Ver.2.32 or later

7.24.4 IVWR

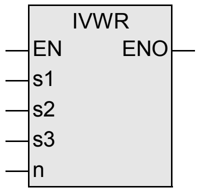
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	△	×	×	×	×	×	×

Outline

This instruction writes a parameter of an inverter using the computer link operation function of the inverter.
This instruction corresponds to the EXTR (K13) instruction in the FX2N and FX2NC series PLCs.

→ **For detailed explanation of the instruction, refer to the Data Communication Edition manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
IVWR	16 bits	Continuous		IVWR(EN,s1,s2,s3,n);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Inverter station number	ANY16
	s2	Inverter parameter number	ANY16
	s3	Set value to be written to the inverter parameter or device storing the data to be set	ANY16
	n	Channel to be used (K1:ch1,K2:ch2*1)	ANY16
Output variable	ENO	Execution state	Bit

*1. "ch2" is not available for use for the FX3G PLCs of 14- and 24-point types.

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R.	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	▲1	▲2			●	●	●			
s2														●	▲1	▲2			●	●	●			
s3														●	▲1	▲2			●	●	●			
n																				●	●			

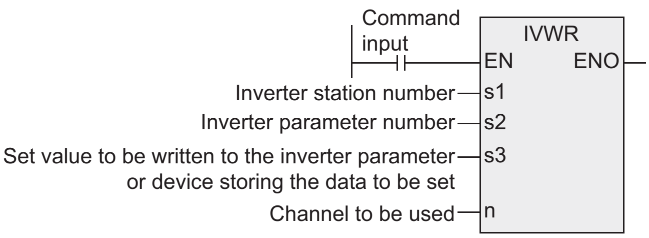
▲: Refer to "Cautions".

Function and operation explanation

→ For detailed explanation of the instruction, refer to the Data Communication Edition manual.

1. 16-bit operation (IVWR)

A value specified in the device specified by s3 is written to a parameter in the device specified by s2 in an inverter*1 connected to a communication port n whose station number is in the device specified by s1.



*1. Mitsubishi Electric's FREQROL - F700, A700, E700, D700, V500, F500, A500, E500 and S500 series general purpose inverters

2. Related devices

→ For the instruction execution complete flag use method, refer to Section 1.3.4.

Number ch1	Number ch2	Description
M8029		Instruction execution complete
M8063	M8438	Serial communication error
M8151	M8156	Inverter communicating*1
M8152	M8157	Inverter communication error*1
M8153	M8158	Inverter communication error latch*1
M8154	M8159	IVBWR instruction error*1

Number ch1	Number ch2	Description
D8063	D8438	Error code of serial communication error
D8150	D8155	Response wait time in inverter communication
D8151	D8156	Step number in inverter communication*2
D8152	D8157	Error code of inverter communication error*1
D8153	D8158	Latch of inverter communication error occurrence step*2
D8154	D8159	IVBWR instruction error parameter number*2

*1. Cleared when the PLC mode switches from STOP to RUN.

*2. Initial value: -1

Cautions

→ For other cautions, refer to the Data Communication Edition manual.

1) The instruction is provided in the FX3G PLC Ver. 1.10 or later.
2) It is not permitted to use the RS and RS2 instructions and an inverter communication instruction (IVCK, IVDR, IVRD, IVWR and IVBWR) for the same port.
3) Two or more inverter communication instructions (IVCK, IVDR, IVRD, IVWR and IVBWR) can be driven for the same port at the same time.
4) Some restrictions to applicable devices
▲1: Applicable to the FX3U, FX3UC and FX3G PLCs only.
▲2: Applicable to the FX3U and FX3UC PLCs only.

PLC applicable version

The table below shows PLC versions applicable to each inverter.

PLC	FREQROL-V500/F500/A500/E500/S500	FREQROL-F700/A700	FREQROL-E700/D700
FX3G	Ver.1.10 or later		
FX3U	Ver.2.20 or later		Ver.2.32 or later
FX3UC	Ver.1.00 or later	Ver.2.20 or later	Ver.2.32 or later

7.24.5 IVBWR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction writes parameters of an inverter at one time using the computer link operation function of the inverter.

→ **For detailed explanation of the instruction, refer to the Data Communication Edition manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
IVBWR	16 bits	Continuous	IVBWR EN ENO s1 s2 s3 n	IVBWR(EN,s1,s2,s3,n);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s1)	Inverter station number	ANY16
	(s2)	Number of parameters in an inverter to be written at one time.	ANY16
	(s3)	Head device of a parameter table to be written to an inverter	ANY16
	(n)	Channel to be used (K1:ch1,K2:ch2)	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

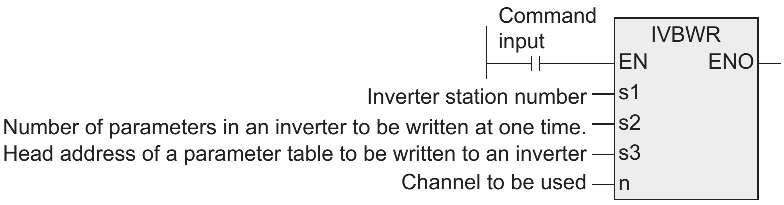
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)														●	●	●			●	●	●			
(s2)														●	●	●			●	●	●			
(s3)														●	●	●			●					
(n)																				●	●			

Function and operation explanation

→ For detailed explanation of the instruction, refer to the Data Communication Edition manual.

1. 16-bit operation (IVBWR)

A data table (parameter numbers and set values) specified in s2 and s3 is written to an inverter*1 connected to a communication port n whose station number is in the device specified by s1 all at once.



*1. Mitsubishi Electric's FREQROL - F700, A700, E700, D700, V500, F500, A500, E500 and S500 series general purpose inverters

*2. The table below shows the data table format.

s2 : Number of parameters to be written

s3 : Head device number of data table

Device	Parameter numbers to be written and set values	
s3	1st parameter	Parameter number
s3+1		Set value
s3+2	2nd parameter	Parameter number
s3+3		Set value
⋮	⋮	⋮
s3+2 s2-4	s2- 1"th parameter	Parameter number
s3+2 s2-3		Set value
s3+2 s2-2	s2 th parameter	Parameter number
s3+2 s2-1		Set value

2. Related devices

→ For the instruction execution complete flag use method, refer to Section 1.3.4.

Number ch1	Number ch2	Description
M8029		Instruction execution complete
M8063	M8438	Serial communication error
M8151	M8156	Inverter communicating*1
M8152	M8157	Inverter communication error*1
M8153	M8158	Inverter communication error latch*1
M8154	M8159	IVBWR instruction error*1

Number ch1	Number ch2	Description
D8063	D8438	Error code of serial communication error
D8150	D8155	Response wait time in inverter communication
D8151	D8156	Step number in inverter communication*2
D8152	D8157	Error code of inverter communication error*1
D8153	D8158	Latch of inverter communication error occurrence step*2
D8154	D8159	IVBWR instruction error parameter number*2

*1. Cleared when the PLC mode switches from STOP to RUN.

*2. Initial value: -1

Cautions

→ For other cautions, refer to the Data Communication Edition manual.

1) It is not permitted to use the RS and RS2 instructions and an inverter communication instruction (IVCK, IVDR, IVRD, IVWR and IVBWR) for the same port.

2) Two or more inverter communication instructions (IVCK, IVDR, IVRD, IVWR and IVBWR) can be driven for the same port at the same time.

Applicable models depending on the PLC version

Available inverter models are added depending on the version, as shown in the table below.

Applicable version		Item	Outline of function
FX3U	FX3UC		
Ver.2.32 or later	Ver.2.32 or later	Addition of applicable models	Mitsubishi FREQROL-E700 Series general-purpose inverters are supported.
Ver.2.20 or later	Ver.2.20 or later	Addition of applicable models	Mitsubishi FREQROL-F700/A700 Series general-purpose inverters are supported.

7.25 Data Transfer 3

7.25.1 RBFM

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction reads data from continuous buffer memories (BFM) in a special function block and unit over several operation cycles by the time division method. This instruction is convenient for reading received data, etc. stored in buffer memories in a special function block and unit for communication by the time division method. FROM instruction is also available to read the buffer memory (BFM) data.

→ For FROM instruction, refer to Section 7.8.9.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
RBFM	16 bits	Continuous	RBFM EN ENO m1 d m2 n1 n2	BFM(EN,m1,m2,n1,n2,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	m1	Unit number [0 to 7]	ANY16
	m2	Head buffer memory (BFM) number	ANY16
	n1	Number of all buffer memories (BFM) to be read [1 to 32767]	ANY16
	n2	Number of points transferred in one operation cycle [1 to 32767]	ANY16
Output variable	ENO	Execution state	Bit
	d	Head device storing data to be read from buffer memory (BFM)	ANY16

3. Applicable devices

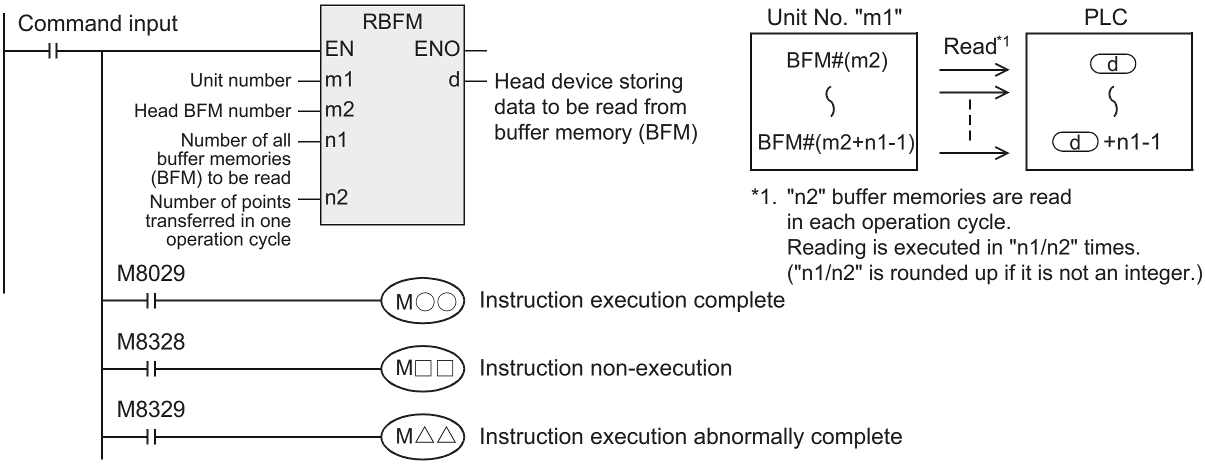
Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
m1														●	●					●	●			
m2														●	●					●	●			
d														▲1	●				●					
n1														●	●					●	●			
n2														●	●					●	●			

▲: Refer to "Cautions".

Function and operation explanation

1. 16-bit operation (RBFM)

"n1" buffer memory (BFM) units at location No. "m2" in special function block/unit No. "m1" are transferred (read) to the device specified by (d) in the PLC. While transferring, "n1" is divided by "n2" so n1/n2 buffer memories (rounded up when there is a remainder) are transferred per scan time.



1) When the instruction is finished normally, the instruction execution complete flag M8029 turns ON. When the instruction is finished abnormally, the instruction execution abnormally complete flag M8329 turns ON.
2) When RBFM or WBFM instruction in another step is executed for the same unit number, the instruction non-execution flag M8328 is set to ON, and execution of such an instruction is paused.
When execution of the other target instruction is complete, the paused instruction resumes.

Related devices

Device	Name	Description
M8029	Instruction execution complete	Turns ON when an instruction is finished normally.
M8328	Instruction non-execution	Turns ON when RBFM or WBFM instruction in another step is executed for the same unit number.
M8329	Instruction execution abnormally complete	Turns ON when an instruction is finished abnormally.

Related instructions

Instruction	Description
FROM	Read from a special function block
TO	Write to a special function block
WBFM	Divided BFM write

Cautions

1) The instruction is provided in the FX3UC PLC Ver. 2.20 or later.
2) Some restrictions to applicable devices
▲1: Except special data register D

Error

An operation error is caused in the following case. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the unit number "m1" does not exist. (Error code: K6708)

Common items between RBFM instruction and WBFM instruction

Specification of unit number of special function block and unit and buffer memory

→ **For the connection method of special extension units and blocks, number of connectable units and blocks, and handling of I/O numbers, refer to the manual of the PLC used and special function block and unit.**

1. Unit number "m1" of a special extension unit and block

Use the unit number to specify to which equipment the RBFM and/or WBFM instruction works.
Setting range: K0 to K7

Unit No.0 Built-in CC-Link/LT		Unit No.1	Unit No.2		Unit No.3
FX3UC-32MT-LT(-2) Main unit	I/O extension block	Special extension block	Special extension block	I/O extension block	Special extension block

A unit number is automatically assigned to each special extension unit and block connected to the PLC.
The unit number is assigned in the way "No. 0 → No. 1 → No. 2 ..." starting from the equipment nearest to the main unit.
Since the FX3UC-32MT-LT(-2) PLC has a built-in CC-Link/LT master, the unit number is given "No. 1 → No. 2 → No. 3 ..." from the equipment nearest to the main unit.

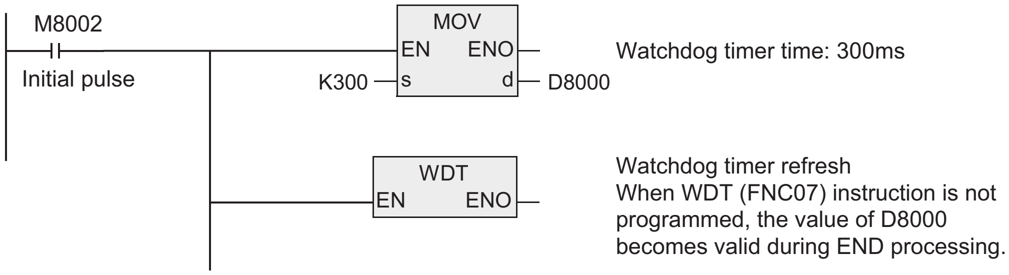
2. Buffer memory (BFM) number "m2"

Up to 32767 16-bit RAM memories are built in a special extension unit and block, and they are called buffer memories (BFM).
The buffer memory number is from "0" to "32766", and the contents are determined according to each special function unit and block.
Setting range: K0 to K32766

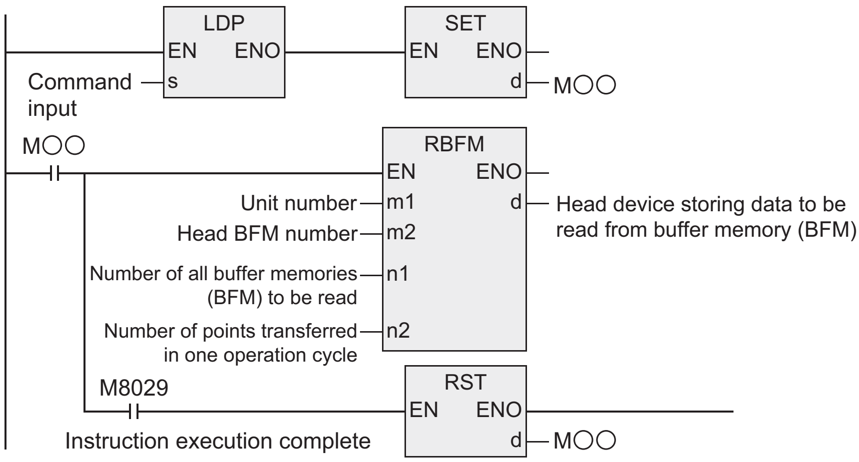
→ **For the contents of buffer memories, refer to the manual of the special function block and unit used.**

Cautions

1) A watchdog timer error may occur when many numbers of points are transferred in one operation cycle. In such a case, take either of the following countermeasures.
   a) Change the watchdog timer time
      By overwriting the contents of D8000 (watchdog timer time), the watchdog timer detection time is changed (initial value: K200).
      When the program shown below is input, the sequence program will be monitored with the new watchdog timer time.



   b) Change the number of transferred points "n2" in each operation cycle.
      Change the number of transferred points "n2" in each operation cycle to a smaller value.
2) Do not stop the driving of the instruction while it is being executed. If driving is stopped, the buffer memory (BFM) reading/writing processing is suspended, but the data acquired in the middle of reading/writing processing is stored in the device specified by (d) and later and buffer memories (BFM).



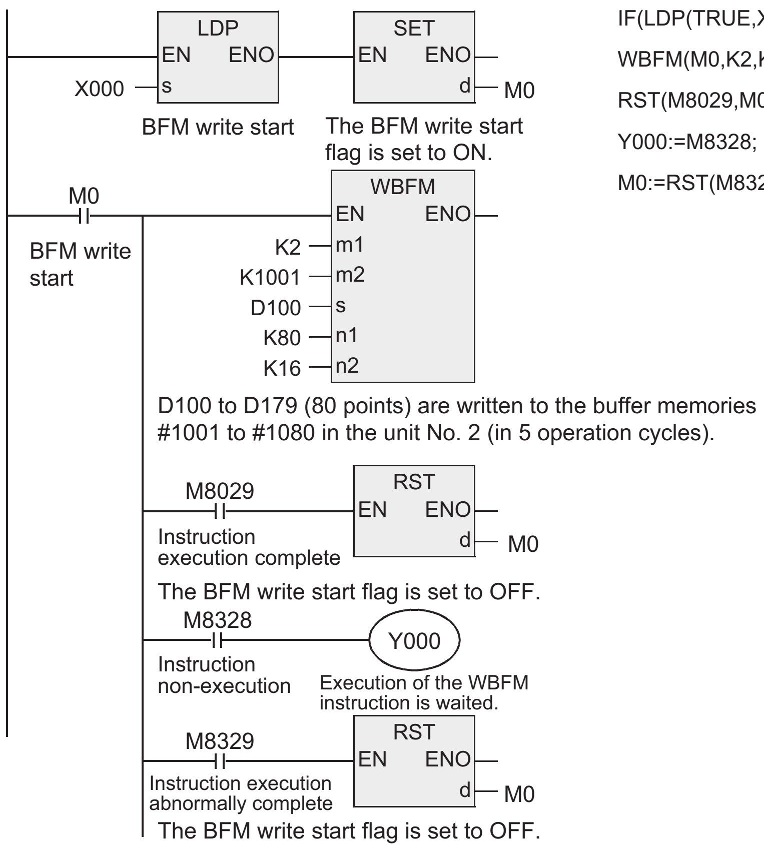
3) When indexing is executed, the contents of index registers at the beginning of execution are used.
   Even if the contents of index registers are changed after the instruction is executed, such changes do not affect the process of the instruction.
4) The contents of "n1" devices starting from the device specified by (d) change while RBFM instruction is executed. After execution of the instruction is completed, execute another instruction for "n1" devices starting from the device specified by (d).
5) Do not update (change) the contents of "n1" devices starting from the device specified by (s) while WBFM instruction is executed. If the contents are updated, the intended data may not be written to the buffer memories (BFM).
6) Do not update (change) the contents of "n1" buffer memories (BFM) starting from the buffer memory NO. "m2" while RBFM instruction is executed. If the contents are updated, the intended data may not be read.
7) The FX3UC PLCs of V 2.20 or later support the RBFM instruction.

Program example

In the program example shown below, data is read from and written to the buffer memories (BFM) in the unit No. 2 as follows:

1) When X000 is set to ON, data stored in D100 to D179 (80 points) are written to the buffer memories (BFM) #1001 to #1080 in the special function block and unit whose unit number is No. 2 by 16 points in each operation cycle.

[Structured ladder]

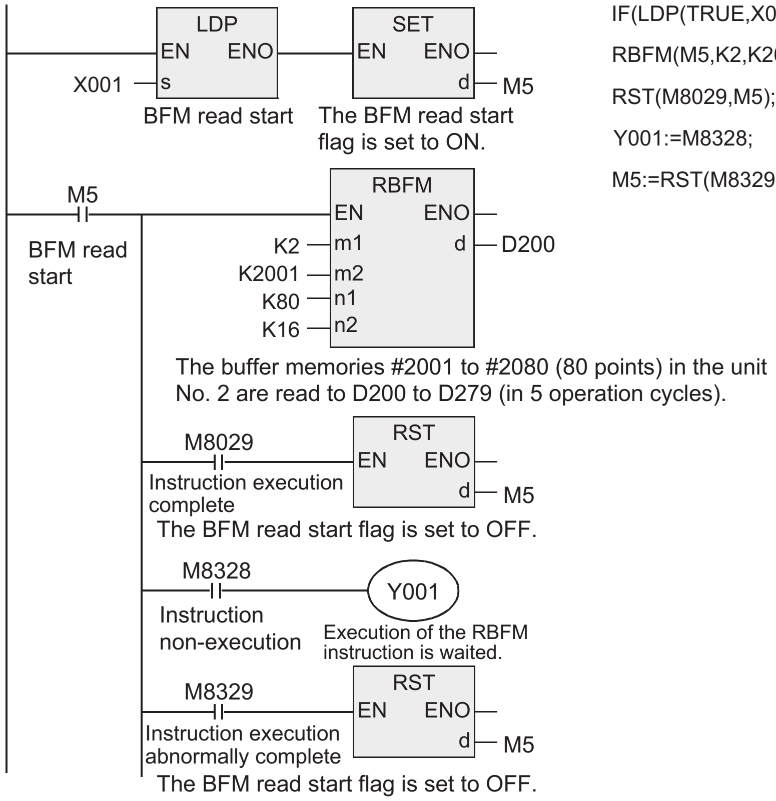


[ST]

```
IF(LDP(TRUE,X000) THEN SET(TRUE,M0));
WBFM(M0,K2,K1001,D100,K80,K16);
RST(M8029,M0);
Y000:=M8328;
M0:=RST(M8329);
```

2) When X001 is set to ON, the buffer memories (BFM) #2001 to #2080 (80 points) in the special function block and unit whose unit number is No. 2 are written to D200 to D279 by 16 points in each operation cycle.

[Structured ladder]



[ST]

```
IF(LDP(TRUE,X000) THEN SET(TRUE,M5));
RBFM(M5,K2,K2001,K80,K16,D200);
RST(M8029,M5);
Y001:=M8328;
M5:=RST(M8329);
```

7.25.2 WBFM

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction writes data to continuous buffer memories (BFM) in a special function block and unit over several operation cycles by the time division method. This instruction is convenient for writing send data, etc. to buffer memories in a special function block and unit for communication by the time division method.
TO instruction is also available for writing data to the buffer memory (BFM).

→ **For TO instruction, refer to Section 7.8.10.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
WBFM	16 bits	Continuous	WBFM EN ENO m1 m2 s n1 n2	WBFM(EN,m1,m2,s,n1,n2);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	m1	Unit number	ANY16
	m2	Head buffer memory (BFM) number	ANY16
	s	Head device storing data to be written to buffer memory (BFM)	ANY16
	n1	Number of all buffer memories (BFM) to be written	ANY16
	n2	Number of points transferred in one operation cycle	ANY16
Output variable	ENO	Execution state	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
m1														●	●					●	●			
m2														●	●					●	●			
s														▲1	●				●					
n1														●	●					●	●			
n2														●	●					●	●			

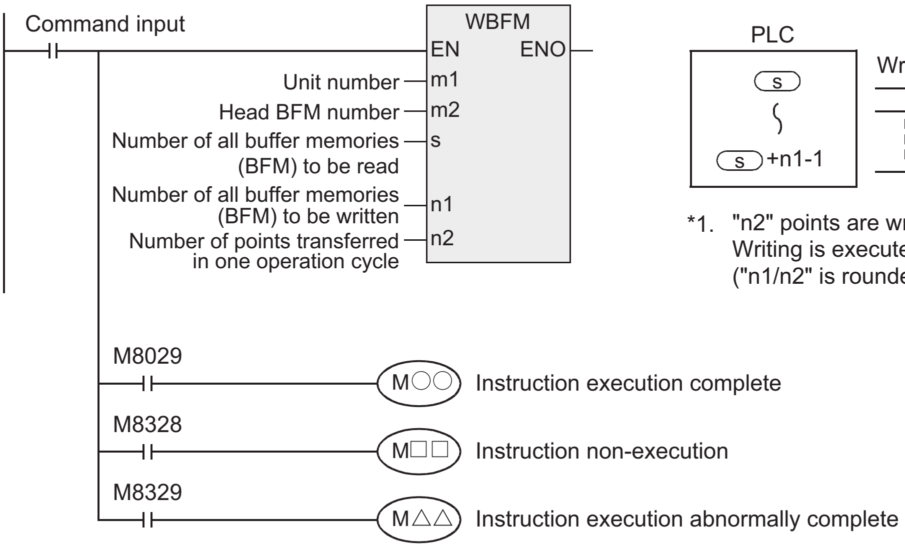
▲: Refer to "Cautions".

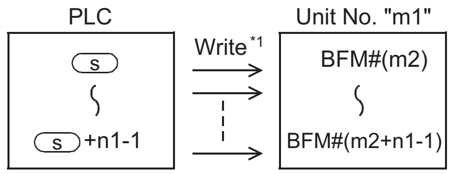
Function and operation explanation

1. 16-bit operation (WBFM)

"n1" word units from the device specified by (s) in the PLC are transferred (written) to buffer memory (BFM) location No. "m2" in special function unit and block No. "m1". While transferring, "n1" is divided by "n2" so n1/ n2 words (rounded up when there is a remainder) are transferred per scan time.

→ **For the unit number, buffer memory (BFM) number, cautions and program example, refer to Section 7.25.1.**





*1. "n2" points are written in each operation cycle. Writing is executed in "n1/n2" times. ("n1/n2" is rounded up if it is not an integer.)

1) When the instruction is finished normally, the instruction execution complete flag M8029 turns ON. When the instruction is finished abnormally, the instruction execution abnormally complete flag M8329 turns ON.
2) When RBFM or WBFM instruction in another step is executed for the same unit number, the instruction non-execution flag M8328 is set to ON, and execution of such an instruction is paused.
When execution of the other target instruction is complete, the paused instruction resumes.

Related devices

Device	Name	Description
M8029	Instruction execution complete	Turns ON when an instruction is finished normally.
M8328	Instruction non-execution	Turns ON when RBFM or WBFM instruction in another step is executed for the same unit number.
M8329	Instruction execution abnormally complete	Turns ON when an instruction is finished abnormally.

Related instructions

Instruction	Error
FROM	Read from a special function block
TO	Write to a special function block
RBFM	Divided BFM write

Cautions

1) The instruction is provided in the FX3UC PLC Ver. 2.20 or later.
2) Some restrictions to applicable devices
▲1: Except special data register D

Error

An operation error is caused in the following case. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the unit number "m1" does not exist. (Error code: K6708)

7.26 High Speed Processing 2

7.26.1 DHSCT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction compares the current value of a high speed counter with a data table of comparison points, and then sets or resets up to 16 output devices.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
DHSCT	32 bits	Continuous	DHSCT EN ENO s1 d m s2 n	DHSCT(EN,s1,m,s2,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	Head device storing the data table	ANY32
	m	Number of comparison points in data table (1 ≤ m ≤ 128)	ANY32
	s2	High speed counter (C235 to C255)	ANY32
	n	Number of devices to which the operation status is output	ANY32
Output variable	ENO	Execution state	Bit
	d	Head device to which the operation status is output	Bit

3. Applicable devices

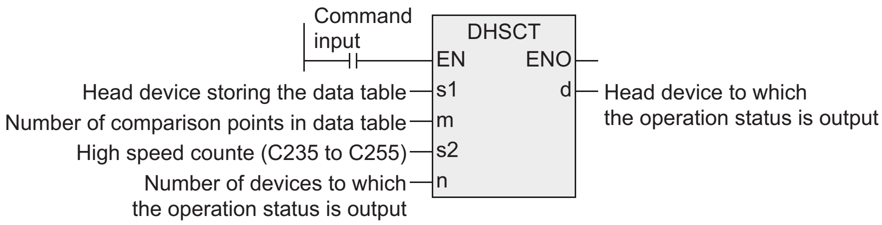
Operand type	Bit Devices							Word Devices												Others					
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	●				●					
m																				●	●			
s2													▲1						●					
d		●	●			●													●					
n																				●	●			

▲: Refer to "Cautions".

Function and operation explanation

1. 32-bit operation (DHSCT)

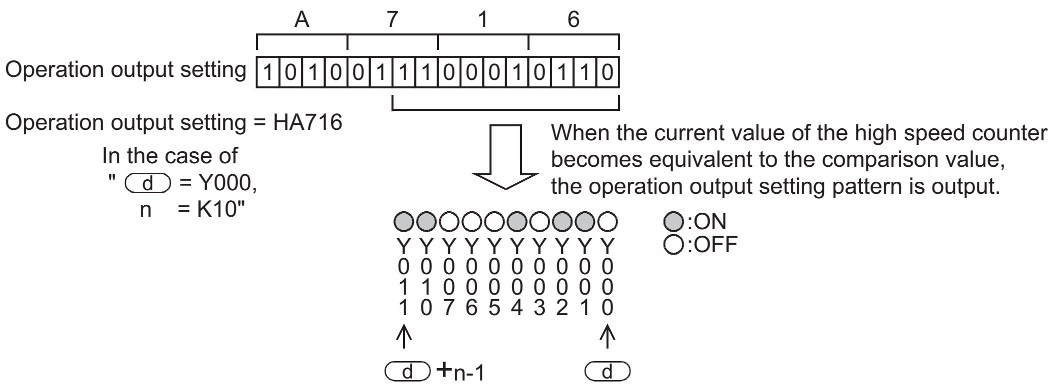
The current value of a high speed counter specified in (s2) is compared with the data table shown below which has "m" points stored in the device specified by (s1) and later, and the operation output set value (ON or OFF) specified in the data table is output to the devices specified by (d).



Data table used for comparison

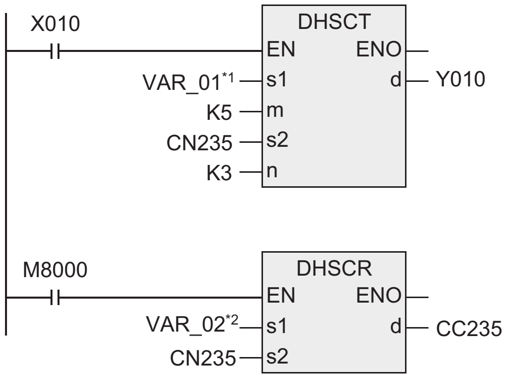
Comparison point number	Comparison data	Operation output set value (SET [1] or RESET [0])	Operation output destination
0	(s1)+1, (s1)	(s1)+2	(d) to (d)+n-1
1	(s1)+4, (s1)+3	(s1)+5	
2	(s1)+7, (s1)+6	(s1)+8	
⋮	⋮	⋮	
m-2	(s1)+3m-5, (s1)+3m-6	(s1)+3m-4	
m-1	(s1)+3m-2, (s1)+3m-3	(s1)+3m-1	

Operation output set value (SET [1] or RESET [0]) [Up to 16 points]



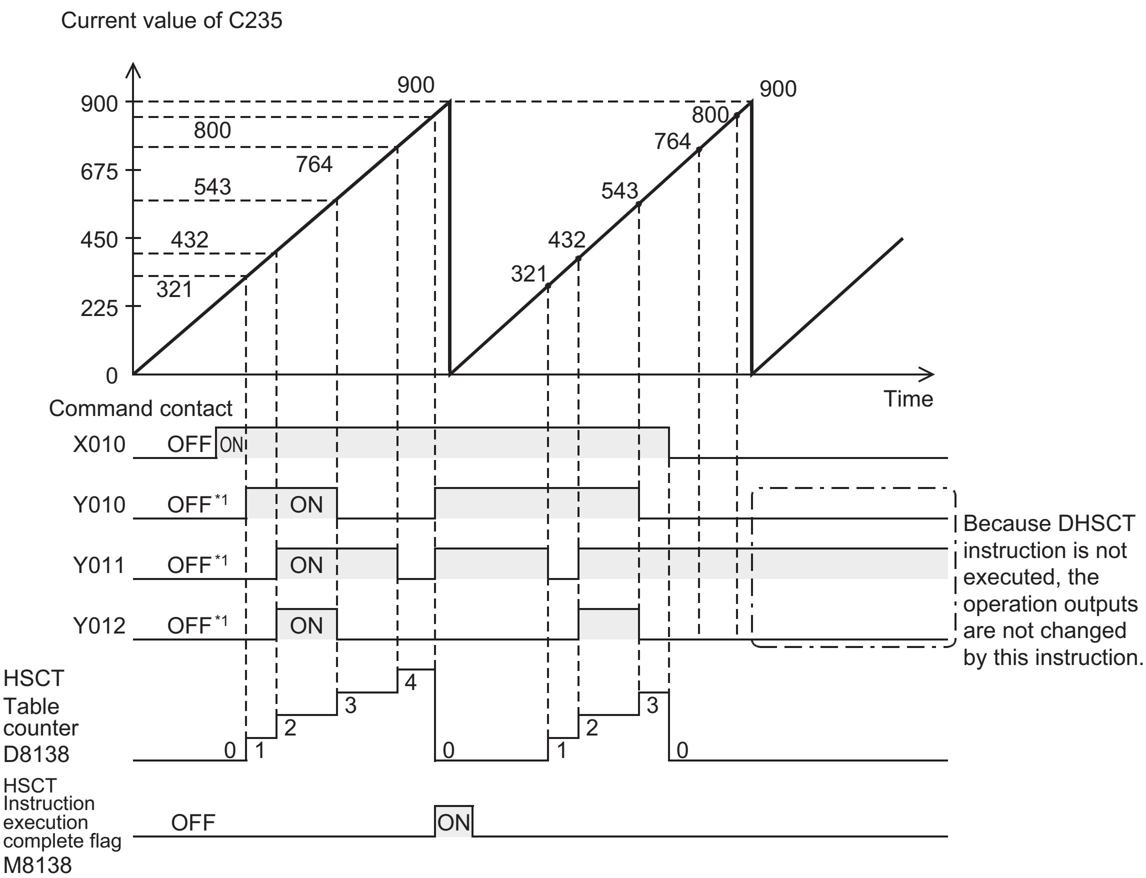
1) When this instruction is executed, the uppermost data table of the data tables is set as the comparison target.
2) When the current value of the high speed counter, specified in (s2), becomes equivalent to the comparison value in the data table, the corresponding operation output specified in the data table is output to the device specified by (d).
   If an output (Y) is specified in (d), the output processing is executed immediately without waiting for the output refresh executed by the END instruction.
   When specifying and output (Y), make sure that the least significant digit of the device number is "0".
   Examples: Y000, Y010 and Y020
3) Immediately after step 2), "1" is added to the current table counter value D8138.
4) The next comparison point is set as the comparison target data.
5) Steps 2) and 3) are repeated until the current value of the table counter D8138 becomes "m".
   When the current value becomes "m", the instruction execution complete flag M8138 turns ON, and the execution returns to step 1). At this time, the table counter D8138 is reset to "0".
6) When the command contact is set to OFF, execution of the instruction is stopped and the table counter D8138 is reset to "0".

Operation example



*1. VAR_01 is a global label and is defined as D200.
*2. VAR_02 is a global label and is defined as K900.

Comparison point number	Comparison data		SET/RESET pattern		Table counter (D8138)
	Device	Comparison value	Device	Operation output set value	
0	D201, D200	K321	D202	H0001	0↓
1	D204, D203	K432	D205	H0007	1↓
2	D207, D206	K543	D208	H0002	2↓
3	D210, D209	K764	D211	H0000	3↓
4	D213, D212	K800	D214	H0003	4↓ (repeated from "0")



*1. If this instruction is not executed, no processing is executed for outputs.
In the operation example shown above, the command contact is "OFF".

2. Related devices

Device	Name	Description
M8138	DHSCT Instruction execution complete flag	Turns ON when the operation for the final table No. "m-1" is completed.
D8138	DHSCT Table counter	Stores the comparison point number handled as the comparison target.

Cautions

1) This instruction can be executed only once in a program.
If this instruction is programmed two or more times, an operation error is caused by the second instruction and later, and the instruction will not be executed.
2) This instruction constructs the data table at the END instruction after the first execution of the instruction. Accordingly, the operation output works after the second scan and later.
3) With regard to DHSCT, DHSCS, DHSCR and DHSZ instructions, up to 32 instructions can be executed in one operation cycle. An operation error is caused by the 33rd instruction and later, and the instruction will not be executed.
4) If an output (Y) is specified in (d), the output processing is executed immediately without waiting for the output refresh executed by END instruction.
When specifying an output (Y), make sure that the least significant digit of the device number is "0".
Examples: Y000, Y010 and Y020
5) When a high speed counter specified in (s2) is indexed with index, all high speed counters are handled as software counters.
6) For this instruction, only one comparison point (one line) is handled as the comparison target at one time. Processing will not move to the next comparison point until the current counter value becomes equivalent to the comparison point currently selected as the comparison target.
If the current value of a high speed counter executes up counting using the comparison data table shown in the operation example on the previous page, be sure to execute the instruction while the current value of the high speed counter is smaller than the comparison value in comparison point No. 1.
7) When handling 32-bit data in a structured program, a 16-bit device cannot be specified directly as in the case of a simple project. Use a label to handle 32-bit data.
A 32-bit counter can be specified directly as it is a 32-bit long device.
Use a global label to specify a device.
8) When this instruction operates in the FX3U and FX3UC PLCs, the hardware counters (C235, C236, C237, C238, C239, C240, C244(OP), C245(OP), C246, C248(OP), C251 and C253) switches automatically to software counters, and the maximum frequency and total frequency are affected.
9) Some restrictions to applicable devices
▲1: Only high speed counters C235 to C255 are available for use.

Error

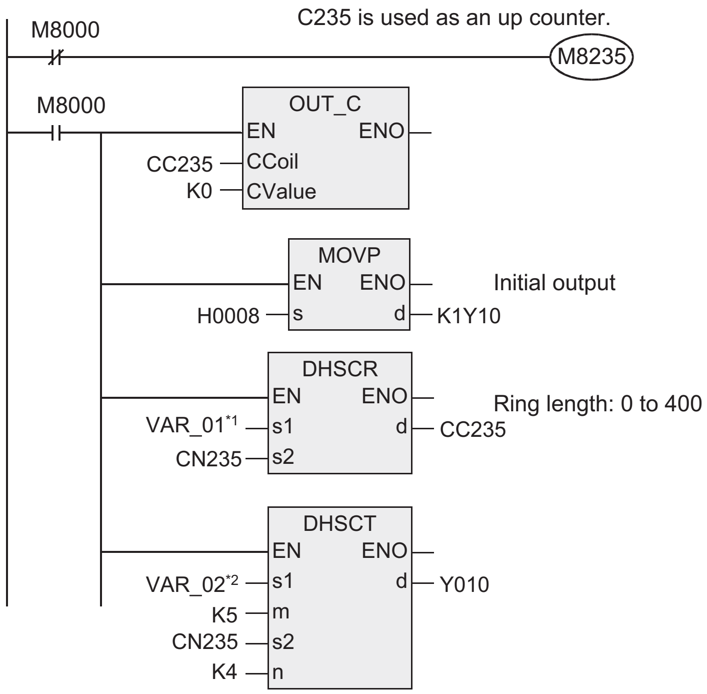
An operation error occurs in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When any devices other than high speed counters C235 to C255 are specified in (s2).
(Error code: K6706)
2) When the "3m-1"th device from a device specified in (s1) exceeds the last number of the device.
(Error code: K6706)
3) When the "n"th device from a device specified in (d) exceeds the last number of the device.
(Error code: K6706)
4) When this instruction is used two or more times in a program. (Error code: K6705)
5) With regard to DHSCT, DHSCS, DHSCR and DHSZ instructions, up to 32 instructions can be executed in one operation cycle. An operation error is caused by the 33rd instruction and later, and the instruction will not be executed. (Error code: K6705)

Program example

In the program example shown below, the current value of C235 (counting X000) is compared with the comparison data table set in R0 and later, and a specified pattern is output to Y010 to Y013.

[Structured ladder]



*1. VAR_01 is a global label and is defined as K400.

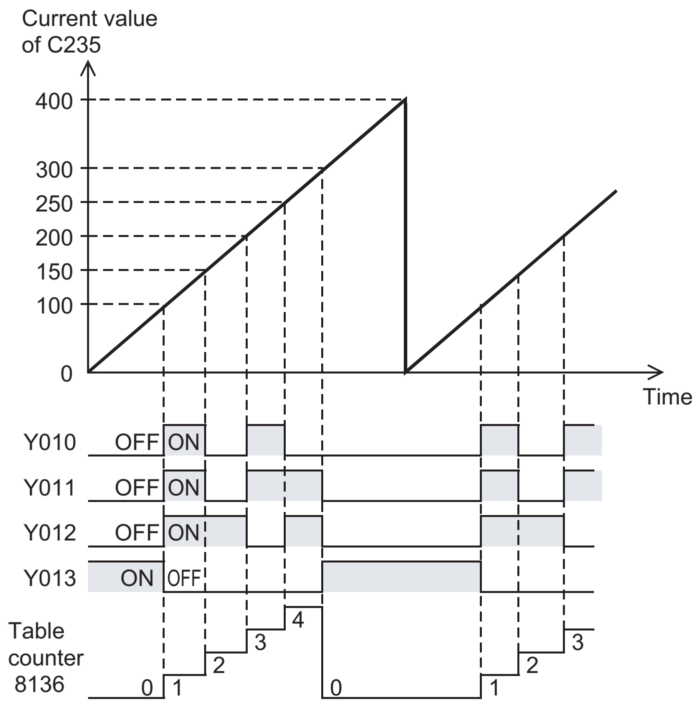
*2. VAR_02 is a global label and is defined as C235.

[ST]

```
M8235:=M8000;
OUT_C(M8000,CC235,K0);
MOVP(M8000,H0008,K1Y10);
DHSCR(VAR_01,C235,C235);
Y010:=DHSCT(M8000,VAR_02,K5,C235,K4);
```

Operation example

Comparison point number	Comparison data		SET/RESET pattern		Table counter (D8138)
	Device	Comparison value	Device	Operation output set value	
0	R1, R0	K100	R2	H0007	0↓
1	R4, R3	K150	R5	H0004	1↓
2	R7, R6	K200	R8	H0003	2↓
3	R10, R9	K250	R11	H0006	3↓
4	R13, R12	K300	R14	H0008	4 ↓ (repeated from "0")


Current value
of C235
400
300
250
200
150
100
0
Time
Y010
Y011
Y012
Y013
OFF
ON
Table
counter
8136
0
1
2
3
4

7.27 Extension File Register Control

7.27.1 LOADR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	○	×	×	×	×	×	×

Outline

This instruction reads the current values of extension file registers (ER) stored in a memory cassette (flash memory and EEPROM) or the file registers (ER) in the PLC's built-in EEPROM, and transfers them to extension registers (R) stored in the PLC's built-in RAM.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
LOADR	16 bits	Continuous	LOADR block: EN, s, n inputs; ENO output	LOADR(EN,s,n);
LOADRP	16 bits	Pulse	LOADRP block: EN, s, n inputs; ENO output	LOADRP(EN,s,n);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Device of extension register (transfer destination) to which data is to be transferred. (The extension file register having the same number is handled as the transfer source.)	ANY16
	n	Number of points to be read (transferred) (FX3G: 1 ≤ n ≤ 24000, FX3U•FX3UC: 0 ≤ n ≤ 32767)	ANY16
Output variable	ENO	Execution state	Bit

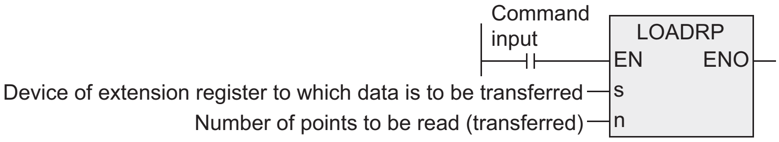
3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit specification				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s															●				●					
n														●						●	●			

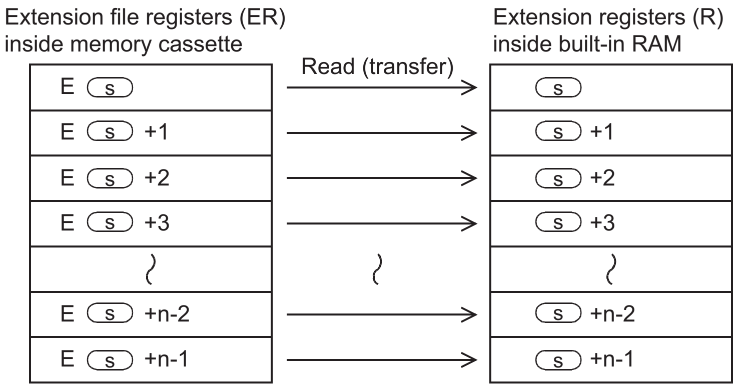
1 Outline
2 Instruction List
3 Configuration of Instruction
4 How to Read Explanation of Instructions
5 Basic Instruction
6 Step Ladder Instructions
7 Applied Instructions
8 Interrupt Function and Pulse Catch Function
A Relationships between devices and addresses

Function and operation explanation

1. 16-bit operation (LOADR/LOADRP)

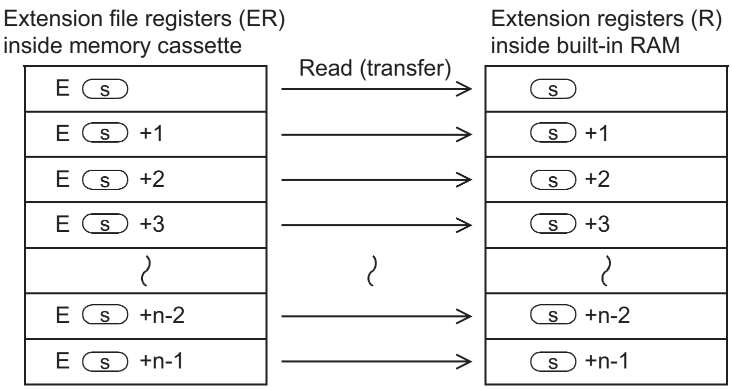


1) For the FX3U and FX3UC PLCs
   The contents (current values) of extension file registers (ER) stored in a memory cassette (flash memory) having the same numbers with the extension registers specified by (s) to (s)+n-1 are read, and transferred to the extension registers (R) specified by (s) to (s)+n-1 stored in the PLC's built-in RAM.



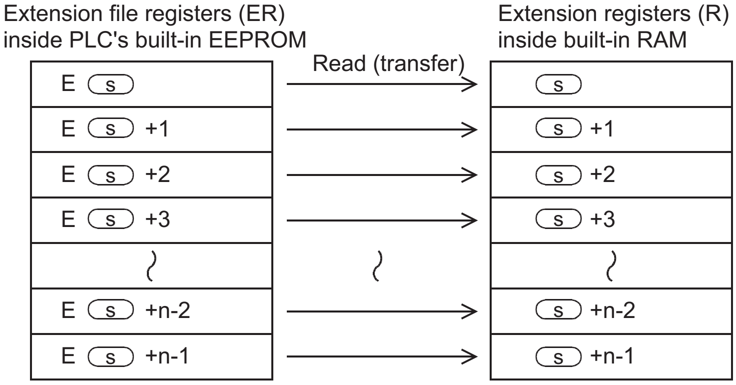
- Reading and transfer are executed in units of device. Up to 32768 devices can be read and transferred.
- Different from SAVER, INITR and LOGR instructions, it is not necessary to execute this instruction in units of sector.
- If "n" is set to "0", it is handled as "32768" when the instruction is executed.

2) For the FX3G PLCs
   a) When connecting to a memory cassette
      The contents (current values) of extension file registers (ER) stored in a memory cassette (EEPROM) having the same numbers with the extension registers (R) specified by (s) to (s)+n-1 are read, and transferred to the extension registers specified by (s) to (s)+n-1 stored in the PLC's built-in RAM.



- Reading and transfer are executed in units of device. Up to 24000 devices can be read and transferred.

b) When not connecting to a memory cassette
The contents (current values) of extension file registers (ER) stored in PLC's built-in EEPROM having the same numbers with the extension registers specified by (S) to (S)+n-1 are read, and transferred to the extension registers (R) specified by (S) to (S)+n-1 stored in the PLC's built-in RAM.



- Reading and transfer are executed in units of device. Up to 24000 devices can be read and transferred.

Cautions

1. About the allowable number of times of writing operations in memory

Note the following when accessing the extension file registers:

- For the FX3U and FX3UC PLCs
The memory cassette (flash memory) allows up to 10,000 times of writing operations.
The number of times of writing operations counts up each time the INITR, RWER or INITER instruction is executed. Do not exceed the allowable number of times of writing operations.
When a continuous operation type instruction is executed, writing operation to the memory occurs for each operation cycle of the PLC. To avoid this, be sure to use pulse operation type instructions.
The number of times of writing operations does not count up when the LOADR, SAVER or LOGR instruction is executed. However, the SAVER and LOGR instructions require the target write sectors to be initialized before executing the instructions. Note that, when initializing by using the INITR or INITER instruction, the number of times of writing operations in the memory counts up every time the NITR or INITER instruction is executed.

- For the FX3G PLCs
The memory cassette (EEPROM) and PLC's built-in memory (EEPROM) allow up to 10,000 times and 20,000 times of writing operations, respectively.
The number of times of writing operations counts up each time the RWER instruction is executed. Do not exceed the allowable number of times of writing operations.
When a continuous operation type instruction is executed, writing operation to the memory occurs for each operation cycle of the PLC. To avoid this, be sure to use pulse operation type instructions.
The number of times of writing operations does not count up when the LOADR instruction is executed.

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

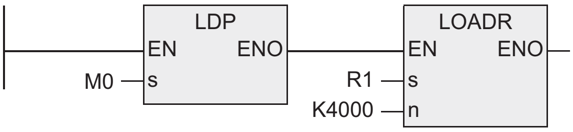
1) When the last device number to be transferred exceeds "32767" (error code: K6706).
   At this time, devices up to the last one (R32767) are read and transferred.
2) When a memory cassette is not connected. (Error code: K6771)*1

*1. This does not cause an error with the FX3G PLCs because the PLCs read the contents of the extension file registers stored in the PLC's built-in EEPROM when a memory cassette is not connected.

Program example

In the program example shown below, the contents (current values) of 4000 extension file registers ER1 to ER4000 inside the memory cassette are read, and transferred to 4000 extension registers R1 to R4000 inside the built-in RAM.

[Structured ladder]



[ST]

```
IF(LDP(TRUE,M0)THEN  LOADR(TRUE,R1,K4000));
```

Extension file registers (ER) inside memory cassette → Read (transfer) → Extension registers (R) inside built-in RAM

Device number	Current value		Device number	Current value
ER1	K100	→	R1	K100
ER2	K50	→	R2	K50
ER3	H0003	→	R3	H0003
ER4	H0101	→	R4	H0101
⁓	⁓	⁓	⁓	⁓
ER3999	K55	→	R3999	K55
ER4000	K59	→	R4000	K59

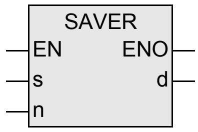
7.27.2 SAVER

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction writes the current values of extension registers (R) stored in the PLC's built-in RAM to extension file registers (ER) stored in a memory cassette (flash memory) in units of sector (2048 points).
RWER instruction provided in FX3UC PLCs Ver. 1.30 or later and FX3U PLCs writes (transfers) only arbitrary number of points. It is not necessary to execute INITR or INITER instruction every time when RWER instruction is used.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
SAVER	16 bits	Continuous		SAVER(EN,s,n,d);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Device of extension register to which data is to be written (Only the head device of a sector of extension registers can be specified.)	ANY16
	(n)	Number of points written (transferred) in one operation cycle.	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Device storing the number of already written points	ANY16

3. Applicable devices

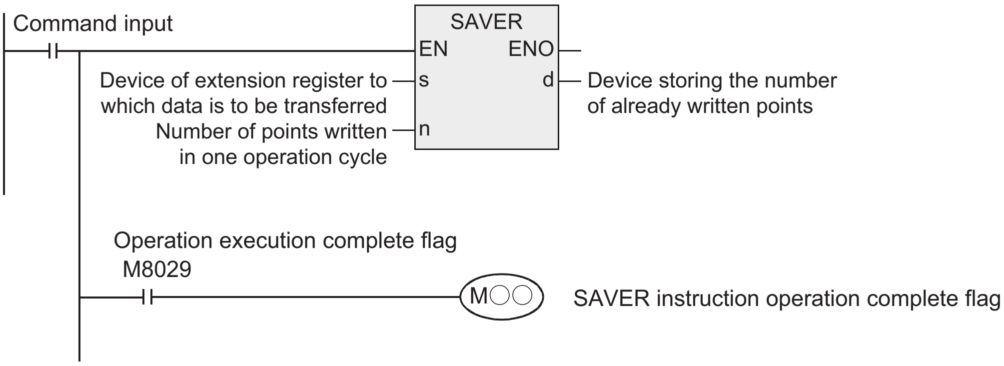
Operand type	Bit Devices: System user							Word Devices: Digit specification				Word Devices: System user				Word Devices: Special unit	Word Devices: Index			Others: Constant		Others: Real Number	Others: Character String	Others: Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)															●				●					
(n)																				●	●			
(d)														●					●					

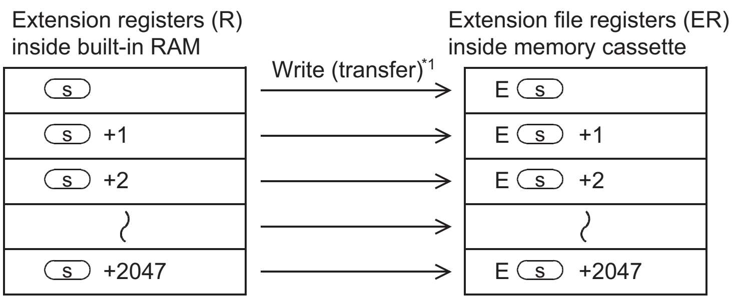
Function and operation explanation

1. 16-bit operation (SAVER)

The contents (current values) of 2,048 extension registers (R) starting from the device specified by (s) are written (transferred) to extension file registers (ER) inside a memory cassette (flash memory) having the same device numbers in "2048/n" operation cycles ("2048/n +1" cycles if there is the remainder).

When the instruction is being executed, the number of already written points is stored in (d).





*1. "n" points are written (transferred) in each operation cycle.
(Number of points specified by (n))

1) Extension file registers are written in units of sector (2048 points).
   The table below shows the head device number in each sector.

Sector number	Head device number	Written device range	Sector number	Head device number	Written device range
Sector 0	R0	ER0 to ER2047	Sector 8	R16384	ER16384 to ER18431
Sector 1	R2048	ER2048 to ER4095	Sector 9	R18432	ER18432 to ER20479
Sector 2	R4096	ER4096 to ER6143	Sector 10	R20480	ER20480 to ER22527
Sector 3	R6144	ER6144 to ER8191	Sector 11	R22528	ER22528 to ER24575
Sector 4	R8192	ER8192 to ER10239	Sector 12	R24576	ER24576 to ER26623
Sector 5	R10240	ER10240 to ER12287	Sector 13	R26624	ER26624 to ER28671
Sector 6	R12288	ER12288 to ER14335	Sector 14	R28672	ER28672 to ER30719
Sector 7	R14336	ER14336 to ER16383	Sector 15	R30720	ER30720 to ER32767

2) If "n" is set to "0", it is handled as "2048" when the instruction is executed.
3) When writing (transfer) of 2048 points is finished, execution of the instruction is completed and the instruction execution complete flag M8029 turns ON.
4) The number of already written points is stored in the device specified by (d).

2. Related devices

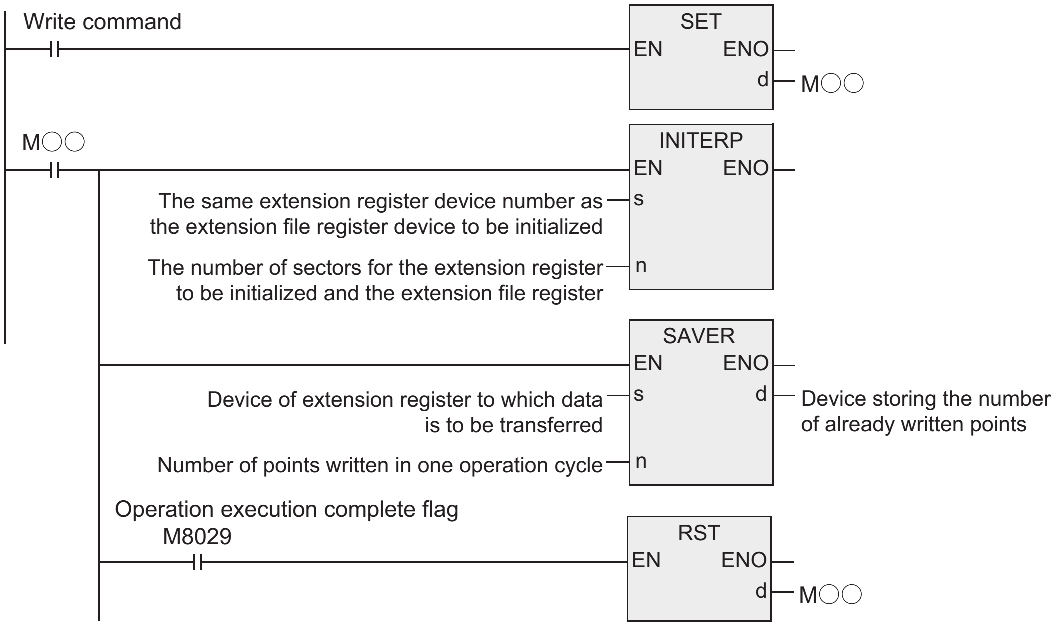
Device number	Name	Description
M8029	Instruction execution complete	When execution of the target instruction is completed, the instruction execution complete flag M8028 turns ON. In a program, however, there may be two or more instructions which can use the flag M8029. To avoid confusion, be sure to use the NO (normally open) contact of this flag immediately under SAVER instruction so that this flag works only for SAVER instruction

Cautions

1. Cautions on writing data to a memory cassette

Memory cassettes adopt flash memory. Note the following contents when writing data to extension file registers in a memory cassettes with the SAVER instruction.

1) It takes about 340 ms to write all points (2048 points). If "n" is set to K0 or K2048, the operation cycle for executing this instruction becomes about 340 ms longer than normal.
   If the operation cycle is severely affected, write data in two or more operation cycles.
   When writing data in two or more operation cycles, set "n" in the range from K1 to K1024.
2) Do not abort execution of this instruction in the middle of operation. If execution is aborted, unexpected data may be written to extension file registers.
   If execution of this instruction is aborted by turning OFF the power, execute the instruction again using step [2] described on the next page after turning ON the power again.



3) Execute INITER or INITR instruction to target extension file registers (ER) before executing SAVER instruction. If SAVER instruction is driven before INITER or INITR instruction is executed, an operation error (error code: K6770) may be caused.
   To avoid such an operation error, make a program for executing SAVER instruction in the following sequence

When the FX3U and FX3UC PLCs are Ver. 1.30 or later

[1] When storing data of 2048 extension registers (R) in one sector to extension file register (ER).

a) Execute INITER instruction to extension file registers (ER) specified as targets in SAVER instruction.
b) Execute SAVER instruction.

[2] When storing the contents of an arbitrary number of extension registers (R) to extension file registers (ER)

Use RWER instruction.

→ **For RWER instruction, refer to Section 7.27.5.**

When the FX3UC PLCs are former than Ver. 1.30.

[1] When storing data of 2048 extension registers (R) in one sector to extension file registers (ER)

If the extension registers (R) have data to be stored in extension file registers (ER), use the procedure [2].

a) Execute INITR instruction to extension registers (R) and extension file registers (ER) specified as targets in SAVER instruction.
b) Store data to extension registers (R) specified as targets.
c) Execute SAVER instruction.

[2] When storing data of 2048 extension registers (R) in one sector to extension file registers (ER)

a) Temporarily withdraw the data of extension registers (R) specified as targets in SAVER instruction to data registers or unused 2048 extension registers (R) by using BMOV instruction.

b) Execute INITR instruction to extension registers (R) and extension file registers (ER) specified as targets in SAVER instruction.

c) Return the data of 2048 points temporarily withdrawn in step a) to extension registers (R) specified as targets by using BMOV instruction.

d) Execute SAVER instruction.

2. About the allowable number of times of writing operations in memory

Note the following when accessing the extension file registers:

- The memory cassette (flash memory) allows up to 10,000 times of writing operations.
  The number of times of writing operations counts up each time the INITR, RWER or INITER instruction is executed. Do not let the number of times of writing operations exceed the allowable number of times of writing operations.
  When a continuous operation type instruction is executed, writing operation to the memory occurs for each operation cycle of the PLC. To avoid this, be sure to use pulse operation type instructions.
- The number of times of writing operations does not count up when the LOADR, SAVER or LOGR instruction is executed. However, the SAVER and LOGR instructions require the target write sectors to be initialized before executing the instructions. Note that, when initializing by using the INITR or INITER instruction, the number of times of writing operations in the memory counts up every time the NITR or INITER instruction is executed.

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

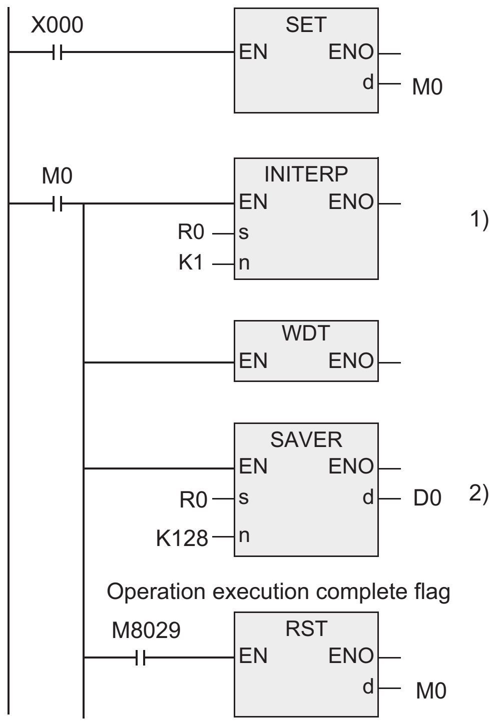
1) When any device number other than the head device number of a sector of extension file registers is set to (s).
   (Error code: K6706)
2) When a memory cassette is not connected. (Error code: K6771)
3) When the protect switch of the memory cassette is set to ON. (Error code: K6770)
4) When the collation result after data writing is "mismatch" due to omission of initialization or for another reason. (Error code: K6770)

Program example

1) In the case of FX3UC PLCs Ver. 1.30 or later and FX3U PLCs Ver. 2.20 or later
In the program example shown below, the changed contents of extension registers R10 to R19 (sector 0) used for setting data are reflected on the extension file registers (ER) when X000 is set to ON. (128 points are written in one operation cycle.)

Program

[Structured ladder]



[ST]

```
SET(X000,M0);
INTERP(M0,R0,K1);
WDT(M0);
SAVER(M0,R0,K128,D0);
RST(M8029,M0);
```

Operation example

Setting data (A): R0 to R99; Changed data: R10 to R19; Unused: R100 to R2047

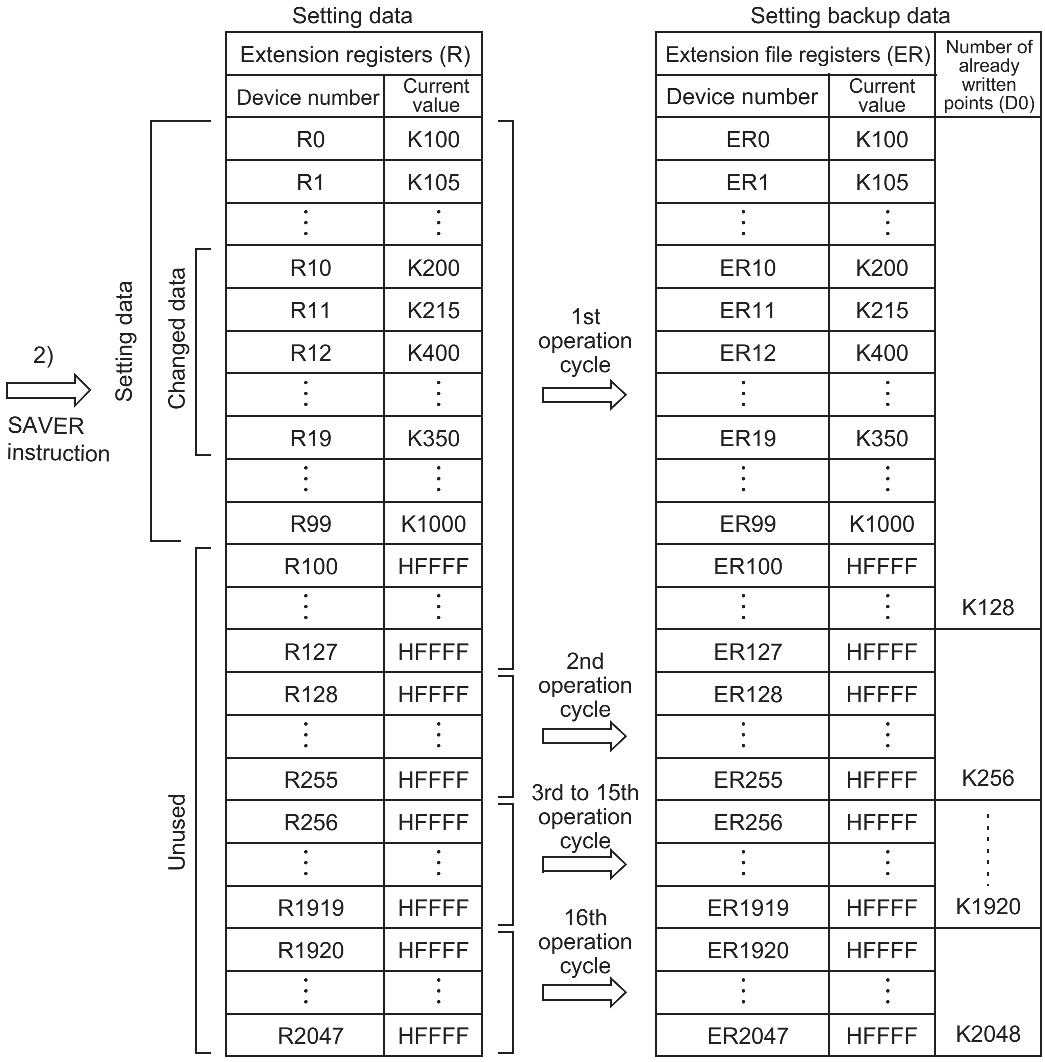
Setting data: Extension registers (R) Device number	Current value	Setting backup data: Extension file registers (ER) Device number	Current value
R0	K100	ER0	K100
R1	K105	ER1	K105
⋮	⋮	⋮	⋮
R10	K200	ER10	K300
R11	K215	ER11	K330
R12	K400	ER12	K350
⋮	⋮	⋮	
R19	K350	ER19	K400
⋮	⋮	⋮	⋮
R99	K1000	ER99	K1000
R100	HFFFF	ER100	HFFFF
⋮	⋮	⋮	⋮
R2047	HFFFF	ER2047	HFFFF

1) ⇒ INITER instruction

Setting backup data: Extension file registers (ER) Device number	Current value
ER0	HFFFF
ER1	HFFFF
⋮	⋮
ER10	HFFFF
ER11	HFFFF
ER12	HFFFF
⋮	⋮
ER19	HFFFF
⋮	⋮
ER99	HFFFF
ER100	HFFFF
⋮	⋮
ER2047	HFFFF

2) ⇒ SAVER instruction

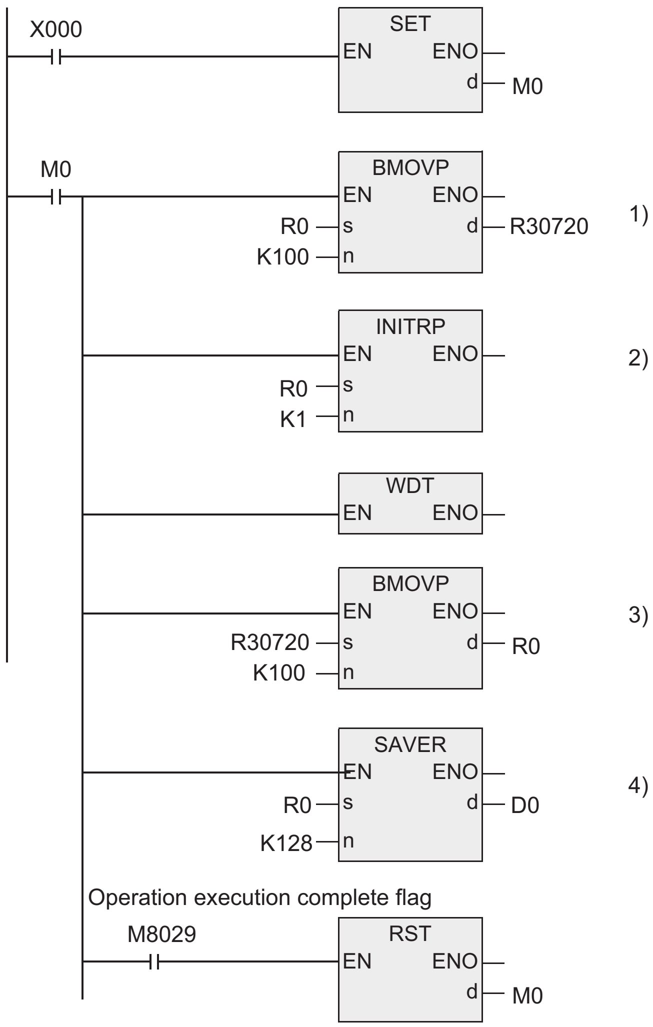
To the next page


Setting data
Extension registers (R)
Device number
Current value
R0 K100
R1 K105
R10 K200
R11 K215
R12 K400
R19 K350
R99 K1000
R100 HFFFF
R127 HFFFF
R128 HFFFF
R255 HFFFF
R256 HFFFF
R1919 HFFFF
R1920 HFFFF
R2047 HFFFF
2)
SAVER instruction
Setting data
Changed data
Unused
1st operation cycle
2nd operation cycle
3rd to 15th operation cycle
16th operation cycle
Setting backup data
Extension file registers (ER)
Device number
Current value
Number of already written points (D0)
ER0 K100
ER1 K105
ER10 K200
ER11 K215
ER12 K400
ER19 K350
ER99 K1000
ER100 HFFFF
K128
ER127 HFFFF
ER128 HFFFF
ER255 HFFFF
K256
ER256 HFFFF
ER1919 HFFFF
K1920
ER1920 HFFFF
ER2047 HFFFF
K2048

2) In the case of FX3UC PLCs former than Ver. 1.30
In the program example shown below, the changed contents of the extension registers R10 to R19 (sector 0) used for setting data are reflected on extension file registers (ER) when X000 is set to ON. (128 points are written in one operation cycle.)

Program

[Structured ladder]



Operation example

Setting data		Setting backup data	
Extension registers (R)		Extension file registers (ER)	
Device number	Current value	Device number	Current value
R0	K100	ER0	K100
R1	K105	ER1	K105
⋮	⋮	⋮	⋮
R10	K200	ER10	K300
R11	K215	ER11	K330
R12	K400	ER12	K350
⋮	⋮	⋮	⋮
R19	K350	ER19	K400
⋮	⋮	⋮	⋮
R99	K1000	ER99	K1000
R100	HFFFF	ER100	HFFFF
⋮	⋮	⋮	⋮
R2047	HFFFF	ER2047	HFFFF

1) BMOV instruction

Temporarily withdrawn data *1	
Unused extension registers	
Device number	Current value
R30720	K100
R30721	K105
⋮	⋮
R30730	K200
R30731	K215
R30732	K400
⋮	⋮
R30739	K350
⋮	⋮
R30819	K1000

2) INITR instruction

*1. Use unused devices as an area to which data is temporarily withdrawn.

To the next page

2) INITR instruction

Setting data: Extension registers (R)		Setting backup data: Extension file registers (ER)	
Device number	Current value	Device number	Current value
R0	HFFFF	ER0	HFFFF
R1	HFFFF	ER1	HFFFF
⋮	⋮	⋮	⋮
R10	HFFFF	ER10	HFFFF
R11	HFFFF	ER11	HFFFF
R12	HFFFF	ER12	HFFFF
⋮	⋮	⋮	⋮
R19	HFFFF	ER19	HFFFF
⋮	⋮	⋮	⋮
R99	HFFFF	ER99	HFFFF
R100	HFFFF	ER100	HFFFF
⋮	⋮	⋮	⋮
R2047	HFFFF	ER2047	HFFFF

3) BMOV instruction

Setting data (A)

Setting data: Unused extension registers	
Device number	Current value
R0	K100
R1	K105
⋮	⋮
R10	K200
R11	K215
R12	K400
⋮	⋮
R19	K350
⋮	⋮
R99	K1000
R100	HFFFF
⋮	⋮
R2047	HFFFF

4) SAVER instruction

Setting data / Changed data / Unused

1st operation cycle / 2nd operation cycle / 3rd to 15th operation cycle / 16th operation cycle

Setting data: Extension registers (R)		Setting backup data: Extension file registers (ER)		
Device number	Current value	Device number	Current value	Number of already written points (D0)
R0	K100	ER0	K100	
R1	K105	ER1	K105	
⋮	⋮	⋮	⋮	
R10	K200	ER10	K200	
R11	K215	ER11	K215	
R12	K400	ER12	K400	
⋮	⋮	⋮	⋮	
R19	K350	ER19	K350	
⋮	⋮	⋮	⋮	
R99	K1000	ER99	K1000	
R100	HFFFF	ER100	HFFFF	
⋮	⋮	⋮	⋮	
R127	HFFFF	ER127	HFFFF	K128
R128	HFFFF	ER128	HFFFF	
⋮	⋮	⋮	⋮	
R255	HFFFF	ER255	HFFFF	K256
R256	HFFFF	ER256	HFFFF	
⋮	⋮	⋮	⋮	⋮
R1919	HFFFF	ER1919	HFFFF	K1920
R1920	HFFFF	ER1920	HFFFF	
⋮	⋮	⋮	⋮	
R2047	HFFFF	ER2047	HFFFF	K2048

7.27.3 INITR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction initializes (to "HFFFF <K-1>") extension registers (R) in the RAM built in a PLC and extension file registers in a memory cassette (flash memory) before data logging by LOGR instruction.
In FX3UC PLCs former than Ver. 1.30, use this instruction to initialize extension file registers (ER) before writing data to them using SAVER instruction.
In FX3UC PLCs Ver. 1.30 or later and FX3U PLCs, INITER instruction is also provided to initialize (to "HFFFF" <K-1>) only extension file registers (ER) in a memory cassette (flash memory) in units of sector.

→ **For SAVER instruction, refer to Section 7.27.2.**
→ **For LOGR instruction, refer to Section 7.27.4.**
→ **For INITER instruction, refer to Section 4.27.6.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
INITR	16 bits	Continuous	INITR EN ENO s n	INITR(EN,s,n);
INITRP	16 bits	Pulse	INITRP EN ENO s n	INITRP(EN,s,n);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Device of extension register and extension file register to be initialized It is possible to specify only the head device in a sector of extension registers.	ANY16
	(n)	Number of sectors of extension registers and extension file registers to be initialized.	ANY16
Output variable	ENO	Execution state	Bit

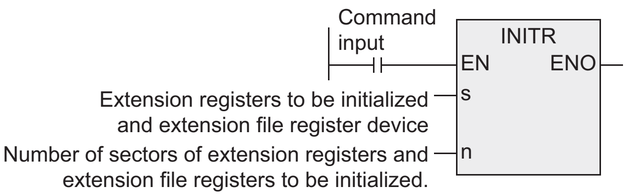
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)															●				●					
(n)																				●	●			

Function and operation explanation

1. 16-bit operation (INITR/INITRP)

"n" sectors of extension registers in the PLC's built-in RAM starting from the one specified by (S) and "n" sectors of extension file registers in a memory cassette (flash memory) having the same device numbers are initialized (initial value "HFFFF" <K-1> is written.).
Initialization is executed in units of sector.



The table below shows the head device number in each sector.

Sector number	Head device number	Initialized device range
Sector 0	R0	R0 to R2047, ER0 to ER2047
Sector 1	R2048	R2048 to R4095, ER2048 to ER4095
Sector 2	R4096	R4096 to R6143, ER4096 to ER6143
Sector 3	R6144	R6144 to R8191, ER6144 to ER8191
Sector 4	R8192	R8192 to R10239, ER8192 to ER10239
Sector 5	R10240	R10240 to R12287, ER10240 to ER12287
Sector 6	R12288	R12288 to R14335, ER12288 to ER14335
Sector 7	R14336	R14336 to R16383, ER14336 to ER16383

Sector number	Head device number	Initialized device range
Sector 8	R16384	R16384 to R18431, ER16384 to ER18431
Sector 9	R18432	R18432 to R20479, ER18432 to ER20479
Sector 10	R20480	R20480 to R22527, ER20480 to ER22527
Sector 11	R22528	R22528 to R24575, ER22528 to ER24575
Sector 12	R24576	R24576 to R26623, ER24576 to ER26623
Sector 13	R26624	R26624 to R28671, ER26624 to ER28671
Sector 14	R28672	R28672 to R30719, ER28672 to ER30719
Sector 15	R30720	R30720 to R32767, ER30720 to ER32767

Operation (when a memory cassette is used)

1) Extension registers (R) [inside the built-in RAM memory]

Device number	Current value	
	Before execution	After execution
(S)	H0010	HFFFF
(S)+1	H0020	HFFFF
(S)+2	H0011	HFFFF
⋮	⋮	⋮
(S)+(2048×n)-1	HABCD	HFFFF

2) Extension file registers (ER) [inside the memory cassette]

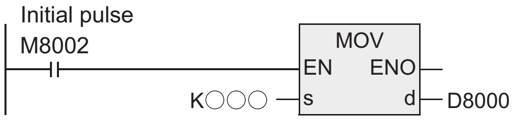
Device number	Current value	
	Before execution	After execution
(S)	H1234	HFFFF
(S)+1	H5678	HFFFF
(S)+2	H90AB	HFFFF
⋮	⋮	⋮
(S)+(2048×n)-1	HCDEF	HFFFF

Cautions

1. Initializing two or more sectors

When a memory cassette is attached, 18 ms is required to initialize one sector.
(When a memory cassette is not attached, only 1 ms is required to initialize one sector.)
When initializing two or more sectors, take either measures shown below.

1) Set a large value to the watchdog timer D8000 using the following program.



Guideline of the watchdog timer set value

A value acquired by the following procedure can be regarded as the guideline of the watchdog timer set value.
If an acquired value is 200 ms or less, however, it is not necessary to change the watchdog timer set value.

a) Write a program to be executed from GX Developer to the PLC.
[Online] → [Write to PLC ...]

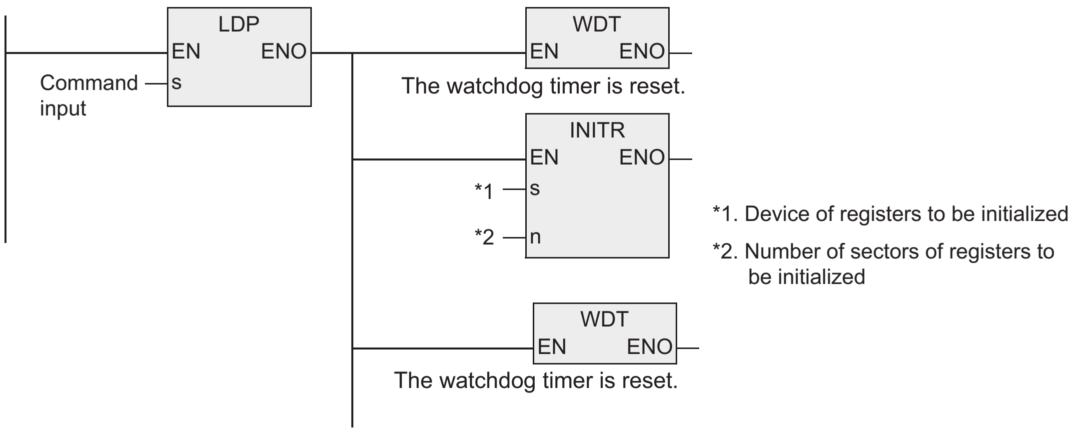
b) Set the current value of D8000 (unit: ms) to "1000" using the device test function in GX Developer.
[Online] → [Debug] → [Device test ...] → "Word device / buffer memory" in Device test dialog box

c) Set the PLC mode to RUN, and execute the program. (Execute this instruction also.)

d) Monitor the maximum scan time D8012 (unit: 0.1 ms) using the device batch monitoring function in GX Developer.

e) Set the watchdog timer to the maximum scan time (D8012) or more.
D8012 stores the maximum scan time in increments of 0.1 ms.
Rough guide to the watchdog timer set value D8000 (unit: ms) is the "value stored in D8012 divided by 10" added by 50 to 100.

2) Setting WDT instruction just before and after INITR instruction as shown below.



If the processing time of the INITR instruction exceeds 200 ms, set the value of D8000 (unit: ms) to the processing time or more.

2. About the allowable number of times of writing operations in memory

Note the following when accessing the extension file registers:

- The memory cassette (flash memory) allows up to 10,000 times of writing operations.
  The number of times of writing operations counts up each time the INITR, RWER or INITER instruction is executed. Do not let the number of times of writing operations exceed the allowable number of times of writing operations.
  When a continuous operation type instruction is executed, writing operation to the memory occurs for each operation cycle of the PLC. To avoid this, be sure to use pulse operation type instructions.
- The number of times of writing operations does not count up when the LOADR, SAVER or LOGR instruction is executed. However, the SAVER and LOGR instructions require the target write sectors to be initialized before executing the instructions. Note that, when initializing by using the INITR or INITER instruction, the number of times of writing operations in the memory counts up every time the NITR or INITER instruction is executed.

Error

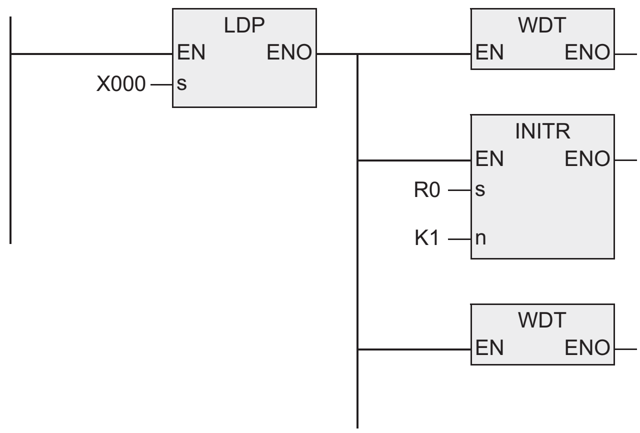
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When any device number other than the head device number of a sector of extension file registers is set to (s).
(Error code: K6706)
2) When a device number to be initialized exceeds "32767". In this case, devices up to R32767 (ER32767) are initialized.
(Error code: K6706)
3) When the protect switch of the memory cassette is set to ON. (Error code: K6770)

Program example

In the program example shown below, the extension registers R0 to R2047 in the sector 0 are initialized. Note that the extension file registers ER0 to ER2047 are also initialized if a memory cassette is attached.

[Structured ladder]



[ST]

```
IF(LDP(TRUE,X000) THEN WDT(TRUE);
IF(LDP(TRUE,X000) THEN INITR(TRUE,R0,K1));
IF(LDP(TRUE,X000) THEN WDT(TRUE);
```

1) Extension registers (R) [inside the built-in RAM memory]

Device number	Current value	
	Before execution	After execution
RO	H1234	HFFFF
R1	H5678	HFFFF
R2	H90AB	HFFFF
⋮	⋮	⋮
R2047	HCDEF	HFFFF

7.27.4 LOGR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
○	×	×	×	×	×	×	×

Outline

This instruction logs specified devices, and stores the logged data to extension registers (R) and extension file registers (ER) in a memory cassette.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Variable	Expression in each language: ST
LOGR	16 bits	Continuous	LOGR EN ENO s d1 m d2 n	LOGR(EN,s,m,n,d1,d2);
LOGRP	16 bits	Pulse	LOGRP EN ENO s d1 m d2 n	LOGRP(EN,s,m,n,d1,d2);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Head device to be logged	ANY16
	m	Number of devices to be logged (1≤ m ≤ 8000)	ANY16
	n	Number of sectors of devices used in logging (1 ≤ n ≤ 16)	ANY16
Output variable	ENO	Execution state	Bit
	d1	Head device used in logging	ANY16
	d2	Number of pieces of logged data	ANY16

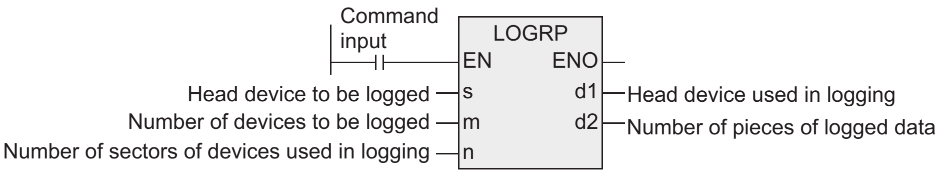
3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s												●	●	●					●					
m														●						●	●			
d1															●									
n																				●	●			
d2														●					●					

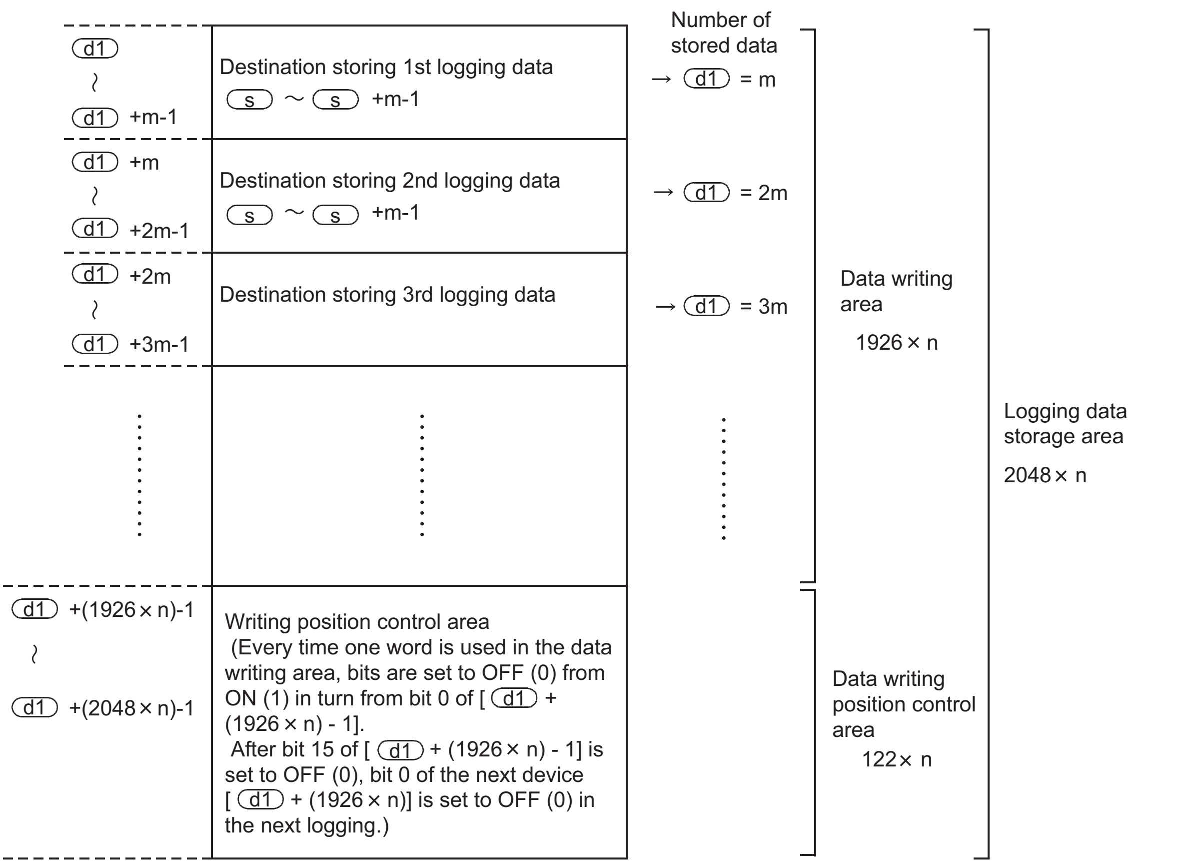
Function and operation explanation

1. 16-bit operation (LOGR/LOGRP)

While the instruction is driven, "m" devices starting from the device specified by (s) are logged until "n" sectors of extension registers (R) starting from the device specified by (d1) and extension file registers (ER) in a memory cassette are filled.
The number of pieces of logged data is stored to the device specified by (d2).
If a memory cassette is not used, data is not written to extension file registers (ER).



Logging data format



The table below shows the head device number in each sector.

Sector number	Head device number	Written device range
Sector 0	R0	R0 to R2047, ER0 to ER2047
Sector 1	R2048	R2048 to R4095, ER2048 to ER4095
Sector 2	R4096	R4096 to R6143, ER4096 to ER6143
Sector 3	R6144	R6144 to R8191, ER6144 to ER8191
Sector 4	R8192	R8192 to R10239, ER8192 to ER10239
Sector 5	R10240	R10240 to R12287, ER10240 to ER12287
Sector 6	R12288	R12288 to R14335, ER12288 to ER14335
Sector 7	R14336	R14336 to R16383, ER14336 to ER16383

Sector number	Head device number	Written device range
Sector 8	R16384	R16384 to R18431, ER16384 to ER18431
Sector 9	R18432	R18432 to R20479, ER18432 to ER20479
Sector 10	R20480	R20480 to R22527, ER20480 to ER22527
Sector 11	R22528	R22528 to R24575, ER22528 to ER24575
Sector 12	R24576	R24576 to R26623, ER24576 to ER26623
Sector 13	R26624	R26624 to R28671, ER26624 to ER28671
Sector 14	R28672	R28672 to R30719, ER28672 to ER30719
Sector 15	R30720	R30720 to R32767, ER30720 to ER32767

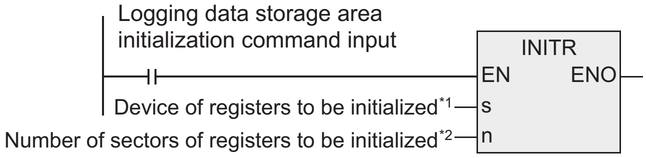
Cautions

1. About LOGR instruction

LOGR instruction executes logging in each operation in the continuous operation type.
When logging should be executed only once by one input, use the pulse operation type.

2. Cautions on using a memory cassette

Flash memory is adopted in a memory cassette. Be sure to initialize the data storage area in units of sector before starting logging.
If this instruction is executed without initialization, an operation error (error code: K6770) may be caused.



*1. Specify the same device as (d1) in LOGR instruction.

*2. Specify the same number as (n) in LOGR instruction.

3. About the allowable number of times of writing operations in memory

- Note the following when accessing the extension file registers:
  The memory cassette (flash memory) allows up to 10,000 times of writing operations.
  The number of times of writing operations counts up each time the INITR, RWER or INITER instruction is executed. Do not let the number of times of writing operations exceed the allowable number of times of writing operations.
  When a continuous operation type instruction is executed, writing operation to the memory occurs for each operation cycle of the PLC. To avoid this, be sure to use pulse operation type instructions.
- The number of times of writing operations does not count up when the LOADR, SAVER or LOGR instruction is executed. However, the SAVER and LOGR instructions require the target write sectors to be initialized before executing the instructions. Note that, when initializing by using the INITR or INITER instruction, the number of times of writing operations in the memory counts up every time the NITR or INITER instruction is executed.

Error

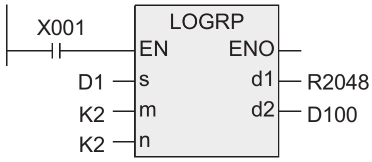
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When any number other than the head device number of a sector of extension file registers is set to the device specified by (s).
   (Error code: K6706)
2) While data is written, the remaining area and the data quantity to be written are compared with each other. If the remaining storage area is insufficient, only a limited amount of data is written. (Error code: K6706)
3) When the protect switch of the memory cassette is set to ON. (Error code: K6770)
4) When the collation result after data writing is "mismatch" due to omission of initialization or for another reason. (Error code: K6770)

Program example

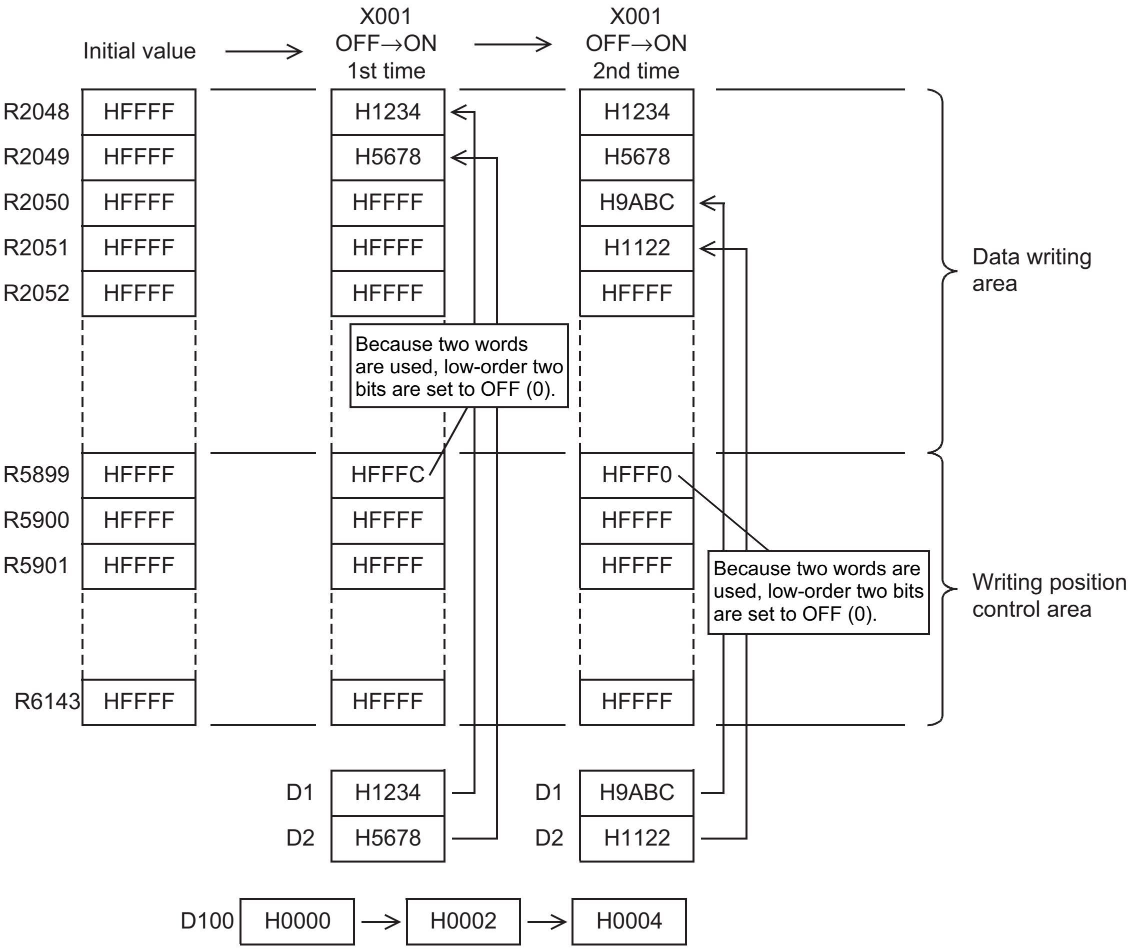
In the program example shown below, D1 and D2 are logged to the area from R2048 to R6143 every time X001 turns ON.

[Structured ladder]



[ST]

```
LOGR(X001,D1,K2,K2,R2048,D100);
```



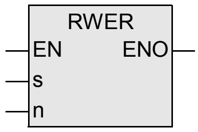
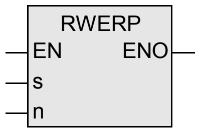
7.27.5 RWER

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	○	×	×	×	×	×	×

Outline

This instruction writes the current values of an arbitrary number of extension registers (R) in the PLC's built-in RAM to extension file registers (ER) in a memory cassette (flash memory or EEPROM) or to the extension file registers (ER) in the PLC's built-in EEPROM.
Because RWER instruction is not supported in FX3UC PLCs former than Ver. 1.30, use SAVER instruction instead.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
RWER	16 bits	Continuous		RWER(EN,s,n);
RWERP	16 bits	Pulse		RWERP(EN,s,n);

2. Set data

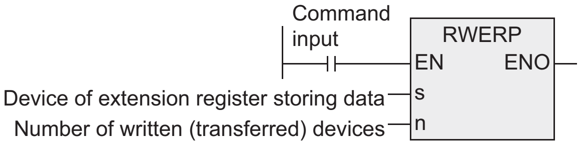
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Device of extension register storing data	ANY16
	n	Number of written (transferred) devices [FX3G: 1 ≤ n ≤ 24000, FX3U/FX3UC: 0 ≤ n ≤ 32767]	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s															●				●					
n														●						●	●			

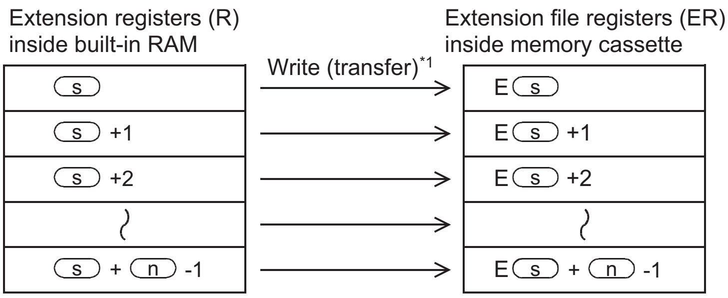
Function and operation explanation

1. 16-bit operation (RWER)



1) For the FX3U and FX3UC PLCs

The contents (current values) of "n" extension registers (R) starting from (s) are written (transferred) to extension file registers having the same device numbers in a memory cassette (flash memory).



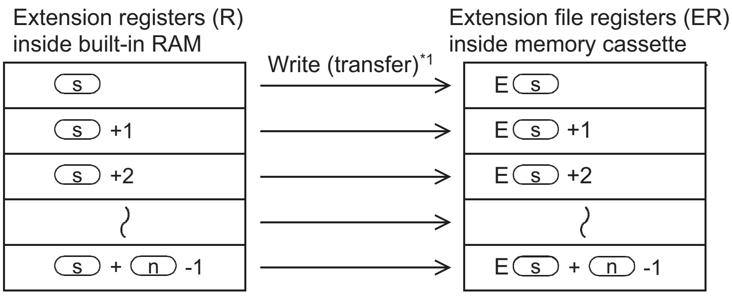
*1. All points specified by the instruction are written (transferred).

- When "n" is set to "0", it is handled as "32768" when the instruction is executed.

2) For the FX3G PLCs

a) When connecting a memory cassette

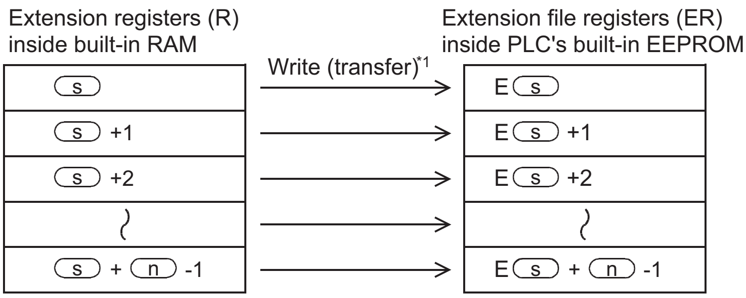
The contents (current values) of the "n" points of extension registers (R) starting from (s) are written (transferred) to the extension file registers (ER) having the same numbers as the extension registers (R) in the memory cassette (EEPROM).



*1. All points specified by the instruction are written (transferred).

b) When not connecting a memory cassette

The contents (current values) of the "n" points of extension registers (R) starting from (s) are written (transferred) to the extension file registers (ER) having the same numbers as the extension registers (R) in the PLC's built-in EEPROM.



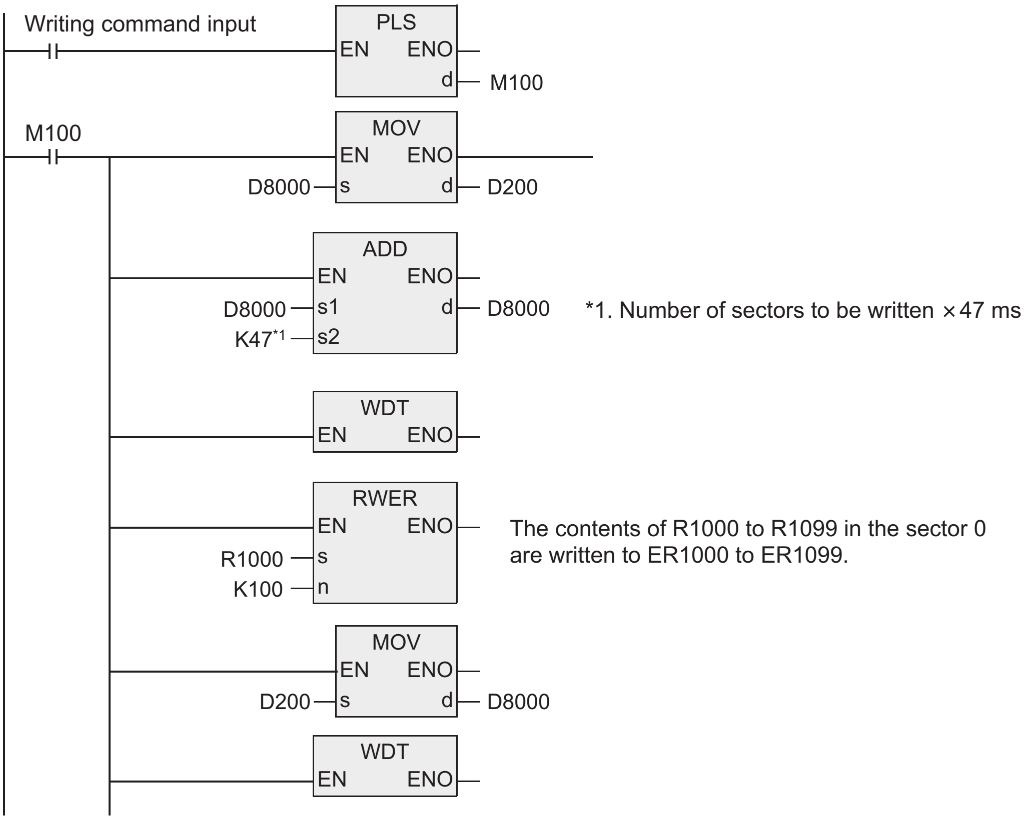
*1. All points specified by the instruction are written (transferred).

Cautions

1. Cautions on writing data to the memory cassette (flash memory) for the FX3U and FX3UC PLCs

Memory cassettes adopt flash memory. Note the following contents when writing data to extension file registers in a memory cassette with the RWER instruction.

1) Though extension file registers to be written can be specified arbitrarily, writing is executed in units of sector.
It takes about 47 ms to write one sector. If the extension file registers to be written are located in two sectors, the instruction execution time will be about 94 ms.
Be sure to change the set value of the watchdog timer D8000 before executing this instruction.



The table below shows the device range in each sector.

Sector number	Device range	Sector number	Device range
Sector 0	ER0 to ER2047	Sector 8	ER16384 to ER18431
Sector 1	ER2048 to ER4095	Sector 9	ER18432 to ER20479
Sector 2	ER4096 to ER6143	Sector 10	ER20480 to ER22527
Sector 3	ER6144 to ER8191	Sector 11	ER22528 to ER24575
Sector 4	ER8192 to ER10239	Sector 12	ER24576 to ER26623
Sector 5	ER10240 to ER12287	Sector 13	ER26624 to ER28671
Sector 6	ER12288 to ER14335	Sector 14	ER28672 to ER30719
Sector 7	ER14336 to ER16383	Sector 15	ER30720 to ER32767

2) Do not turn OFF the power while this instruction is being executed. If the power is turned OFF, execution of this instruction may be aborted. If execution is aborted, the data may be lost.
Be sure to back up the data before executing this instruction.
→ **For the backup method, refer to the next page.**

3) The FX3UC PLCs of V1.30 or later support the RWER instruction.

2. Cautions on writing data to the memory cassette (EEPROM) for the FX3G PLCs

Memory cassettes adopt EEPROM. Note the following contents when writing data to extension file registers in a memory cassette with the RWER instruction.

- Do not turn OFF the power while this instruction is being executed. If the power is turned OFF, execution of this instruction may be aborted. If execution is aborted, the data may be lost.
  Be sure to back up the data before executing this instruction.

3. About the allowable number of times of writing operations in memory

Note the following when accessing the extension file registers:

- For the FX3U and FX3UC PLCs
  The memory cassette (flash memory) allows up to 10,000 times of writing operations.
  The number of times of writing operations counts up each time the INITR, RWER or INITER instruction is executed. Do not let the number of times of writing operations exceed the allowable number of times of writing operations. When a continuous operation type instruction is executed, writing operation to the memory occurs for each operation cycle of the PLC. To avoid this, be sure to use pulse operation type instructions.
  The number of times of writing operations does not count up when the LOADR, SAVER or LOGR instruction is executed. However, the SAVER and LOGR instructions require the target write sectors to be initialized before executing the instructions. Note that, when initializing by using the INITR or INITER instruction, the number of times of writing operations in the memory counts up every time the NITR or INITER instruction is executed.
- For the FX3G PLCs
  The memory cassette (EEPROM) and PLC's built-in memory (EEPROM) allow up to 10,000 times and 20,000 times of writing operations, respectively. The number of times of writing operations counts up each time the RWER instruction is executed. Do not let the number of times of writing operations exceed the allowable number of times of writing operations.
  When a continuous operation type instruction is executed, writing operation to the memory occurs for each operation cycle of the PLC.
  To avoid this, be sure to use pulse operation type instructions.
  The number of times of writing operations does not count up when the LOADR instruction is executed.

Error

An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When the last device number to be transferred exceeds "32767" (Error code: K6706).
   At this time, the data up to the last device number of "R32767"*1 is read (transferred).
2) When a memory cassette is not connected. (Error code: K6771)*2
3) When the protect switch of the memory cassette is set to ON. (Error code: K6770)

*1. For the FX3G PLCs, the last device number is "23999".

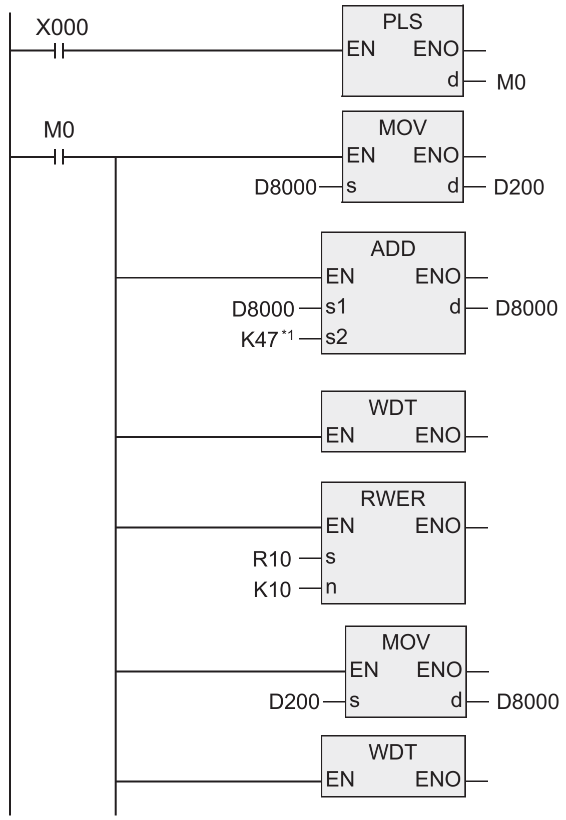
*2. This does not cause an error with the FX3G PLCs because the PLCs read the contents of the extension file registers stored in the PLC's built-in EEPROM even if a memory cassette is not connected.

Program example

In the program example shown below, the changed contents of extension registers R10 to R19 (sector 0) used for setting data are reflected on extension file registers (ER) when X000 turns ON.

Program

[Structured ladder]



[ST]

```
PLS(X000,M0);
MOV(M0,D8000,D200);
ADD(M0,D8000,K47,D8000);
WDT(M0);
RWER(M0,R10,K10);
MOV(M0,D200,D8000);
WDT(M0);
```

Operation example

Setting data

Extension registers (R)	
Device number	Current value
R0	K100
R1	K105
⋮	⋮
R10	K200
R11	K215
R12	K400
⋮	⋮
R19	K350
⋮	⋮
R99	K1000
R100	HFFFF
⋮	⋮
R2047	HFFFF

Setting data (A): R0 to R99; Changed data: R10 to R19; Unused: R100 to R2047

Write (when X000 turns ON.) ⇒

Setting backup data

Extension file registers (ER)	
Device number	Current value
ER0	K100
ER1	K105
⋮	⋮
ER10	K200
ER11	K215
ER12	K400
⋮	⋮
ER19	K350
⋮	⋮
ER99	K1000
ER100	HFFFF
⋮	⋮
ER2047	HFFFF

7.27.6 INITER

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

This instruction initializes extension file registers (ER) to "HFFFF" (<K-1>) in a memory cassette (flash memory) before executing the SAVER instruction.
Because the INITER instruction is not supported in FX3UC PLCs earlier than Ver. 1.30, use INITR instruction instead.

→ For SAVER instruction, refer to Section 7.27.2.
→ For INITR instruction, refer to Section 7.27.3.

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
INITER	16 bits	Continuous	INITER EN ENO s n	INITER(EN,s,n);
INITERP	16 bits	Pulse	INITERP EN ENO s n	INITERP(EN,s,n);

2. Set data

Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Device number of extension register sector with the same device number as the extension file register to be initialized. It is possible to specify only the head device in a sector of extension registers.	ANY16
	(n)	Number of sectors of extension registers and extension file registers to be initialized.	ANY16
Output variable	ENO	Execution state	Bit

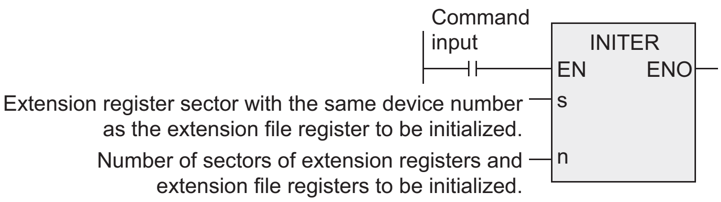
3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit specification				System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)															●				●					
(n)																				●	●			

Function and operation explanation

1. 16-bit operation (INITER/INITERP)

"n" sectors of extension file registers (ER) in a memory cassette (flash memory) with the same device number as the device specified by (s) are initialized (initial value "HFFFF" (<K-1>) is writen.).
Initialization is executed in sectors.



The table below shows the head device number in each sector.

Sector number	Head device number	Initialized device range	Sector number	Head device number	Initialized device range
Sector 0	R0	ER0 to ER2047	Sector 8	R16384	ER16384 to ER18431
Sector 1	R2048	ER2048 to ER4095	Sector 9	R18432	ER18432 to ER20479
Sector 2	R4096	ER4096 to ER6143	Sector 10	R20480	ER20480 to ER22527
Sector 3	R6144	ER6144 to ER8191	Sector 11	R22528	ER22528 to ER24575
Sector 4	R8192	ER8192 to ER10239	Sector 12	R24576	ER24576 to ER26623
Sector 5	R10240	ER10240 to ER12287	Sector 13	R26624	ER26624 to ER28671
Sector 6	R12288	ER12288 to ER14335	Sector 14	R28672	ER28672 to ER30719
Sector 7	R14336	ER14336 to ER16383	Sector 15	R30720	ER30720 to ER32767

Operation example

1) Extension file registers (ER) [inside the memory cassette]

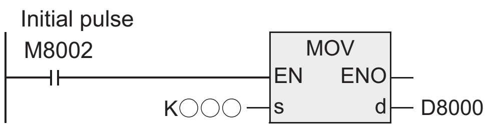
Device number	Current value	
	Before execution	After execution
(s)	H1234	HFFFF
(s)+1	H5678	HFFFF
(s)+2	H90AB	HFFFF
⋮	⋮	⋮
(s)+(2048×n)-1	HCDEF	HFFFF

Cautions

1. About 25 ms is required to initialize one sector.

When initializing two or more sectors, take either measure shown below.

1) Set a large value to the watchdog timer D8000 using the following program.



Guideline of the watchdog timer set value

A value acquired by the following procedure can be regarded as the guideline of the watchdog timer set value.
If an acquired value is 200 ms or less, however, it is not necessary to change the watchdog timer set value.

a) Write a program to be executed from GX Developer to the PLC.
[Online] → [Write to PLC ...]

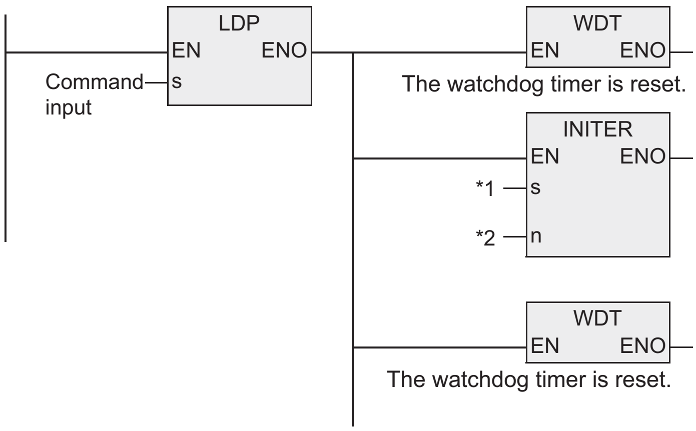
b) Set the current value of D8000 (unit: ms) to "1000" using the device test function in GX Developer.
[Online] → [Debug] → [Device test ...] → "Word device / buffer memory" in Device test dialog box

c) Set the PLC mode to RUN, and execute the program. (Execute this instruction also.)

d) Monitor the maximum scan time D8012 (unit: 0.1 ms) using the device batch monitoring function in GX Developer.

e) Set the watchdog timer to the maximum scan time (D8012) or more.
D8012 stores the maximum scan time in increments of 0.1 ms.
Rough guide to the watchdog timer set value D8000 (unit: ms) is the "value stored in D8012 divided by 10" added by 50 to 100.

2) Setting WDT instruction just before and after INITER instruction as shown below:



*1. Device number of the extension registers to be initialized

*2. Device number of the extension registers to be initialized

If the processing time of the INITR instruction exceeds 200 ms, set the value of D8000 (unit: ms) to the processing time or more.

2. About the allowable number of times of writing operations in memory

Note the following when accessing the extension file registers:

- The memory cassette (flash memory) allows up to 10,000 times of writing operations.
  The number of times of writing operations counts up each time the INITR, RWER or INITER instruction is executed. Do not let the number of times of writing operations exceed the allowable number of times of writing operations. When a continuous operation type instruction is executed, writing operation to the memory occurs for each operation cycle of the PLC. To avoid this, be sure to use pulse operation type instructions.
- The number of times of writing operations does not count up when the LOADR, SAVER or LOGR instruction is executed. However, the SAVER and LOGR instructions require the target write sectors to be initialized before executing the instructions. Note that, when initializing by using the INITR or INITER instruction, the number of times of writing operations in the memory counts up every time the NITR or INITER instruction is executed.

3. The FX3UC PLCs of V1.30 or later support the RWER instruction.

Error

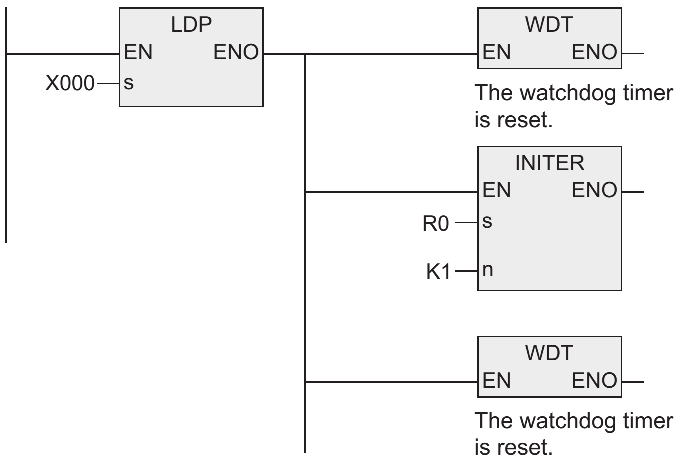
An operation error is caused in the following cases. The error flag M8067 turns ON, and the error code is stored in D8067.

1) When any device number other than the head device number of a sector of extension file registers (ER) is set to (s).
   (Error code: K6706)
2) When a device number to be initialized exceeds "32767".
   In this case, devices up to ER32767 are initialized. (Error code: K6706)
3) When the protect switch of the memory cassette is set to ON. (Error code: K6770)
4) When a memory cassette is not connected. (Error code: K6771)

Program example

In the program example shown below, the extension file registers ER0 to ER2047 in sector 0 are initialized.

[Structured ladder]



[ST]

```
IF(LDP(TRUE,X000) THEN WDT(TRUE));
IF(LDP(TRUE,X000) THEN INITER(TRUE,R0,K1));
IF(LDP(TRUE,X000) THEN WDT(TRUE));
```

1) Extension file registers (ER) [inside the memory cassette]

Device number	Current value	
	Before execution	After execution
ERO	H1234	HFFFF
ER1	H5678	HFFFF
ER2	H90AB	HFFFF
⋮	⋮	⋮
ER2047	HCDEF	HFFFF

7.28 FX3U-CF-ADP

7.28.1 FLCRT

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

The FLCRT instruction creates a file inside the CompactFlash[TM] card mounted in the FX3U-CF-ADP. When executed after creation of a new file, the FLCRT instruction checks the association with the file ID, and evaluates it.

→ **As for explanation of the instruction, see the FX3U-CF-ADP User's Manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
FLCRT	16 bits	Continuous	FLCRT EN ENO s1 s2 s3 n	FLCRT(EN,s1,s2,s3,n);

2. Set data

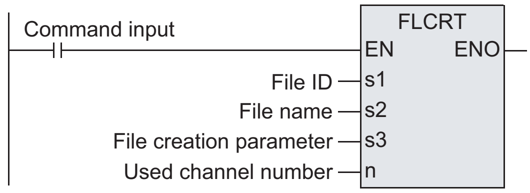
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	File ID (Refer to Detailed explanation of setting data)	ANY16
	s2	File name (Refer to Detailed explanation of setting data)	String
	s3	File creation parameter (Refer to Detailed explanation of setting data)	ANY16(0..3)
	n	Used channel number [contents of setting : K1 = ch1, K2 = ch2]	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	●				●	●	●			
s2														●	●				●				●	
s3														●	●				●					
n																				●	●			

Function and operation explanation

1. 16-bit operation (FLCRT)



1) When the file ID is "K0"
   When s1 is "K0", the FLCRT instruction creates a FIFO file.
   When the PLC creates two or more files for FIFO file, and executes FIFO (first in, first out) in units of files. The PLC keeps the latest file, and deletes older files so that the total capacity of FIFO files and other files does not exceed the specified capacity.
2) When the file ID is "K1" to "K63"
   When s1 is "K1" to "K63", the FLCRT instruction creates a file having the specified file name.
   Sequence programs use the file ID for specifying a file. Accordingly, each file name saved in the CompactFlash™ card is associated with the file ID, and controlled by the file ID table.
   If a file having the specified file name already exists and is registered in the file ID table, the PLC finishes the FLCRT instruction without executing any processing.
   If a file having the specified name already exists but is not registered in the file ID table, the PLC only registers the existing file to the file ID table.

Detailed explanation of setting data

Details of the setting data in the FLCRT instruction are as shown below.

Setting items		Description	Data Type
(s1)		File ID This ID number is associated with the file name. The FLCRT instruction creates a file, and associates the file name with the file ID at the same time. The user should use the file ID for specifying a file after that. Allowable setting range : K0 to K63 ("K0" indicates "FIFO file".)	ANY16
(s2)		File name When (s1) is "K0 (FIFO file)" Not used (ignored) Use an unused device. (D or R) When (s1) is "K1" to "K63" Specify the file name in up to 8 characters until "null" or "null + null". Half-width alphanumeric characters and half-width symbols permitted in the MS-DOS are available. Half-width symbols : !, #, $, %, &, ', (,), +, -, @, ^, _, ', [,], ~ The extension is fixed to "CSV"	String
File creation parameter	(s3)	Time stamp setting Set whether or not the time stamp is added to the file. Specify the format when adding the time stamp. K0 : None (NULL) K1 : yyyy/mm/dd hh:mm:ss K2 : yy/mm/dd hh:mm:ss K3 : dd/mm/yyyy hh:mm:ss K4 : dd/mm/yy hh:mm:ss K5 : mm/dd/yyyy hh:mm:ss K6 : mm/dd/yy hh:mm:ss K7 : hh:mm:ss	ANY16(0..3)
	(s3) +1	Data type Set the data type to be saved. K0 : No data type specification (mixed type) K1 : Bit type K2 : Decimal type (16-bit) K3 : Decimal type (32-bit) K4 : Hexadecimal type (16-bit) K5 : Hexadecimal type (32-bit) K6 : Real numbers(Floating point data) Exponent expression type (32-bit) K7 : Character string	
	(s3) +2	Maximum number of lines Set the maximum number of lines. Allowable setting range : K1 to K32767*1	
	(s3) +3	When (s1) is "K0 (FIFO file)" Set the CompactFlash™ card use ratio. Specify the ratio (%) out of the whole CompactFlash™ card capacity to be used. Allowable setting range : 10 to 90 (%)	
		When (s1) is "K1" to "K63" File processing to be executed when the specified maximum number of lines is reached. Set the file processing method to be executed when the number of lines reaches the specified maximum value. K0 : Stops execution. (The line position remains the specified maximum line position.) K1 : Returns to the head (ring buffer file).	
(n)		Channel number used by the CF-ADP K1 : ch1 K2 : ch2	ANY16

*1. Adjust the maximum number of lines to specify the file size available in the used application software used.
For the file size calculation formula, refer to FX3U-CF-ADP User's Manual

Cautions

1) When the file ID is "K0"
   a) The CF-ADP can create up to 1000 files (within the CompactFlash™ card capacity).
   b) The file name is set to "FILE0000.CSV" to "FILE0999.CSV".
2) When the file ID is "K1" to "K63"
   a) The user can create up to 63 files (within the CompactFlash™ card capacity).
   b) The FLCRT instruction is completed abnormally if different file names are specified for the same file ID or if the same file name is specified for different file IDs.
3) The FX3U and FX3UC PLCs supports the instruction at V2.61 or later.
   The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

7.28.2 FLDEL

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

The FLDEL instruction deletes files stored in the CompactFlash™ card, or formats the CompactFlash™ card.

→ **As for explanation of the instruction, see the FX3U-CF-ADP User's Manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language: Structured ladder	Expression in each language: ST
FLDEL	16 bits	Continuous	FLDEL EN ENO s1 s2 n	FLDEL(EN,s1,s2,n);

2. Set data

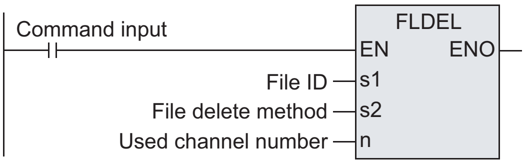
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	File ID (Refer to Detailed explanation of setting data)	ANY16
	s2	File delete method (Refer to Detailed explanation of setting data)	ANY16
	n	Used channel number [contents of setting : K1 = ch1, K2 = ch2]	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices: System user							Word Devices: Digit specification				System user				Special unit	Index			Others: Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	●				●	●	●			
s2														●	●				●	●	●			
n																				●	●			

Function and operation explanation

1. 16-bit operation (FLDEL)



The FLDEL instruction deletes files stored in the CompactFlash™ card, or formats the CompactFlash™ card in the following method.

1) Specify file deletion or file formatting using s1.
   a) When s1 is "K-1 (H0FFFF)", the FLDEL instruction deletes all files whose ID is 0 to 63.
   b) When s1 is "K0" to "K63", the FLDEL instruction deletes the file associated with the specified file ID.
   c) When s1 is "K512 (H200)", the FLDEL instruction formats the CompactFlash™ card.
2) Specify the file deletion method or format type using s2.
   a) When s1 is "K-1 (H0FFFF)" or "K0" to "K63", specify the deletion method
      K0: The FLDEL instruction deletes the specified file.
      K1: The FLDEL instruction deletes the association between the file name and the file ID (, but does not delete the file itself).
      However, when the file ID specified in s1 is "0", the FLDEL instruction deletes the file without regard to the setting of s2.
   b) When s1 is "K512 (H200)", specify the format type.
      k256(H100) : The FLDEL instruction formats the CompactFlash™ card in FAT16 format.

For details, refer to Detailed explanation of setting data.

Detailed explanation of setting data

Details of the setting data in the FLDEL instruction are as shown below.

Setting items	Description	Data Type
s1	File ID K-1(H0FFFF) : The FLDEL instruction deletes all files. K0 to K63 : The FLDEL instruction deletes a file associated with the specified file ID. K512(H200) : The FLDEL instruction formats the CompactFlash™ card.	ANY16
s2	When s1 is "K-1 (H0FFFF)" or "K0" to "K63" Specify the deletion method. K0 : The FLDEL instruction deletes the specified file. K1 : The FLDEL instruction deletes the association between the file name and the file ID (but does not delete the file itself). However, when the file ID specified in s1 is "0", the FLDEL instruction deletes the file itself without regard to the setting of s2.	ANY16
	When s1 is "K512 (H200)" Specify the format type. K256(H100) : The FLDEL instruction formats the CompactFlash™ card in the FAT16 format.	
n	Channel number used by the CF-ADP K1 : ch1 K2 : ch2	ANY16

Cautions

1) When the file ID "K0 (FIFO file)" or "K-1 (all files)" is specified, it may take approximately 1 minute to delete the files depending on the number of stored files.
2) The FX3U and FX3UC PLCs supports the instruction at V2.61 or later.
   The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

7.28.3 FLWR

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

The FLWR instruction writes data to the CompactFlash™ card or to the buffer inside the FX3U-CF-ADP.

→ **As for explanation of the instruction, see the FX3U-CF-ADP User's Manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
FLWR	16 bits	Continuous	FLWR: EN, s1, s2, s3, n → ENO, d	FLWR(EN,s1,s2,s3,n,d);

2. Set data

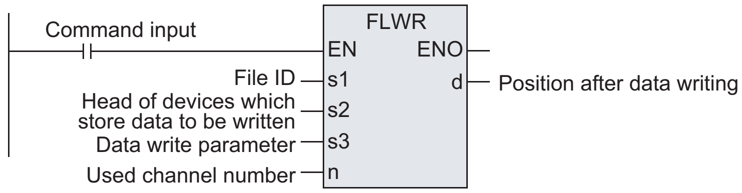
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s1	File ID (Refer to Detailed explanation of setting data)	ANY16
	s2	Head of devices which store data to be written (Refer to Detailed explanation of setting data)	ANY_SIMPLE
	s3	Data write parameter (Refer to Detailed explanation of setting data)	ANY16(0..4)
	n	Position after data writing (Refer to Detailed explanation of setting data)	ANY16
Output variable	ENO	Execution state	Bit
	d	Used channel number [contents of setting : K1 = ch1, K2 = ch2]	ANY16(0..1)

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s1														●	●				●	●	●			
s2	●	●	●			●						●	●	●	●				●					
s3														●	●				●					
d														●	●				●					
n																				●	●			

Function and operation explanation

1. 16-bit operation (FLWR)



The FLWR instruction writes data specified by the device (s2) to a file stored in the CompactFlash™ card specified by the file ID or to the buffer inside the CF-ADP. The FLWR instruction can overwrite data in the line position specified by the device (s3)+1, and can write additional data (K-1). When the writing destination is the buffer inside the CF-ADP, the FLWR instruction can only execute additional writing. When writing is completed, the line position and column position after writing are as follows.

- When data in 1 line is written additionally
  - Line position after writing : Written line position + K1
  - Column position after writing : K1
- When a line having existing data is overwritten
  - Line position after writing :
    Written line position if data is not written to the final column position of the specified the line position
    Line position next to the written line position if data is written to the final column position of the line
  - Column position after writing:
    Column position next to the final written data point K1 if data is written to the final data point in the line
    "K1" if data is written to the final data point in the line

Both additional writing and overwriting are executed to the maximum number of lines specified during file creation. If data is written up to the final column position, the line position after writing varies depending on the file type and setting.

- When the processing is stopped by the maximum line position in a normal file
  Line position after writing = Maximum line position + K1
  K-32768 when the maximum line position is "K32767"
- In the case of a normal file in which processing returns to the head of the file from the end of the file (ring buffer file)
  Line position after writing = K1
- In the case of FIFO file
  Line position after writing = K1

In either case, the column position after writing is "K1".

Detailed explanation of setting data

Details of the setting data in the FLWR instruction are as shown below.

Setting items		Description	Data Type
s1		File ID K0 to K63	ANY16
s2		Head of devices which store data to be written. Specify the head of devices which store the data to be written to the CompactFlash™ card.	ANY_SIMPLE
Data write parameter	s3	Specify the data writing type K0 : Mixed type K1 : Bit type K2 : Decimal type (16-bit) K3 : Decimal type (32-bit) K4 : Hexadecimal type (16-bit) K5 : Hexadecimal type (32-bit) K6 : Real numbers(Floating point data) Exponent expression type (32-bit) K7 : Character string (512 half-width/full-width characters maximum) K8 : Data name :Character string consisting of up to 32 half-width/full-width characters. Index, DATE TIME are added automatically.	ANY16(0..4)
	s3 +1	Specify the line position of the writing destination, or specify additional writing. Line position of the writing destination : K1 to specified maximum number of lines Additional writing : K-1	
	s3 +2	Specify the data column position in the writing destination. Column position : K1 to K254 Additional writing : K-1	
	s3 +3	Number of written data points K1 to K254	
	s3 +4	Writing destination K0 : CompactFlash™ card K1 : Buffer inside the CF-ADP	
d		Line position after writing K1 to specified maximum number of lines	ANY16(0..1)
d +1		Column position after writing K1 to K254	
d		Channel number used by the CF-ADP K1 : ch1 K2 : ch2	ANY16

Cautions

1) The FLWR instruction is completed abnormally if a CompactFlash™ card is not mounted.
2) The user should pay close attention to the number of times data is written when the writing destination is set to the CompactFlash™ card because data is written every time the FLWR instruction is executed.
   For example, if data is written to the CompactFlash™ card every one minute, data is written 100,000 times in approximately 2 months.
3) Even if the writing destination is set to the buffer inside the CF-ADP, data is written to the CompactFlash™ card in the case of overwriting.
4) The FLWR instruction writes data to the CompactFlash™ card after the internal buffer inside the CF-ADP becomes full when the writing destination is set to the buffer. Data stored in the internal buffer inside the CF-ADP is erased when a (instantaneous or long) power interruption occurs.
5) When the data type is a data name (K8), the user can specify only the head line position before writing other data. Index and DATE TIME are added automatically.
6) The FLWR instruction may require several scans to acquire data. Take proper measures such as saving acquired data in another device if data consistency is required.
7) It is necessary to set the device number in multiples of 16 when a bit device is specified in s2 and the data type is set to anything other than bit type. When a word device is specified in s2 and the data type is set to bit, the FLWR instruction acquires data to be written from the least significant bit of the specified device.
8) When s3 is "K7" or "K8", 00H, which indicates the end of the string, must be added to the end of the character string.
9) The FX3U and FX3UC PLCs supports the instruction at V2.61 or later.
   The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

7.28.4 FLRD

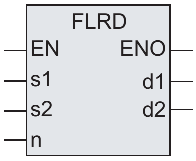
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

The FLRD instruction reads data from the CompactFlash™ card.

→ **As for explanation of the instruction, see the FX3U-CF-ADP User's Manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
FLRD	16 bits	Continuous		FLRD(EN,s1,s2,n,d1,d2);

2. Set data

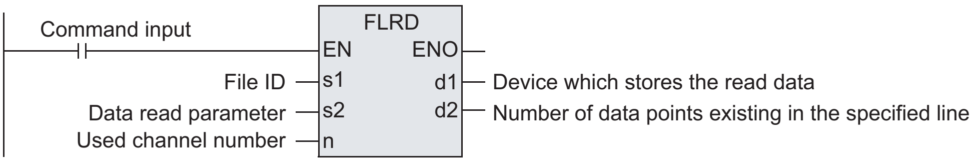
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s1)	File ID (Refer to Detailed explanation of setting data)	ANY16
	(s2)	Data read parameter (Refer to Detailed explanation of setting data)	ANY16(0..3)
	(n)	Used channel number [contents of setting : K1 = ch1, K2 = ch2]	ANY16
Output variable	ENO	Execution state	Bit
	(d1)	Device which stores the read data (Refer to Detailed explanation of setting data)	ANY_SIMPLE
	(d2)	Number of data points existing in the specified line	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s1)														●	●				●	●	●			
(s2)														●	●				●					
(d1)		●	●			●								●	●				●					
(d2)														●	●				●					
(n)																				●	●			

Function and operation explanation

1. 16-bit operation (FLRD)



The FLRD instruction reads corresponding number of data from the position determined by the line position and column position in the file specified by the file ID, and stores the read data to a device specified in (d1).

When reading data from a file in which only the same type of data exists in one line, refer to FX3U-CF-ADP User's Manual.
When reading data from a file in which different types of data exist in one line, refer to FX3U-CF-ADP User's Manual.

Detailed explanation of setting data

Details of the setting data in the FLRD instruction are as shown below.

Setting items		Description	Data Type
s1		File ID K0 to K63	ANY16
Data read parameter	s2	Specify the data reading type K0 : Mixed type K1 : Bit type K2 : Decimal type (16-bit) K3 : Decimal type (32-bit) K4 : Hexadecimal type (16-bit) K5 : Hexadecimal type (32-bit) K6 : Real numbers(Floating point data) Exponent expression type K7 : Character string (512 half-width/full-width characters maximum)	ANY16(0..3)
Data read parameter	s2 +1	Specify the line position from which data is read. Line position : K1 to specified maximum number of lines	
Data read parameter	s2 +2	Specify the column position from which data is read. Column position : K1 to K254	
Data read parameter	s2 +3	Read points K1 to K254	
d1		Device which stores the read data Specify the head of devices which store the data read from the CompactFlash™ card.	ANY_SIMPLE
d2		Number of data points existing in the specified line K1 to K254 K0 : No data	ANY16
n		Channel number used by the CF-ADP K1 : ch1 K2 : ch2	ANY16

Cautions

1) The FLRD instruction is completed abnormally if a CompactFlash™ card is not mounted.
2) The FLRD instruction may require several scans to acquire data. Use the acquired data only after confirming completion of the FLRD instruction if data consistency is required.
3) It is necessary to set the device number in a multiple of 16 when a bit device is specified in d1 and the read data type is anything other than bit. When a word device is specified in d1 and the read data type is bit, the FLRD instruction stores data read from the least significant bit of the specified word device.
4) When the data type is anything other than character string and the number of devices which store the read data is insufficient, the FLRD instruction does not read data from the CF-ADP. An error occurs.
5) When the data type is a character string, the character string length is unknown. The PLC stores as much read data as possible. When reading is not completed even after the final device is reached, an error occurs.
6) The FX3U and FX3UC PLCs supports the instruction at V2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

7.28.5 FLCMD

FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

The FLCMD instruction gives instruction for operation to the FX3U-CF-ADP.

→ **As for explanation of the instruction, see the FX3U-CF-ADP User's Manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
FLCMD	16 bits	Continuous	FLCMD block (EN, ENO, s, n)	FLCMD(EN,s,n);

2. Set data

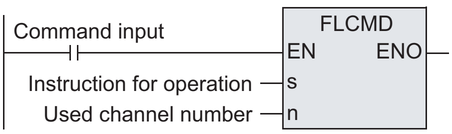
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	s	Instruction for operation (Refer to Detailed explanation of setting data)	ANY16
	n	Used channel number [contents of setting : K1 = ch1, K2 = ch2]	ANY16
Output variable	ENO	Execution state	Bit

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Cons tant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
s														●	●				●	●	●			
n																				●	●			

Function and operation explanation

1. 16-bit operation (FLCMD)



The FLCMD instruction gives instruction for operation to the CF-ADP.
The contents of instruction are as follows.

1) When s is "K-1", the FLCMD instruction forcibly writes all buffered data (stored in the buffer inside the CF-ADP) to the CompactFlash™ card.
2) When s is "K0" to "K63", the FLCMD instruction forcibly writes the buffered data of the specified file ID (stored in the buffer inside the CF-ADP) to the CompactFlash™ card.
3) When s is "K256 (H100)", the FLCMD instruction sets the CompactFlash™ card to the mounted status if it is in the unmounted status.
4) When s is "K512 (H200)", the FLCMD instruction sets the CompactFlash™ card to the unmounted status if it is in the mounted status.
5) When s is "K1280 (H500)", the FLCMD instruction clears error codes stored in the CF-ADP.

For details, refer to Detailed explanation of setting data.

Detailed explanation of setting data

Details of the setting data in the FLCMD instruction are as shown below.

Setting items	Description	Data Type
d	Contents of instruction for operation K-1 : Forcibly writes all buffered data to the CompactFlash™ card. K0 to K63 : Forcibly writes the buffered data of the specified file ID to the CompactFlash™ card. K256(H100) : Sets the CompactFlash™ card to the mounted status*1. K512(H200) : Sets the CompactFlash™ card to the unmounted status*2. K1280(H500) : Clears error codes stored in the CF-ADP.	ANY16
n	Channel number used by the CF-ADP K1 : ch1 K2 : ch2	ANY16

*1. The CompactFlash™ card is available in the "mounted" status.

*2. The CompactFlash™ card is unavailable in the "unmounted" status.

Caution

1) The FX3U and FX3UC PLCs supports the instruction at V2.61 or later.
The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

7.28.6 FLSTRD

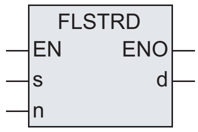
FX3U(C)	FX3G	FX2N(C)	FX1N(C)	FX1S	FXU/FX2C	FX0N	FX0(S)
△	×	×	×	×	×	×	×

Outline

The FLSTRD instruction reads the status (including the error information and file information) of the FX3U-CFADP.

→ **As for explanation of the instruction, see the FX3U-CF-ADP User's Manual.**

1. Format and operation, execution form

Instruction name	Operation	Execution form	Expression in each language	
			Structured ladder	ST
FLSTRD	16 bits	Continuous		FLSTRD(EN,s,d,n);

2. Set data

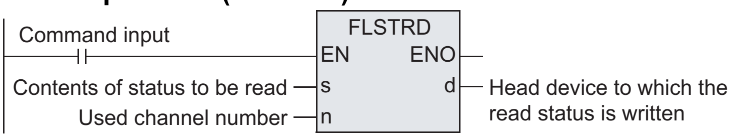
Variable		Description	Data type
Input variable	EN	Execution condition	Bit
	(s)	Contents of status to be read (Refer to Detailed explanation of setting data)	ANY16
	(n)	Used channel number [contents of setting : K1 = ch1, K2 = ch2]	ANY16
Output variable	ENO	Execution state	Bit
	(d)	Head device to which the read status is written (Refer to Detailed explanation of setting data)	ANY16

3. Applicable devices

Operand type	Bit Devices							Word Devices												Others				
	System user							Digit specification				System user				Special unit	Index			Constant		Real Number	Character String	Pointer
	X	Y	M	T	C	S	D□.b	KnX	KnY	KnM	KnS	T	C	D	R	U□\G□	V	Z	Modifier	K	H	E	"□"	P
(s)														●	●				●	●	●			
(d)														●	●				●					
(n)																				●	●			

Function and operation explanation

1. 16-bit operation (FLSTRD)



The FLSTRD instruction reads the status information of the CF-ADP. The following contents can be read.
The number of data stored in (d) varies depending on the contents of the read status.

1) When (s) is "K0" to "K63" the FLSTRD instruction reads the final line number and final column position of each file.
2) When (s) is "K256 (H100)" the FLSTRD instruction reads file IDs stored in the CompactFlash™ card.
3) When (s) is "K512 (H200)" the FLSTRD instruction reads the data capacity.
4) When (s) is "K768 (H300)" the FLSTRD instruction reads the version information of the CF-ADP.
5) When (s) is "K1024 (H400)" the FLSTRD instruction reads the error information (error flag) for errors having occurred in the CF-ADP.
6) When (s) is "K1280 (H500)" the FLSTRD instruction reads error codes and error code details. Up to 5 of the latest error codes and error code details can be stored.

For details, refer to Detailed explanation of setting data.

Detailed explanation of setting data

Details of the setting data in the FLSTRD instruction are as shown below.

Setting items	Description	Data Type
s	Contents of status to be read K0 to K63 : Final line position of each file K256(H100) : File IDs stored in the CompactFlash™ card K512(H200) : Capacity of the CompactFlash™ card K768(H300) : Version of the CF-ADP K1024(H400) : Error information (error flag) K1280(H500) : Error codes	ANY16
d	Head device to which the read status is written The number of data points stored in d varies depending on the contents of the read status.	ANY16
n	Channel number used by the CF-ADP K1 : ch1 K2 : ch2	ANY16

- When s is "K0" to "K63"
  The FLSTRD instruction reads the final line position and final column position of each file.

Setting items	Description
d	Final line position K1 to the specified maximum line position
d +1	Final column position

- When s is "K256 (H100)"
  The FLSTRD instruction reads file IDs stored in the CompactFlash™ card. For a file ID corresponding to the read data, refer to the file ID correspondence table shown below.
  When a file exists, a bit corresponding to the file ID turns ON.

Setting items	Description
d	Stores the existence of file IDs.
d +1	
d +2	
d +3	

File ID correspondence table

Setting items	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
d	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
d +1	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
d +2	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
d +3	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48

- When s is "K512 (H200)"
  The FLSTRD instruction reads the data capacity, used space and free space of the CompactFlash™ card to the following devices respectively.

Setting items	Description
d +1, d	Data capacity of the CompactFlash™ card (kB) Units, If the data capacity is less than 1 kB, "1" is stored.
d +3, d +2	Used space of the CompactFlash™ card (kB) Units, If the data size is less than 1 kB, "1" is stored.
d +5, d +4	Free space of the CompactFlash™ card (kB) Units, If the data size is less than 1 kB, "1" is stored.

- When s is "K768 (H300)"
  The FLSTRD instruction reads the version information of the CF-ADP.

Setting items	Description
d	Stores the version of CF-ADP. (Example) K100 = Ver.1.00

- When (s) is "K1024 (H400)"
  The FLSTRD instruction reads the error information (error flag).

Setting items	Description
(d)	Error detection signal b0 : The CompactFlash[TM] card is not mounted. b1 : The CompactFlash[TM] card is full. b2 : An error has occurred in the CF-ADP. b3 : CF-ADP H/W error b4 : CompactFlash[TM] card error b5 to b15 : Not used

- When (s) is "K1280 (H500)"
  The FLSTRD instruction reads the error code and error code details for errors having occurred in the CF-ADP. Up to 5 of the latest error codes and error code details can be stored.

Setting items	Description
(d)	Error code 1
(d)+1	Error code details 1
(d)+2	Error code 2
(d)+3	Error code details 2
(d)+4	Error code 3
(d)+5	Error code details 3
(d)+6	Error code 4
(d)+7	Error code details 4
(d)+8	Error code 5
(d)+9	Error code details 5

Caution

1) The FX3U and FX3UC PLCs supports the instruction at V2.61 or later.
   The FX3UC-32MT-LT-2 PLC is due to be upgraded later.

8. Interrupt Function and Pulse Catch Function

This chapter explains the built-in interrupt function and pulse catch function in FX PLCs.
The input, special devices and timers in the explanations relate to the FX3U and FX3UC PLCs. Note that these differ from one model of PLC to another.

→ **FX Structured Programming Manual (Device & Common)**

8.1 Outline

This section explains the function to immediately execute an interrupt program (interrupt routine) without being affected by the operation cycle of the sequence program (main) while using an interrupt function as a trigger.
The delay by operation cycle and machine operation affected by uneven time intervals in normal sequence program process can be improved.

1. **Input interrupt function (interrupt of external signal input (X))**
   By the input signal from an input (X000 to X005), the normal sequence program is paused, and an interrupt routine program is executed with priority.
   The input interrupt execution timing can be specified on the rising edge or falling edge of the signal by the pointer number.
   → **For details, refer to Section 8.3.**

2. **Input interrupt delay function (interrupt of external signal input (X))**
   By the input signal from an input (X000 to X005), the normal sequence program is paused, and an interrupt routine program is executed with priority after the delay time (set in units of 1 ms).
   The input interrupt execution timing can be specified on the rising edge or falling edge of the signal by the pointer number.
   → **For details, refer to Section 8.4.**

3. **Timer interrupt function (timer interrupt activated in a constant cycle)**
   The normal sequence program is paused in a constant cycle of 10 to 99 ms, and an interrupt routine program is executed with priority.
   → **For details, refer to Section 8.5.**

4. **High speed counter interrupt function (interrupt function given at counting up)**
   When the current value of a high speed counter reaches a specified value, the normal sequence program is paused and an interrupt routine program is executed with priority.
   → **For details, refer to Section 8.6.**

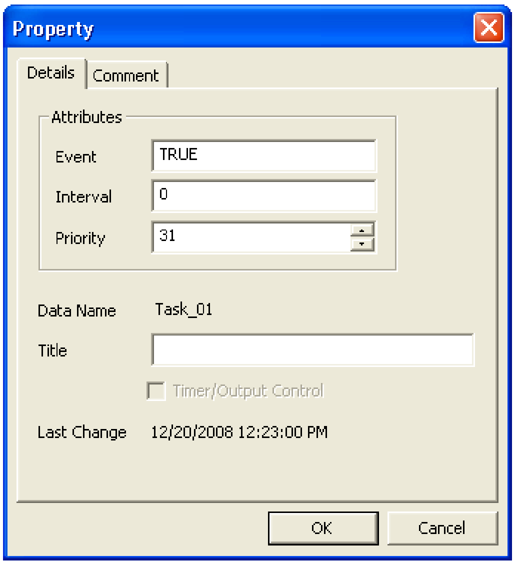
5. **Pulse catch function**
   When the input signal from an input (X000 to X007) turns ON from OFF, a special auxiliary relay ;M8170 to M8177 is set in the interrupt processing. By a relay M8170 to M8177 in a normal sequence program, a signal that remains ON longer than the receivable range with regular input processing can be easily received.
   When processing such a signal that turns ON and OFF several times in one operation cycle, however, use the input interrupt function.
   → **For details, refer to Section 8.7.**

8.2 Common items

8.2.1 Interrupt function

Three types of interrupt, namely, input interrupt, timer interrupt and counter interrupt, are available.
Observe the following in creating an interrupt program.

1) Create a task for interrupt and main program.
2) Set an interrupt pointer in the event box for the interrupt program.



For the interrupt pointers set in the event box, refer to their respective explanations.

3) The IRET instruction does not need to be programmed because, at the time of compilation, it is automatically added to the end of the program registered for the interrupt program task.

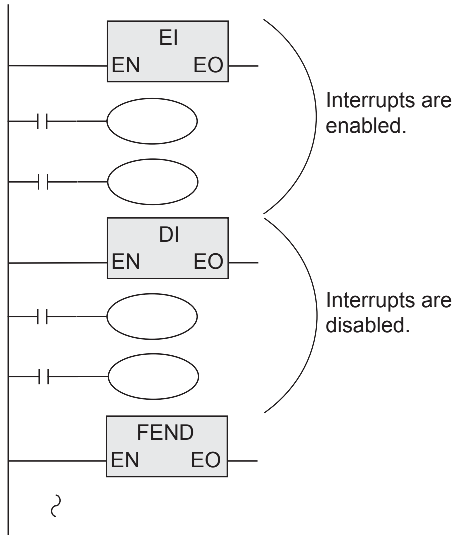
8.2.2 How to disable interrupt function and pulse catch function

This section describes how to disable the interrupt function and pulse catch function.

1. Limiting the program interrupt range [interrupt function and pulse catch function]

1) Programming method
Program the DI instruction to set the interrupt disabled zone.
Even if an interrupt is generated between the DI instruction and EI instruction (interrupt disabled zone), the interrupt is executed after the EI instruction.

2) Program example



3) Cautions
a) The interrupt inputs with special auxiliary relay for interrupt disable (M8050 to M8059) turned ON are excluded.
This special auxiliary relay is not available for pulse catch function.
b) When the disabled zone is long, interrupts are accepted, but the interrupt processing is started after considerable time.
When the interrupt disabling setting is not required, program only EI instruction. It is not always necessary to program DI instruction.

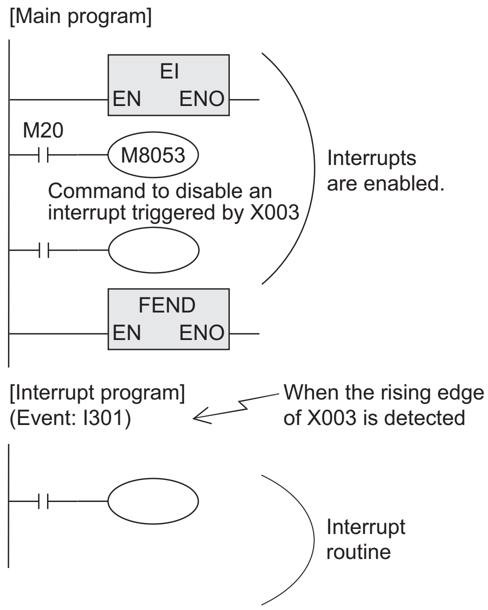
2. Disabling interrupt pointers (for each interrupt routine) [interrupt function]

1) Programming method
The special auxiliary relays M8050 to M8059 for disabling interrupt are provided.
While an interrupt disable flag (M8050 to M8059) is ON, a corresponding interrupt program is not executed even if the interrupt disable flag is set to OFF after a corresponding interrupt is generated.

Input interrupt	The input interrupts X000 to X005 correspond to M8050 to M8055*1 respectively. When a relay M8050 to M8055 turns ON, a corresponding input interrupt is disabled.
Timer interrupt	The timer interrupts 16□□ to 18□□ correspond to M8056 to M8058*1 respectively. When a relay M8056 to M8058 turns ON, a corresponding timer interrupt is disabled.
High speed counter interrupt	When M8059*1 turns ON, all of the high speed counter interrupts I010 to I060 are disabled.

*1. Cleared when the PLC mode is changed from RUN to STOP.

2) Program example
In the program example shown below, when M8053 is set to ON by M20, the interrupt input I301 triggered by X003 is disabled.



8.2.3 Related items

1. Using the I/O refresh function (REF instruction)

When controlling an input relay or output relay in an interrupt program, the I/O refresh instruction REF can be used to acquire the latest input information and immediately output the operation result. As a result, high speed control is achieved without being affected by the operation cycle of the PLC.

2. Interrupt operation while FROM/TO instruction is executed. The interrupt operation is executed as follows depending on the ON/OFF status of the special auxiliary relay M8028.

1) While M8028 is OFF
While FROM/TO instructions are being executed, interrupts are automatically disabled. Input interrupts and timer interrupts are not executed.
Interrupts generated during this period are immediately executed when the execution of FROM/TO instructions are completed.
FROM/TO instruction can be used in an interrupt program when M8028 is OFF.

2) While M8028 is ON
When an interrupt is generated while FROM/TO instruction is being executed, execution of the FROM/TO instruction is paused and the interrupt program is executed.
FROM/TO instructions cannot be used in an interrupt routine program when M8028 is ON.

8.2.4 Cautions on use (common)

This section explains common cautions on using the interrupt function or pulse catch function.
Specific cautions on each interrupt function are explained in the description of each interrupt function.

1. Processing when many interrupts are generated

When many interrupts are generated in turn, priority is given to the first one. When many interrupts are generated at the same time, priority is given to the one having the smallest pointer number.
While an interrupt routine is being executed, other interrupts are disabled.

2. When double interrupt (interrupt during another interrupt) is required [interrupt function]

Usually, interrupts are disabled in an interrupt routine (program). When EI and DI instructions are programmed in an interrupt routine, up to two interrupts can be accepted.
The FX3G, FX1S, FX1N or FX1NC PLC does not support this function.

3. Operation when a timer is used [interrupt function]

Note that counting using a general timer is disabled, even a 1 ms retentive type timer.
In an interrupt routine, use timers for routine program T192 to T199*1.

*1. The FX0, FX0S, FX0N, FX1S, FX1N or FX1NC PLC does not support the timers for routine programs.

4. Non-overlap of input [input interrupt (with or without delay function) and pulse catch function]

The inputs X000 to X007 can be used for high speed counters, input interrupts, pulse catch, SPD, ZRN, DSZR and DVIT instructions and for general purpose inputs. Make sure that input terminals do not overlap with each other.

5. Operation of devices latched in the ON status [interrupt function]

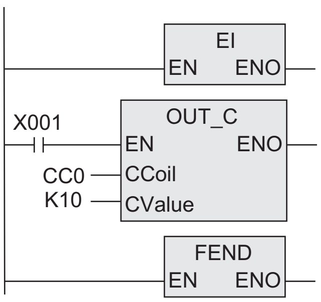
Devices which were set to ON in an interrupt routine are held in the ON status even after the interrupt routine is finished. When RST instruction for a timer or counter is executed, the reset status of the timer or counter is also held.
To turn OFF a device held in the ON status or for canceling such a timer or counter held in the reset status, reset such a device or deactivate RST instruction respectively inside or outside routine.

Example in which outputs are latched

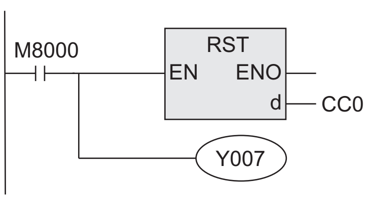
In the program example shown below, the counter C0 is provided to count X001. When X000 turns ON from OFF, the interrupt program I001 is executed only in one scan, and then the counter C0 is reset and Y007 is output.

1) Program example

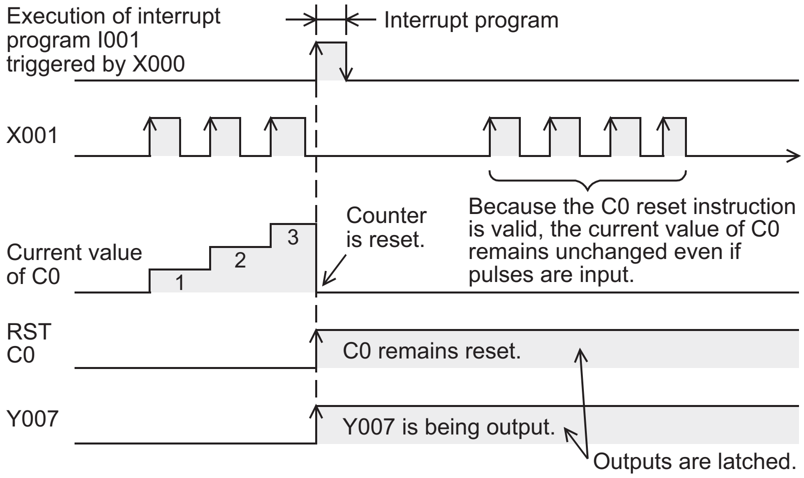
[Main program]



[Interrupt program]
(Event: I001)

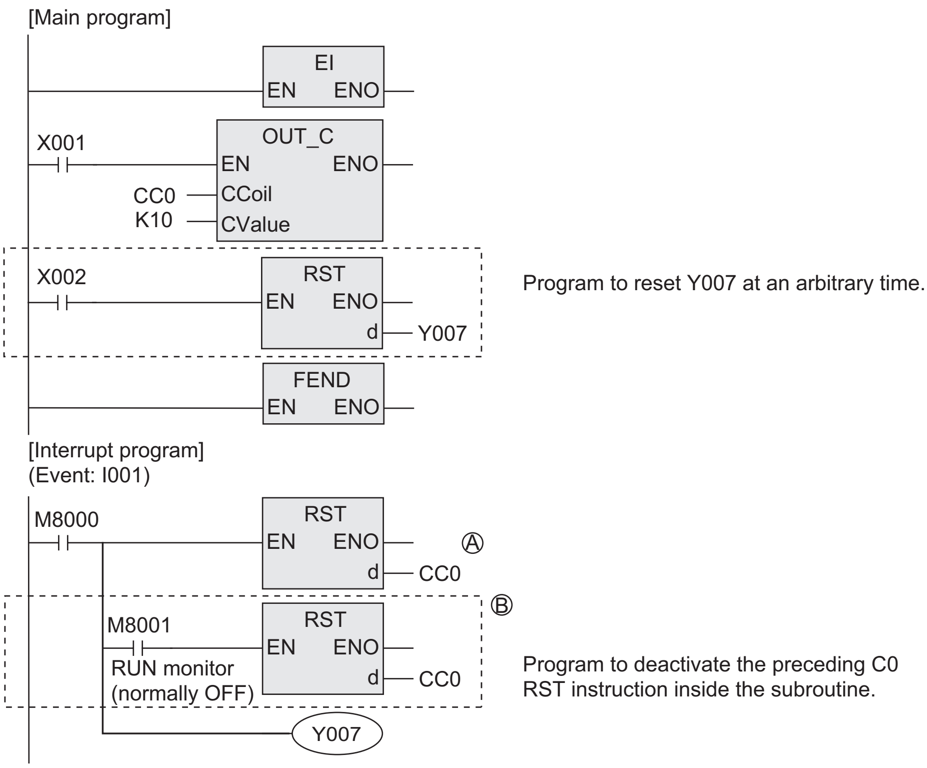


2) Timing chart

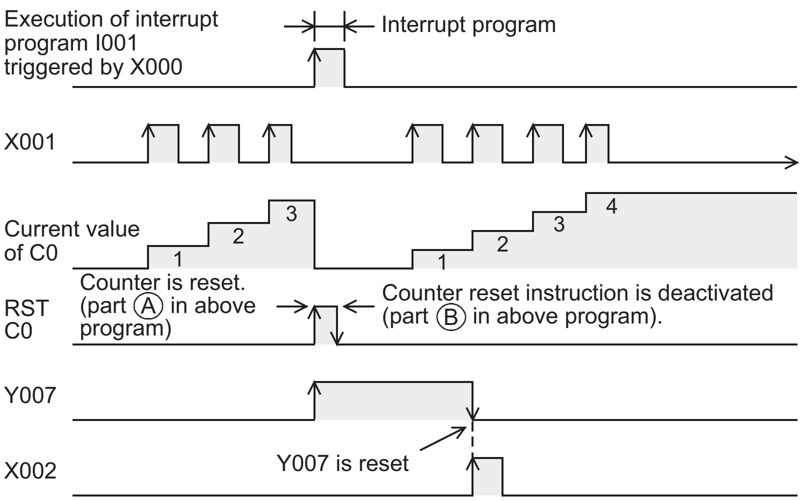


Example in which latched outputs are reset (countermeasures)

1) Program example



2) Timing chart



8.3 Input Interrupt (Interrupt Triggered by External Signal) [Without Delay Function]

8.3.1 Input Interrupt (Interrupt Triggered by External Signal) [Without Delay Function]

1. Outline

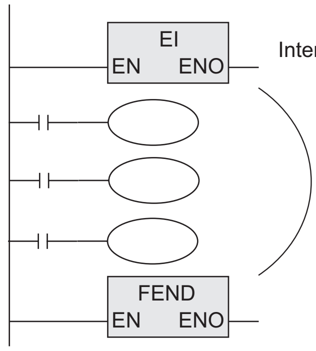
An interrupt routine is executed by the input signal from an input X000 to X005.

2. Application

Because the external input signal can be processed without being affected by the operation cycle of the PLC, this interrupt is suitable to high speed control and receiving of short pulses.

3. Basic program (programming procedure)

[Main program]



Interrupts are enabled.

Main program

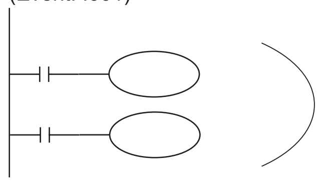
Main program
Interrupt inputs are accepted after EI instruction.
It is not necessary to program DI (disable interrupt) instruction if there is no zone where input interrupts should be disabled.

[Interrupt program]
(Event: I001)

FEND instruction finishes the main program.

[Interrupt program]
(Event: I001)

When the rising edge of X000 is detected

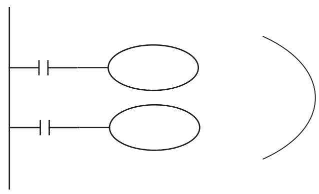


Interrupt routine 1)
(Interrupt program)

When X000 turns ON, its rising edge is detected, and the interrupt routine 1) is executed.
The program execution returns to the main program after executing routine 1).

[Interrupt program]
(Event: I100)

When the falling edge of X001 is detected



Interrupt subroutine 2)
(Interrupt program)

When X001 turns OFF, its falling edge is detected, and the interrupt routine 2) is executed.
The program execution returns to the main program after executing routine 2).

4. Number and operation of (six) interrupt pointers

I□0□

↑ 0:Interrupt at falling edge 1:Interrupt at rising edge

↑ 0 to 5 according to the inputs X000 to X005

Input number*1	Pointer number		Interrupt disable command
	Interrupt at rising edge	Interrupt at falling edge	
X000	I001	I000	M8050*2
X001	I101	I100	M8051*2
X002	I201	I200	M8052*2
X003	I301	I300	M8053*2
X004	I401	I400	M8054*2
X005	I501	I500	M8055*2

*1. The input numbers differ from one type of PLC to another.
FX0, FX0S and FX0N PLCs: supports X000 to X003 only.

*2. Cleared when the PLC mode is changed from RUN to STOP.

5. How to disable each interrupt input

When either one among M8050 to M8055 is set to ON in a program, interrupts from the corresponding input number are disabled.
(Refer to the above table for the correspondence.)

6. Cautions

1) Do not use an input two or more times.
Make sure that an input relay number used as an interrupt pointer is not used in high speed counters, pulse catch functions and pulse density instructions which use the same input range.

2) Automatic adjustment of the input filter
When an input interrupt pointer I□0□ is specified, the input filter of the input relay is automatically changed to the input filter for high speed receiving.
Accordingly, it is not necessary to change the filter value using REFF instruction or special data register D8020 (input filter adjustment).
The input filter of an input relay not being used as an input interrupt pointer operates at 10 ms (initial value).

3) Pulse width of input interrupt
For executing input interrupt by an external signal, it is necessary to input the ON or OFF signal having the duration shown in the table below or more.

For the FX3U, FX3UC and FX3G PLCs

PLC	Input number	Input filter value when "0" is set
FX3U, FX3UC	X000 to X005	5μs*1
FX3G	X000, X001, X003, X004	10μs
	X002, X005	50μs

*1. When using the input filter at the filter value of 5 μs or when receiving a pulse whose response frequency is 50 k to 100 kHz using a high speed counter, perform the following.
- Make sure that the wiring length is 5 m or less.
- Connect a bleeder resistor of 1.5 Ω (1 W or more) to an input terminal, and make sure that the load current of the open collector transistor output in the counterpart equipment is 20 mA or more including the input current in the main unit.

For the FX1S, FX1N, FX2N, FX1NC and FX2NC PLCs

PLC	Input number	Pulse width
FX1S, FX1N, FX1NC	X000, X001	10µs
	X002 to X005	50µs
FX2N, FX2NC	X000, X001	20µs
	X002 to X005	50µs

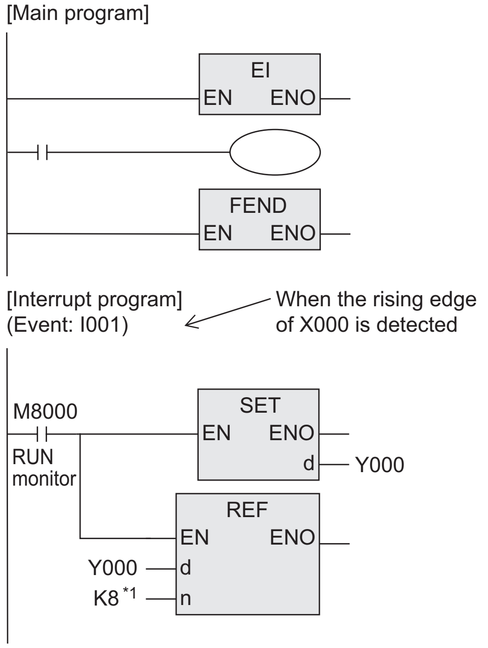
For the FX0, FX0S, FX0N, FXU and FX2C PLCs

PLC	Input number	Pulse width
FX0, FX0S, FX0N	X000 to X003	100µs
FXU, FX2C	X000 to X005	200µs

4) Using a pointer number two or more times
It is not possible to program an interrupt at the rising edge and an interrupt at the falling edge for an input such as I001 or I000.

7. Program example

1) When using both an external input interrupt at the rising edge and the output refresh (REF instruction)
   In the program example shown below, the output Y000 immediately turns ON when the rising edge of the external input X000 is detected.



Interrupts are enabled by EI instruction.
The main program is described.

The main program is finished by FEND instruction.

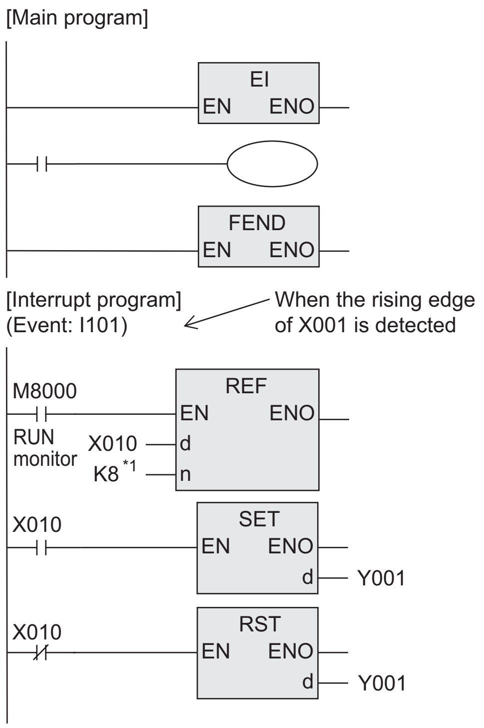
When an interrupt routine is executed by turning ON of X000, Y000 is reset unconditionally.

The outputs Y000 to Y007 are overwritten with the latest information by the output refresh instruction.
If the output refresh instruction is not provided, Y000 turns ON after END instruction after the program execution returned to the main routine.

If "SET Y000" is changed to "RST Y000", Y000 is immediately set to OFF by turning ON of X000.

*1. Be sure to specify a multiple of "8" for the number of inputs/outputs to be refreshed by REF instruction. If any value other than a multiple of "8" is specified, an operation error occurs and REF instruction is not executed.

2) When using both an input interrupt and the input refresh (REF instruction)
   In the program example shown below, an interrupt is executed using the latest input information.



Interrupts are enabled by EI instruction.
The main program is described.

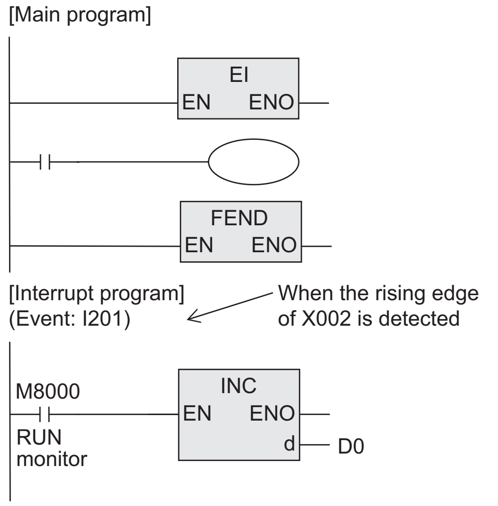
The main program is finished by FEND instruction.

When an interrupt routine is executed by turning X001 to ON, the input refresh is executed unconditionally, and the ON/OFF information of X010 to X017 at the current time is received.

Y001 is set to ON or OFF according to the ON/OFF status of X010.

*1. Be sure to specify a multiple of "8" for the number of inputs/outputs to be refreshed by REF instruction. If any value other than a multiple of "8" is specified, an operation error occurs and REF instruction is not executed.

3) When counting the number of times of input generation (in the same way as single phase high speed counter)
In the program example shown below, external inputs are counted.

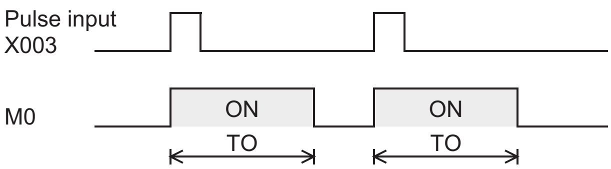


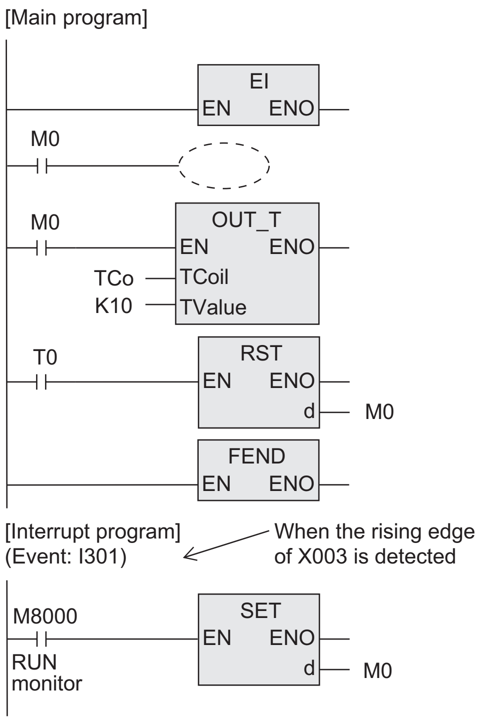
Interrupts are enabled by EI instruction.
The main program is described.

The main program is finished by FEND instruction.

When X002 turns ON, "1" is added to the value of D0.
INC instruction executes increment in every operation cycle, but the interrupt routine is executed only once by an input signal.
Accordingly, it is not necessary to use INCP (pulse operation type) instruction.

4) When catching a short pulse
In the program example shown below, the ON status is held for a certain period of time after a short pulse turns ON.





Interrupts are enabled by EI instruction.

Preparation for measurement

The period of time to hold M0 is specified.

After the timer time, M0 is reset.

The main program is finished by FEND instruction.

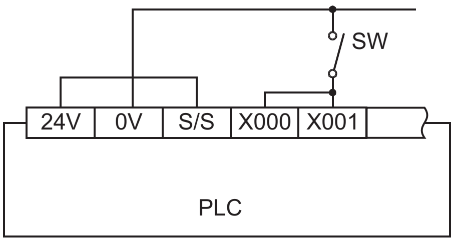
When X003 turns ON and the interrupt routine is executed, M0 is set to ON unconditionally.

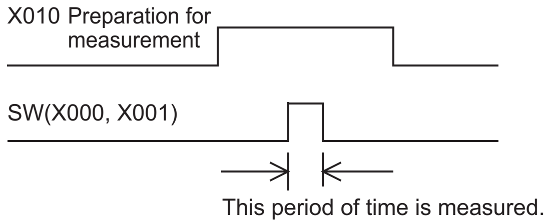
8.3.2 Examples of practical programs (programs to measure short pulse width)

By using a 1 ms retentive type timer or the special data register D8099 (high speed ring counter), the short pulse width can be measured in 1 ms or 0.1 ms units.

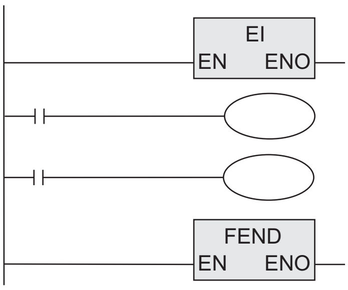
1. Example of measuring short pulse width with retentive 1 ms timer

The sequence diagram below takes the FX3U PLC (sink input) as an example.





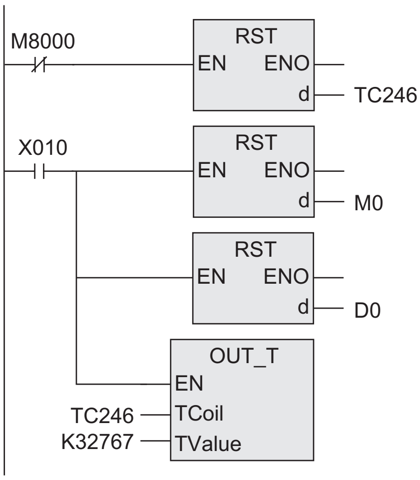
[Main program]



Interrupts are enabled by EI instruction.
The main program is described.

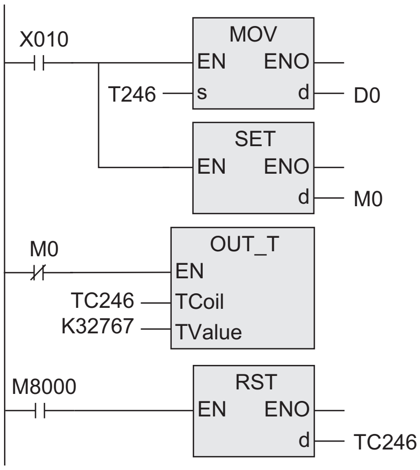
The main program is finished by FEND instruction.

[Interrupt program]
(Event: I001)
When the rising edge of X000 is detected



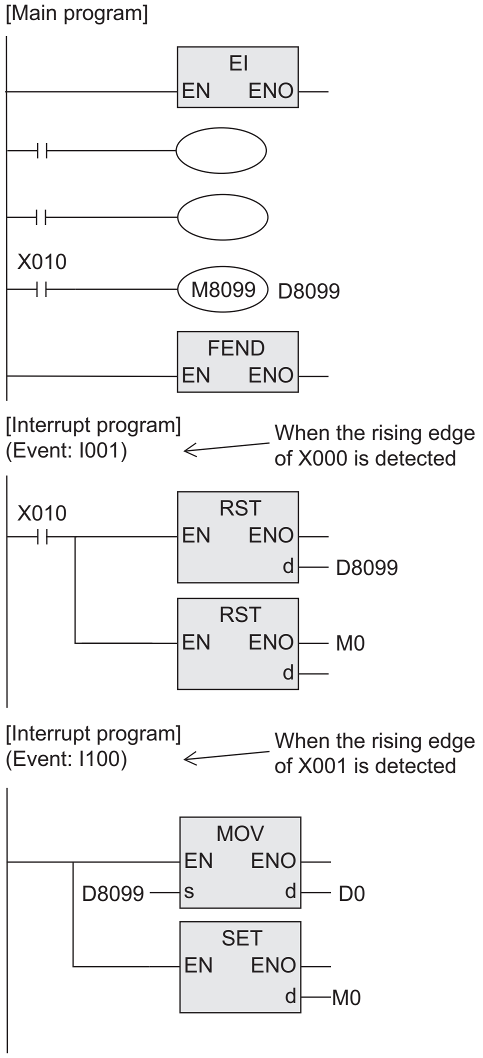
When X000 turns ON, the 1 ms timer T246 is started up by the interrupt I001.

[Interrupt program]
(Event: I100)
When the falling edge of X001 is detected



When X001 turns OFF, the current value of T246 is transferred to the data register D0 for storing the measured value by the interrupt I100, and M0 for the complete signal is set to ON.

2. Example of program to measure the short pulse width using a high speed ring counter (For FX3U and FX3UC PLCs only)



Interrupts are enabled by EI instruction.
The main program is described.

The ring counter is set to ON.

When X000 turns ON: The ring counter is reset to OFF, and measurement is started.

When X001 turns OFF: The ring counter value is transferred to D0, and measurement is completed.

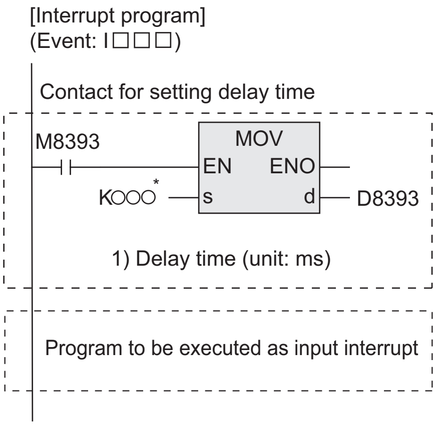
The special data register M8099 up-counts the 0.1 ms clock from the next operation cycle after being driven.
When the count value exceeds "32,767", it is returned to "0".

8.4 Input Interrupt (Interrupt by External Signal) [With Delay function]

1. Outline

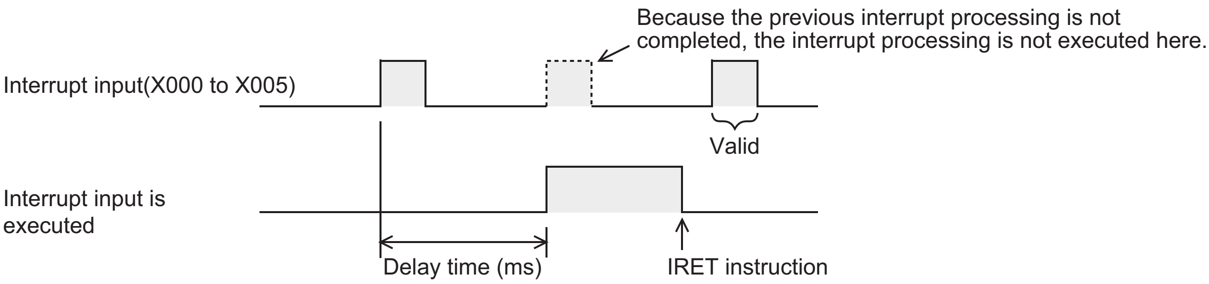
An input interrupt has the function to delay execution of an interrupt routine in units of 1 ms.
The delay time can be specified using the pattern program shown below.
By using the delay function, the mounting position of a sensor used for input interrupts can be adjusted electrically without changing the actual position.
The FX0, FX0S, FX0N, FXU, FX2C, FX1S, FX1N, FX1NC, FX2N, FX2NC or FX3G PLC does not support this function.

2. Programming procedure



- Delay time specification program
  Be sure to describe the delay time specification program shown on the left at the head of an interrupt routine program. Because this program is a pattern program, change only the delay time 1).
  Only a constant (K) or data register (D) is available for specifying the time here *.

3. Timing chart



8.5 Timer Interrupt (Interrupt in Constant Cycle)

8.5.1 Timer Interrupt (Interrupt in Constant Cycle)

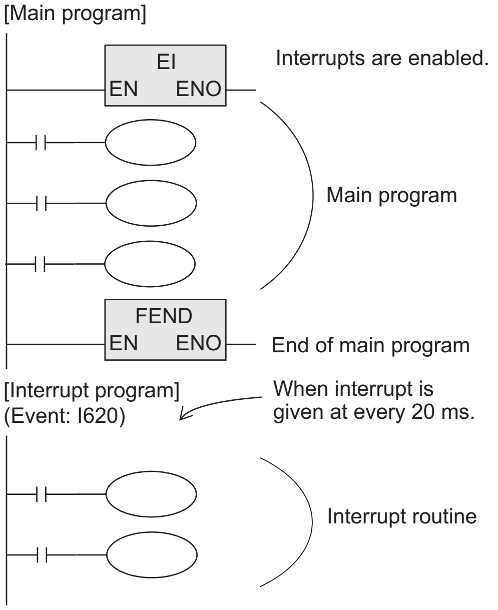
1. Outline

An interrupt routine is executed at every 10 to 99 ms without being affected by the operation cycle of a PLC.
The FX0, FX0S, FX0N, FX1S, FX1N or FX1NC PLC does not support the timer interrupt function.

2. Application

This type of interrupt is suitable when a certain program should be executed at high speed while the main program operation time is long or when a program should be executed at a constant time interval in sequence operations.

3. Basic program (programming procedure)

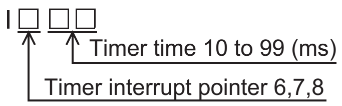


Main program
Timer interrupts are enabled after EI instruction.
It is not necessary to program DI (disable interrupt) if there is no zone where timer interrupts should be disabled.

FEND instruction indicates the end of the main program.
Be sure to describe an interrupt routine after FEND instruction.

Interrupt program
The interrupt routine is executed at every 20 ms.
Create a program to be executed as interrupt.
IRET instruction returns the program execution to the main program.

4. Number and operation of (three) timer interrupt pointers



An interrupt program is executed at every specified interrupt cycle time (10 to 99 ms). Use this type of interrupt in control requiring cyclic interrupt processing regardless of the operation cycle of a PLC.

Input number	Interrupt cycle (ms)	Interrupt disable flag
I6□□	An integer in the range from 10 to 99 is put in "□□" in the pointer name. Example: "I610" indicates a timer interrupt at every 10 ms.	M8056*1
I7□□		M8057*1
I8□□		M8058*1

*1. Cleared when the PLC mode is changed from RUN to STOP.

Caution

If the timer interrupt time is set to 9 ms or less, the timer interrupt processing may not be executed in an accurate cycle in the following cases. Therefore, using a time that is over 10 ms is recommended.

- When the interrupt program processing time is long.
- When the main program contains an applied instruction whose processing time is long.

5. Cautions

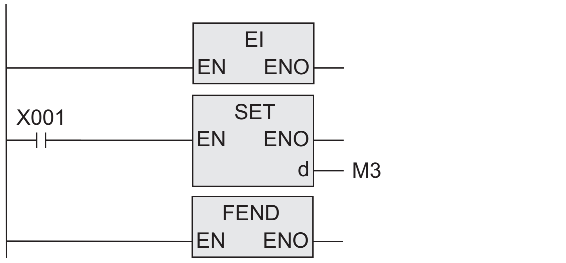
- Each pointer number (I6, I7 or I8) can be used only once.
- When M8056 to M8058 is set to ON in a program, a corresponding timer interrupt is disabled.

6. Program example

In the program example shown below, data is added and addition result is compared with the set value at every 10 ms.

1) Program example

[Main program]



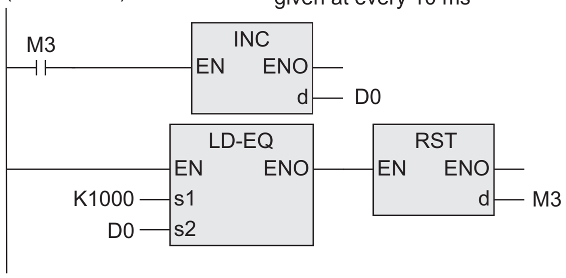
Interrupts are enabled by EI instruction.
The main program is described.

When M3 is set to ON, INC instruction becomes valid.

The main program is finished by FEND instruction.

[Interrupt program]
(Event: I610)

When interrupt is given at every 10 ms



"1" is added to the current value of D0 at every 10 ms.

When the current value of D0 reaches "1000", M3 is reset.

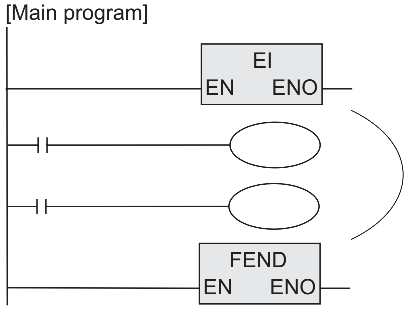
The current value of D0 is ramp data which changes from "0" to "1000" in 10 seconds.
In the program example using RAMP instruction shown later, the ramp data is made using a dedicated instruction.

8.5.2 Example of practical program (timer interrupt program using instruction)

RAMP, HKY, SEGL, ARWS and PR instructions execute a series of operations in synchronization with the scan time.
Because the total time may be too long or time fluctuation may cause a problem in these instructions, it is recommended to execute these instructions at a constant time interval using the timer interrupt function.
When not using the timer interrupt function, use the constant scan mode.

1. Timer interrupt processing of HKY instruction

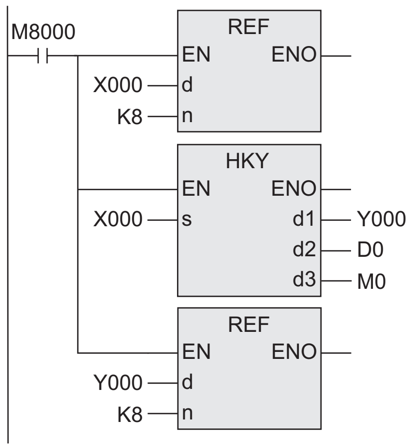
[Main program]



Interrupts are enabled by EI instruction.
The main program is described.

The main program is finished by FEND instruction.

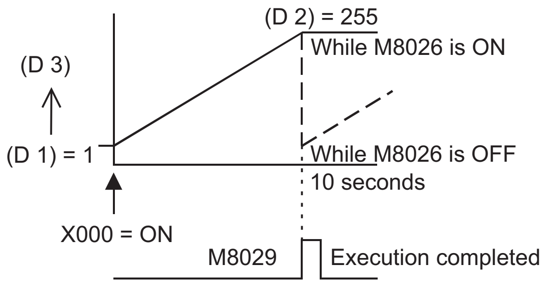
[Interrupt program]
(Event: I620)



2. Timer interrupt processing of RAMP instruction

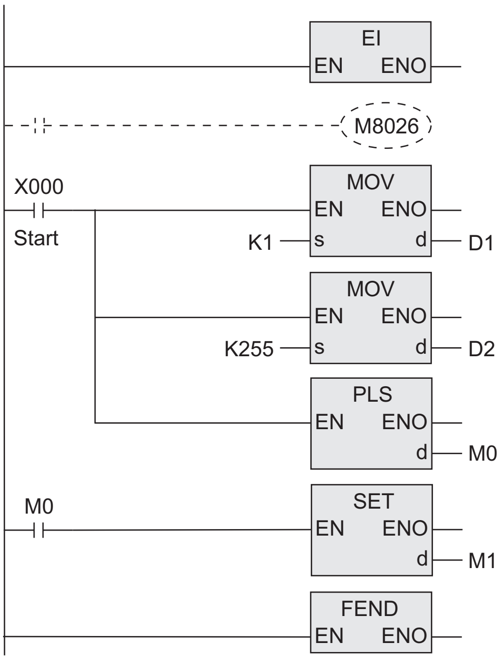
The ramp signal output circuit shown below is programmed using the timer interrupt function executed every 10 ms.

1) Ramp output pattern
D4 is occupied as a register for counting the number of times of execution.



2) Program

[Main program]



Interrupts are enabled by EI instruction.
The main program is described.

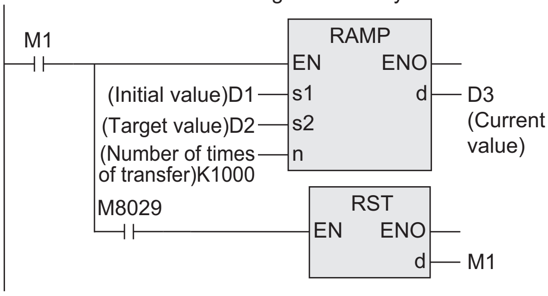
With M8026 turned ON, when the value of (D3) reaches the final value (D2), the value of Y is latched.

As soon as the start command is given, the initial value (D1) and target value (D2) are transferred.

The main program is finished by FEND instruction.

[Interrupt program]
(Event: I610)

When interrupt is given at every 10 ms



While the instruction is executed 1000 times (in 10 seconds), the contents of D3 are changed from the value of D1 to the value of D2
.

When the instruction execution complete flag M8029 turns ON, RAMP instruction drive input is set to OFF.
If RAMP instruction is continuously executed while M8026 is OFF, the value of D3 returns to the initial value (D1) immediately after it reaches the final value (D2), and then the same operation is repeated.
This program is not necessary when M8026 is ON.

8.6 Counter Interrupt - Interrupt Triggered by Counting Up of High Speed Counter

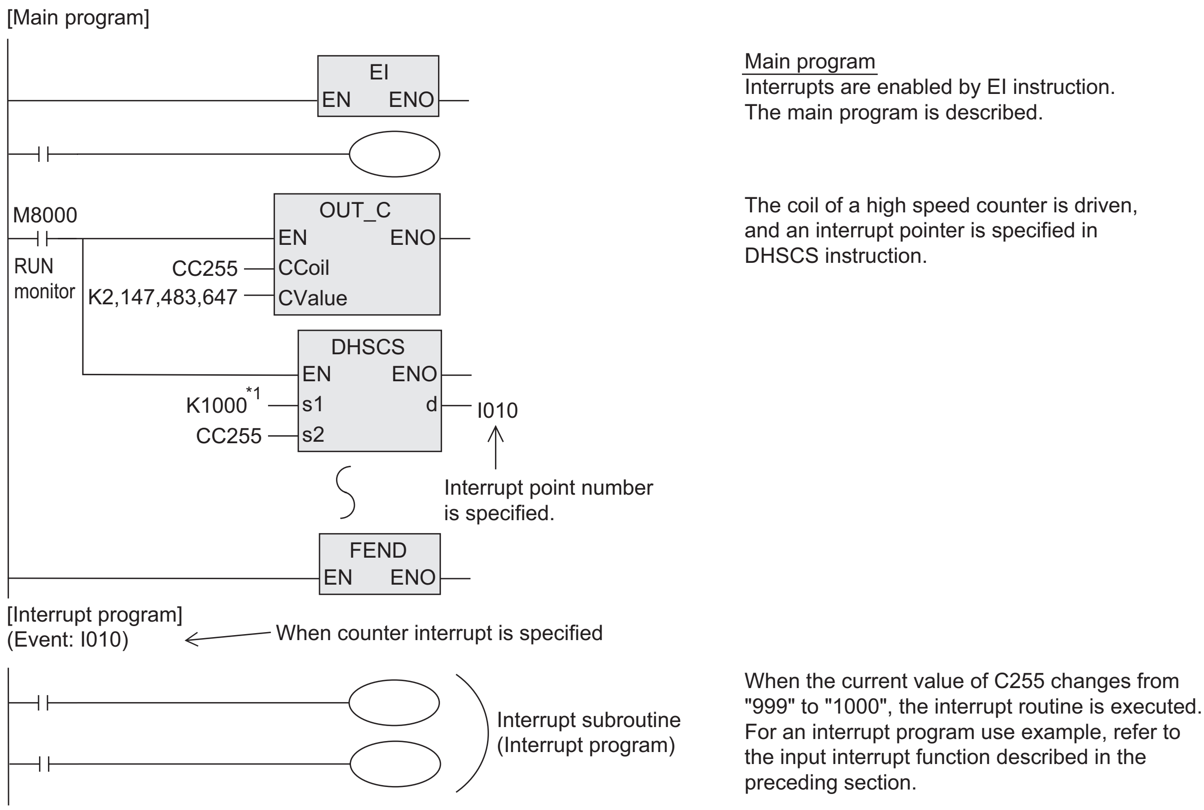
1. Outline

This type of interrupt utilizes the current value of a high speed counter.
The FX0, FX0S, FX0N, FX1S, FX1N, FX1NC or FX3G PLC does not support the counter interrupt function.
The FX2C PLC of V 3.07 or later supports the counter interrupt function.

2. Application

This type of interrupt is used together with the comparison set instruction DHSCS. When the current value of a high speed counter reaches the specified value, an interrupt routine is executed.

3. Basic program (programming procedure)



*1. When the comparison value specified by a data register, etc. is changed, the current value is actually changed to the specified value when END instruction is executed.

4. Number and operation of (six) counter interrupt pointers

I0□0
↑ Counter interrupt pointer (1 to 6)

Pointer No.	Interrupt disable flag
I010	M8059*1
I020	
I030	
I040	
I050	
I060	

*1. Cleared when the PLC mode is changed from RUN to STOP.

5. When setting an interrupt output (Y or M) to ON or OFF using a high speed counter

When only controlling the ON/OFF status of an output relay (Y) or auxiliary relay (M) according to the current value of a high speed counter, a required program can be easily created using DHSCS, DHSCR or DHSZ instruction.

6. Cautions

1) Pointer number
Pointer numbers cannot overlap with each other.

2) Disabling interrupts
When the special auxiliary relay M8059 is set to ON in a program, all counter interrupts are disabled.

8.7 Pulse Catch Function[M8170 to M8177]

When the input relay X000 to X007 turns ON from OFF after the EI instruction is executed, the special auxiliary relay M8170 to M8177 is set for interrupt processing.
The FX0, FX0S, FX0N, FX1S, FX1N or FX3G PLC does not require the EI instruction.

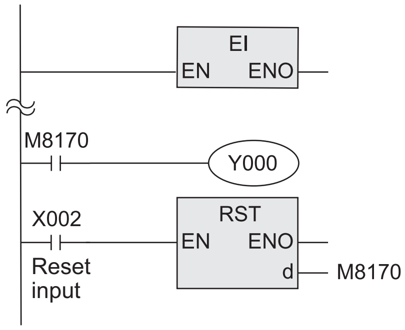
1. Assignment of input numbers and special auxiliary relays

Pulse catch input*1	Pulse catch relay
X000	M8170*2 (M8065*2 in the FX0, FX0S and FX0N PLCs)
X001	M8171*2 (M8057*2 in the FX0, FX0S and FX0N PLCs)
X002	M8172*2 (M8058*2 in the FX0, FX0S and FX0N PLCs)
X003	M8173*2 (M8059*2 in the FX0, FX0S and FX0N PLCs)
X004	M8174*2
X005	M8175*2
X006	M8176*2
X007	M8177*2

*1. Differs from one PLC to another.
FXU, FX2C, FX1S, FX1N, FX2N, FX1NC, FX2NC, FX3G : Supports X000 to X005 only.
FX0, FX0S, FX0N : Supports X000 to X003 only.

*2. Cleared when the PLC mode is changed from STOP to RUN.

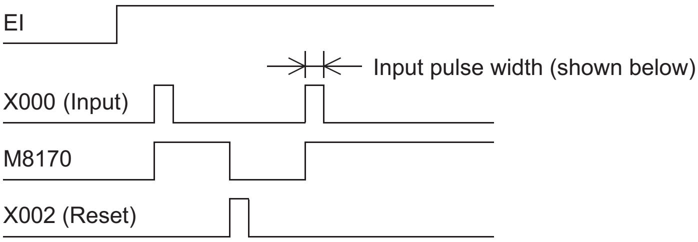
2. Program example



The FX0, FX0S, FX0N, FX1S or FX3G PLC does not require the EI instruction.

When the rising edge of X000 is detected, M8170 is set as interrupt.

The pulse catch result is reset.



- FX3U and FX3UC PLCs
  X000 to X005: 5µS or more[*1]
  X006, X007 : 50µS or more
- FX3G PLCs
  X000, X001, X003, X004: 10µS or more
  X002, X005 : 50µS or more
- FX1S, FX1N, FX1NC, FX2N and FX2NC PLCs
  X000,X001 : 10µS or more (FX1S,FX1N,FX1NC)
  : 20µS or more (FX2N,FX2NC)
  X002 to X005: 50µS or more
- FXU and FX2C PLCs
  X000 to X005: 50µS or more
- FX0, FX0S and FX0N PLCs
  X000 X003: 50µS or more

*1. When using the pulse catch function at 5µs or when receiving a pulse whose response frequency is 50 kHz to 100 kHz using a high speed counter, perform the following:
- Make sure that the wiring length is 5 m or less.
- Connect a bleeder resistor of 1.5 Ω (1 W or more) to an input terminal, and make sure that the load current of the open collector transistor output in the counterpart equipment is 20 mA or more including the input current in the main unit.

3. Cautions on use

1) When receiving an input again, it is necessary to reset the device which was once set using a program. Accordingly, until a device is reset, a new input cannot be received.
2) When it is necessary to receive continuous short pulses (input signals), use the external input interrupt function or high speed counter function.
3) A filter adjustment program is not required.
4) The pulse catch function is executed regardless of the operations of the special auxiliary relays M8050 to M8055 for respectively disabling interrupts.

8.8 Pulse width/Pulse period measurement function [M8075 to M8083, D8074 to D8097]

This function is supported only in FX3G PLC Ver.1.10 or later.

The pulse width/pulse period measurement function stores the values of 1/6 μs ring counters at the input signal rising edge and falling edge to special data registers. This function also divides by "60" the difference in the counter value (pulse width) between the rising edge and the falling edge or the difference in the counter value (pulse period) between the previous rising edge and the current rising edge, and stores the obtained pulse width or pulse period in units of 10 μs to special data registers.

The pulse width/pulse period measurement function becomes valid when a program is described using M8075 as a contact. Specify the pulse width measurement flag in the subsequent OUT instruction, and set an input terminal to be used.
When the pulse width/pulse period measurement function is valid, it always operates while the PLC mode is RUN.

Assignment of special auxiliary relays and special data registers

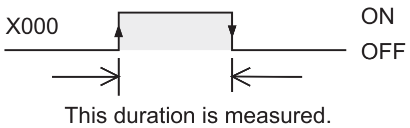
Pulse input	Pulse width/ Pulse period measurement flag	Pulse period measurement mode	Ring counter value for rising edge*1 [Unit: 1/6 μs]	Ring counter value for falling edge*1 [Unit: 1/6 μs]	Pulse width /Pulse period*1*2 [Unit: 10 μs]
X000	M8076	M8080	D8075, D8074	D8077, D8076	D8079, D8078
X001	M8077	M8081	D8081, D8080	D8083, D8082	D8085, D8084
X003	M8078	M8082	D8087, D8086	D8089, D8088	D8091, D8090
X004	M8079	M8083	D8093, D8092	D8095, D8094	D8097, D8096

*1. Cleared when the PLC mode switches from STOP to RUN.

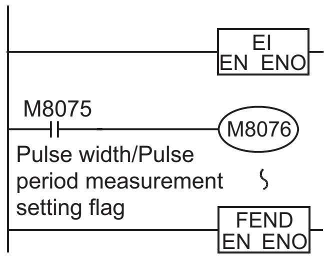
*2. The measurable pulse width is 10 μs minimum and 100 s maximum.
The measurable pulse period is 20 μs minimum.

1. Program example

1) Pulse width measurement
The pulse width of the input signal from X000 is measured.

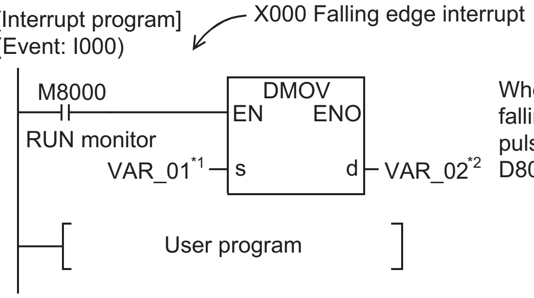


[Main program]



X000 is used for the pulse width/pulse period measurement function.

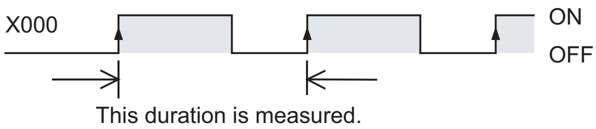
[Interrupt program]
(Event: I000)



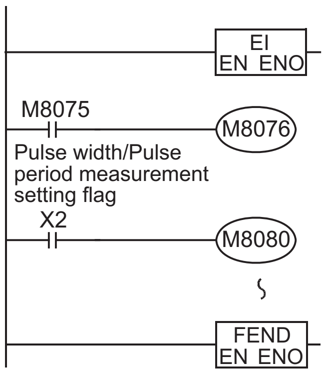
When the interrupt routine is executed at the falling edge of the input signal from X000, the pulse width of input signal from X000 stored in D8078 and D8079 is transferred to D1 and D0.

*1. VAR_01 is a global label and is defined as D8078.
*2. VAR_02 is a global label and is defined as D0.

2) Pulse period measurement
The pulse period of the input signal from X000 is measured.



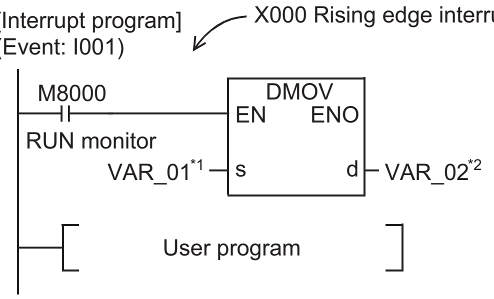
[Main program]



X000 is used for the pulse width/pulse period measurement function.

When X002 turns ON, the pulse period measurement mode is actuated.

[Interrupt program]
(Event: I001)

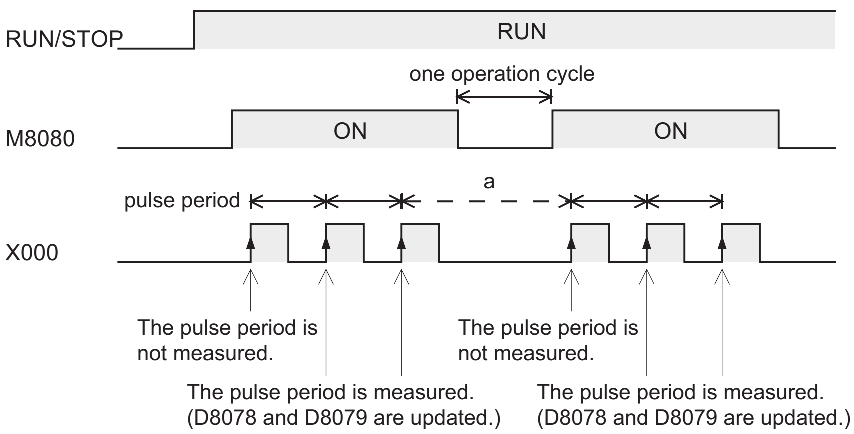


When the interrupt routine is executed at the rising edge of the input signal from X000, the pulse period of input signal from X000 stored in D8078 and D8079 is transferred to D1 and D0.

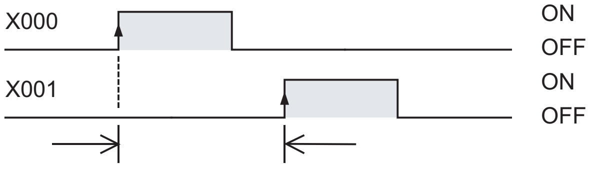
*1. VAR_01 is a global label and is defined as D8078.
*2. VAR_02 is a global label and is defined as D0.

- Timing chart
  The pulse period is not measured when the input signal rises for the first time after the PLC mode is changed from STOP to RUN, or when the input signal rises for the first time after the pulse period measurement mode (M8080) is set to ON from OFF. (Accordingly, D8078 and D8079 are not updated.)
  The pulse period is measured when the input signal rises at the next time. (As a result, D8078 and D8079 are updated.)

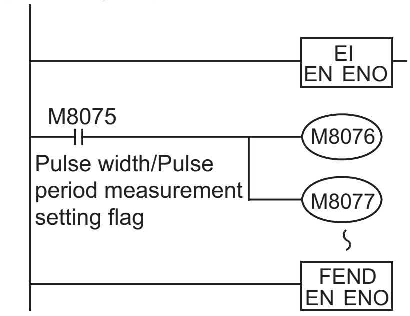
  Make the pulse width/pulse period measurement setting flag (M8080) remain OFF for 1 operation cycle or more when discontinuing the pulse input.
  If M8080 does not remain OFF for 1 operation cycle or more, the "a" period shown below is stored as the pulse period.



3) Signal delay time measurement
The delay time from the rising edge of the input signal from X000 to the rising edge of the input signal from X001 is measured.

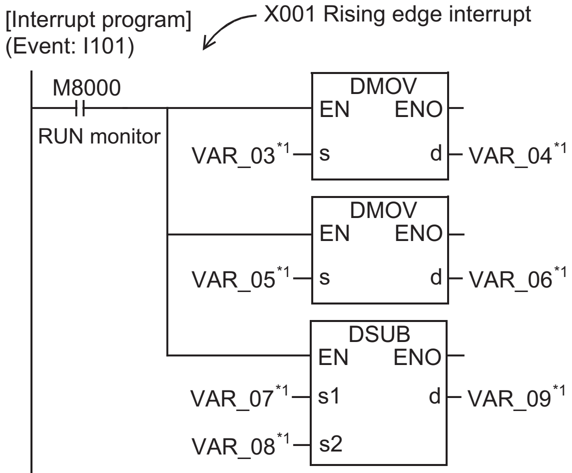


[Main program]



X000 is used for the pulse width/pulse period measurement function.

X001 is used for the pulse width/pulse period measurement function.



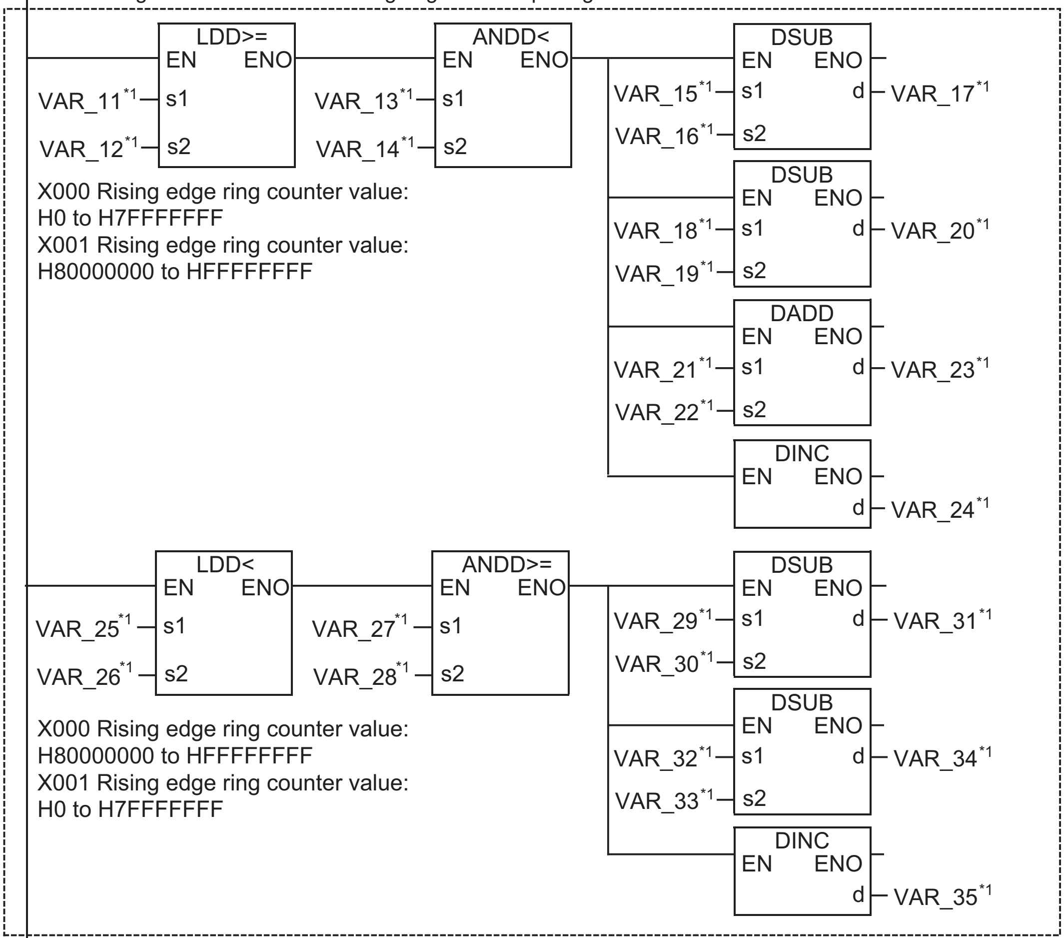
The interrupt routine is executed at the rising edge of the input signal from X001.

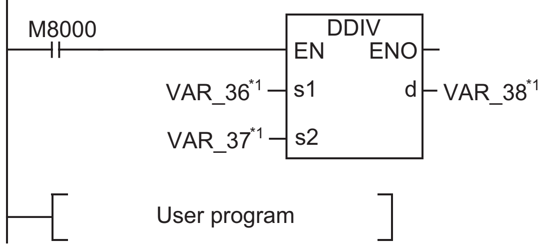
The ring counter value at the rising edge of the input signal from X000 stored in D8074 and D8075 is transferred to D1 and D0.

The ring counter value at the rising edge of the input signal from X001 stored in D8080 and D8081 is transferred to D3 and D2.

The value "Ring counter value at the rising edge of the input signal from X001 - Ring counter value at the rising edge of the input signal from X000" is stored in D9 and D8.

When either one between the ring counter value at the rising edge of the input signal from X000 and the ring counter value at the rising edge of the input signal from X001 is located within the range from H80000000 to HFFFFFFFF, the following process is executed, and the value "Ring counter value at the rising edge of the input signal from X001 - Ring counter value at the rising edge of the input signal from X000" is stored in D9 and D8*2.





The obtained value is converted into a value in units of 10 μs.

*1. VAR_01 to VAR_38 is a global label and define as following.

Global label	Defined device
VAR_03	D8074
VAR_04	D0
VAR_05	D8080
VAR_06	D2
VAR_07	D2
VAR_08	D0
VAR_09	D8
VAR_11	D0
VAR_12	K0
VAR_13	D2
VAR_14	K0
VAR_15	H7FFFFFFF
VAR_16	D0
VAR_17	D4
VAR_18	D2
VAR_19	H80000000
VAR_20	D6
VAR_21	D4
VAR_22	D6
VAR_23	D8
VAR_24	D8
VAR_25	D0
VAR_26	K0
VAR_27	D2
VAR_28	K0
VAR_29	HFFFFFFFF
VAR_30	D0
VAR_31	D4
VAR_32	D2
VAR_33	D4
VAR_34	D8
VAR_35	D8
VAR_36	D8
VAR_37	K60
VAR_38	D10

*2. The ring counter offers 32-bit data including the most significant bit.
The DSUB instruction does not give a correct value because it handles the most significant bit as the sign bit. To obtain a correct value, add the processing inside the dotted frame.

2. Cautions on use

- The pulse width/pulse period measurement function and input interrupts can be used at the same time in a same input terminal.
- When a same input terminal is used by the pulse width/pulse period measurement function and the SPD, DSZR or ZRN instruction, an operation error occurs when the instruction is executed.
- The input terminal used for the pulse width/pulse period measurement function cannot be used for the pulse catch function.
- When a same input terminal is used by the pulse width/pulse period measurement function and a high speed counter, a grammatical error occurs.
- Make sure that the total frequency of four input channels is 50 kHz or less when using the pulse width/ pulse period measurement function.
- When the pulse width/pulse period measurement function and a high speed counter are used together, the overall frequency of the high speed counter is affected.

→ **FX Structured Programming Manual (Device & Common)**

Appendix A: Relationships between devices and addresses

The table below shows the relationships between devices and addresses.

Device			Device and address		Example of corresponding device and address	
			Device	Address	Device	Address
Input relay		X	Xn	%IXn	X367	%IX247
Output relay		Y	Yn	%QXn	Y367	%QX247
Auxiliary relay		M	Mn	%MX0.n	M499	%MX0.499
Timer	Contact	TS	Tn	%MX3.n	TS191	%MX3.191
Timer	Coil	TC	Tn	%MX5.n	TC191	%MX5.191
Timer	Current value	TN	Tn	%MW3.n %MD3.n	TN191 T190	%MW3.191 %MD3.190
Counter	Contact	CS	Cn	%MX4.n	CS99	%MX4.99
Counter	Coil	CC	Cn	%MX6.n	CC99	%MX6.99
Counter	Current value	CN	Cn	%MW4.n %MD4.n	CN99 C98	%MW4.99 %MD4.98
Data register		D	Dn	%MW0.n %MD0.n	D199 D198	%MW0.199 %MD0.198
Intelligent function unit device		G	Ux\Gn	%MW14.x.n %MD14.x.n	U0\G09	%MW14.0.10 %MD14.0.9
Extension register		R	Rn	%MW2.n %MD2.n	R32767 R32766	%MW2.32767 %MD2.32766
Extension file register		ER	ERn	Not corresponding	-	-
Pointer		P	Pn	" "(blank)	P4095	Not corresponding
Interrupt pointer		I	In	Not corresponding	-	-
Nesting		N	Nn	Not corresponding	-	-
Index register		Z	Zn	%MW7.n %MD7.n	Z7 Z6	%MW7.7 %MD7.6
Index register		V	Vn	%MV6.n	V7	%MW6.7
State		S	Sn	%MX2.n	S4095	%MX2.4095

MEMO

Warranty

Please confirm the following product warranty details before using this product.

1. Gratis Warranty Term and Gratis Warranty Range

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company. However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place. Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.
2) Even within the gratis warranty term, repairs shall be charged for in the following cases.
   a) Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
   b) Failure caused by unapproved modifications, etc., to the product by the user.
   c) When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
   d) Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
   e) Relay failure or output contact failure caused by usage beyond the specified Life of contact (cycles).
   f) Failure caused by external irresistible forces such as fires or abnormal voltages, and failure caused by force majeure such as earthquakes, lightning, wind and water damage.
   g) Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
   h) Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

2. Onerous repair term after discontinuation of production

1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued.
   Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.
2) Product supply (including repair parts) is not available after production is discontinued.

3. Overseas service

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

4. Exclusion of loss in opportunity and secondary loss from warranty liability

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation of damages caused by any cause found not to be the responsibility of Mitsubishi, loss in opportunity, lost profits incurred to the user or third person by Failures of Mitsubishi products, special damages and secondary damages whether foreseeable or not , compensation for accidents, and compensation for damages to products other than Mitsubishi products, replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

5. Changes in product specifications

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

6. Product application

1) In using the Mitsubishi MELSEC programmable logic controller, the usage conditions shall be that the application will not lead to a major accident even if any problem or fault should occur in the programmable logic controller device, and that backup and fail-safe functions are systematically provided outside of the device for any problem or fault.
2) The Mitsubishi programmable logic controller has been designed and manufactured for applications in general industries, etc. Thus, applications in which the public could be affected such as in nuclear power plants and other power plants operated by respective power companies, and applications in which a special quality assurance system is required, such as for Railway companies or Public service purposes shall be excluded from the programmable logic controller applications.
   In addition, applications in which human life or property that could be greatly affected, such as in aircraft, medical applications, incineration and fuel devices, manned transportation, equipment for recreation and amusement, and safety devices, shall also be excluded from the programmable logic controller range of applications.
   However, in certain cases, some applications may be possible, providing the user consults their local Mitsubishi representative outlining the special requirements of the project, and providing that all parties concerned agree to the special circumstances, solely at the users discretion.

Revised History

Date created	Revision	Description
1/2009	A	First edition
7/2009	B	• Instructions are added: INV, MEP, MEF, RS,FLCRT, FLDEL, FLWR, FLRD, FLCMD, FLSTRD • The following instructions are provided in the FX3G series. RD3A, RD3AP, WR3A, WR3AP

MEMO

FXCPU

Structured Programming Manual [Basic & Applied Instruction]

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE: TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
HIMEJI WORKS: 840, CHIYODA CHO, HIMEJI, JAPAN

MODEL	FX-KP-SM-E
MODEL CODE	09R926

JY997D34701B
(MEE)

Effective Jul. 2009
Specifications are subject to change without notice.

MITSUBISHI ELECTRIC

HEADQUARTERS

MITSUBISHI ELECTRIC EUROPE B.V. **EUROPE**
German Branch
Gothaer Straße 8
D-40880 Ratingen
Phone: +49 (0)2102 / 486-0
Fax: +49 (0)2102 / 486-1120

MITSUBISHI ELECTRIC EUROPE B.V.-org.sl. **CZECH REP.**
Czech Branch
Avenir Business Park, Radlická 714/113a
CZ-158 00 Praha 5
Phone: +420 - 251 551 470
Fax: +420 - 251-551-471

MITSUBISHI ELECTRIC EUROPE B.V. **FRANCE**
French Branch
25, Boulevard des Bouvets
F-92741 Nanterre Cedex
Phone: +33 (0)1 / 55 68 55 68
Fax: +33 (0)1 / 55 68 57 57

MITSUBISHI ELECTRIC EUROPE B.V. **IRELAND**
Irish Branch
Westgate Business Park, Ballymount
IRL-Dublin 24
Phone: +353 (0)1 4198800
Fax: +353 (0)1 4198890

MITSUBISHI ELECTRIC EUROPE B.V. **ITALY**
Italian Branch
Viale Colleoni 7
I-20041 Agrate Brianza (MB)
Phone: +39 039 / 60 53 1
Fax: +39 039 / 60 53 312

MITSUBISHI ELECTRIC EUROPE B.V. **POLAND**
Poland Branch
Krakowska 50
PL-32-083 Balice
Phone: +48 (0)12 / 630 47 00
Fax: +48 (0)12 / 630 47 01

MITSUBISHI ELECTRIC EUROPE B.V. **RUSSIA**
52, bld. 3 Kosmodamianskaya nab 8 floor
RU-115054 Moscow
Phone: +7 495 721-2070
Fax: +7 495 721-2071

MITSUBISHI ELECTRIC EUROPE B.V. **SPAIN**
Spanish Branch
Carretera de Rubí 76-80
E-08190 Sant Cugat del Vallés (Barcelona)
Phone: 902 131121 // +34 935653131
Fax: +34 935891579

MITSUBISHI ELECTRIC EUROPE B.V. **UK**
UK Branch
Travellers Lane
UK-Hatfield, Herts. AL10 8XB
Phone: +44 (0)1707 / 27 61 00
Fax: +44 (0)1707 / 27 86 95

MITSUBISHI ELECTRIC CORPORATION **JAPAN**
Office Tower "Z" 14 F
8-12,1 chome, Harumi Chuo-Ku
Tokyo 104-6212
Phone: +81 3 622 160 60
Fax: +81 3 622 160 75

MITSUBISHI ELECTRIC AUTOMATION, Inc. **USA**
500 Corporate Woods Parkway
Vernon Hills, IL 60061
Phone: +1 847 478 21 00
Fax: +1 847 478 22 53

EUROPEAN REPRESENTATIVES

GEVA **AUSTRIA**
Wiener Straße 89
AT-2500 Baden
Phone: +43 (0)2252 / 85 55 20
Fax: +43 (0)2252 / 488 60

TEHNIKON **BELARUS**
Oktyabrskaya 16/5, Off. 703-711
BY-220030 Minsk
Phone: +375 (0)17 / 210 46 26
Fax: +375 (0)17 / 210 46 26

ESCO DRIVES & AUTOMATION **BELGIUM**
Culliganlaan 3
BE-1831 Diegem
Phone: +32 (0)2 / 717 64 30
Fax: +32 (0)2 / 717 64 31

Koning & Hartman b.v. **BELGIUM**
Woluwelaan 31
BE-1800 Vilvoorde
Phone: +32 (0)2 / 257 02 40
Fax: +32 (0)2 / 257 02 49

INEA BH d.o.o. **BOSNIA AND HERZEGOVINA**
Aleja Lipa 56
BA-71000 Sarajevo
Phone: +387 (0)33 / 921 164
Fax: +387 (0)33/ 524 539

AKHNATON **BULGARIA**
4 Andrej Ljapchev Blvd. Pb 21
BG-1756 Sofia
Phone: +359 (0)2 / 817 6044
Fax: +359 (0)2 / 97 44 06 1

INEA CR d.o.o. **CROATIA**
Losinjska 4 a
HR-10000 Zagreb
Phone: +385 (0)1 / 36 940 - 01/ -02/ -03
Fax: +385 (0)1 / 36 940 - 03

AutoCont C.S. s.r.o. **CZECH REPUBLIC**
Technologická 374/6
CZ-708 00 Ostrava-Pustkovec
Phone: +420 595 691 150
Fax: +420 595 691 199

Beijer Electronics A/S **DENMARK**
Lykkegårdsvej 17
DK-4000 Roskilde
Phone: +45 (0)46/ 75 76 66
Fax: +45 (0)46 / 75 56 26

Beijer Electronics Eesti OÜ **ESTONIA**
Pärnu mnt.160i
EE-11317 Tallinn
Phone: +372 (0)6 / 51 81 40
Fax: +372 (0)6 / 51 81 49

Beijer Electronics OY **FINLAND**
Peltoie 37
FIN-28400 Ulvila
Phone: +358 (0)207 / 463 540
Fax: +358 (0)207 / 463 541

UTECO **GREECE**
5, Mavrogenous Str.
GR-18542 Piraeus
Phone: +30 211 / 1206 900
Fax: +30 211 / 1206 999

MELTRADE Kft. **HUNGARY**
Fertő utca 14.
HU-1107 Budapest
Phone: +36 (0)1 / 431-9726
Fax: +36 (0)1 / 431-9727

Beijer Electronics SIA **LATVIA**
Ritausmas iela 23
LV-1058 Riga
Phone: +371 (0)784 / 2280
Fax: +371 (0)784 / 2281

Beijer Electronics UAB **LITHUANIA**
Savanoriu Pr. 187
LT-02300 Vilnius
Phone: +370 (0)5 / 232 3101
Fax: +370 (0)5 / 232 2980

EUROPEAN REPRESENTATIVES

ALFATRADE Ltd. **MALTA**
99, Paola Hill
Malta- Paola PLA 1702
Phone: +356 (0)21 / 697 816
Fax: +356 (0)21 / 697 817

INTEHSIS srl **MOLDOVA**
bld. Traian 23/1
MD-2060 Kishinev
Phone: +373 (0)22 / 66 4242
Fax: +373 (0)22 / 66 4280

HIFLEX AUTOM.TECHNIEK B.V. **NETHERLANDS**
Wolweverstraat 22
NL-2984 CD Ridderkerk
Phone: +31 (0)180 – 46 60 04
Fax: +31 (0)180 – 44 23 55

Koning & Hartman b.v. **NETHERLANDS**
Haarlerbergweg 21-23
NL-1101 CH Amsterdam
Phone: +31 (0)20 / 587 76 00
Fax: +31 (0)20 / 587 76 05

Beijer Electronics AS **NORWAY**
Postboks 487
NO-3002 Drammen
Phone: +47 (0)32 / 24 30 00
Fax: +47 (0)32 / 84 85 77

Fonseca S.A. **PORTUGAL**
R. João Francisco do Casal 87/89
PT - 3801-997 Aveiro, Esgueira
Phone: +351 (0)234 / 303 900
Fax: +351 (0)234 / 303 910

Sirius Trading & Services srl **ROMANIA**
Aleea Lacul Morii Nr. 3
RO-060841 Bucuresti, Sector 6
Phone: +40 (0)21 / 430 40 06
Fax: +40 (0)21 / 430 40 02

Craft Con. & Engineering d.o.o. **SERBIA**
Bulevar Svetog Cara Konstantina 80-86
SER-18106 Nis
Phone: +381 (0)18 / 292-24-4/5
Fax: +381 (0)18 / 292-24-4/5

INEA SR d.o.o. **SERBIA**
Izletnicka 10
SER-113000 Smederevo
Phone: +381 (0)26 / 617 163
Fax: +381 (0)26 / 617 163

SIMAP s.r.o. **SLOVAKIA**
Jána Derku 1671
SK-911 01 Trencín
Phone: +421 (0)32 743 04 72
Fax: +421 (0)32 743 75 20

PROCONT, spol. s r.o. Prešov **SLOVAKIA**
Kúpelná 1/A
SK-080 01 Prešov
Phone: +421 (0)51 7580 611
Fax: +421 (0)51 7580 650

INEA d.o.o. **SLOVENIA**
Stegne 11
SI-1000 Ljubljana
Phone: +386 (0)1 / 513 8100
Fax: +386 (0)1 / 513 8170

Beijer Electronics AB **SWEDEN**
Box 426
SE-20124 Malmö
Phone: +46 (0)40 / 35 86 00
Fax: +46 (0)40 / 93 23 01

Omni Ray AG **SWITZERLAND**
Im Schörli 5
CH-8600 Dübendorf
Phone: +41 (0)44 / 802 28 80
Fax: +41 (0)44 / 802 28 28

GTS **TURKEY**
Bayraktar Bulvari Nutuk Sok. No:5
TR-34775 Yukarı Dudullu-Ümraniye-İSTANBUL
Phone: +90 (0)216 526 39 90
Fax: +90 (0)216 526 3995

CSC Automation Ltd. **UKRAINE**
4-B, M. Raskovoyi St.
UA-02660 Kiev
Phone: +380 (0)44 / 494 33 55
Fax: +380 (0)44 / 494-33-66

EURASIAN REPRESENTATIVES

Kazpromautomatics Ltd. **KAZAKHSTAN**
Mustafina Str. 7/2
KAZ-470046 Karaganda
Phone: +7 7212 / 50 11 50
Fax: +7 7212 / 50 11 50

MIDDLE EAST REPRESENTATIVES

TEXEL ELECTRONICS Ltd. **ISRAEL**
2 Ha´umanut, P.O.B. 6272
IL-42160 Netanya
Phone: +972 (0)9 / 863 39 80
Fax: +972 (0)9 / 885 24 30

CEG INTERNATIONAL **LEBANON**
Cebaco Center/Block A Autostrade DORA
Lebanon - Beirut
Phone: +961 (0)1 / 240 430
Fax: +961 (0)1 / 240 438

AFRICAN REPRESENTATIVE

CBI Ltd. **SOUTH AFRICA**
Private Bag 2016
ZA-1600 Isando
Phone: + 27 (0)11 / 977 0770
Fax: + 27 (0)11 / 977 0761