Digitized Automation for a Changing World

DVP-ES2 / EX2 / EC5 / SS2 / SA2 / SX2 / SE & TP Operation Manual - Programming

www.deltaww.com

DVP-ES2/EX2/SS2/SA2/SX2/SE&TP

Operation Manual

Programming

Revision History

Issue	Description of Changes	Date
1st	The first edition is issued.	2010/02/26
2nd	The second edition is issued.	2011/03/03
3rd	1. Chapter 2.8 M Relay: Add M1037, M1119, M1182, M1308, M1346, and M1356, and update the description of the functions of M1055~M1057and M1183. 2. Chapter 2.13 Special Data Register: Add D1037, D1312, D1354, and D1900~D1931, and modify the attributes of the latched functions of D1062, D1114, D1115, and D1118. 3. Chapter 2.16 Applications of Special M Relays and D Registers: Update the description of the functions of RTCs; add M1037, D1037 (Enable SPD function), M1119 (Enable 2-speed output function of DDRVI instruction), M1308, D1312 (Output specified pulses or seek Z phase signal when zero point is achieved), and M1346 (Output clear signals when ZRN is completed); Easy PLC Link is changed to PLC Link, and the description is added. 4. Chapter 3.1 Basic Instructions (without API numbers) and Chapter 3.2 Explanations to Basic Instructions: Add NP and PN instructions, and add Chapter 3.7 Numerical List of Instructions (in alphabetic order) 5. Chapter 3.6 Numerical List of Instructions and	2011/09/29

Issue	Description of Changes	Date
	Chapter 3.8 Detailed Instruction Explanation: Increase explanations of DSPA instruction, and add floating-point contact type comparison instructions FLD=, FLD>, FLD<, FLD<>, FLD<=, FLD>=, FAND=, FAND>, FAND<, FAND<>, FAND<=, FAND>=, FOR=, FOR>, FOR<, FOR<>, FOR<=, FOR>=; add the supplementary description of PLSR instruction and the description of K11~K19 in DTM instruction mode; update the description of API166 instruction.	
4th	1. SE is added in the title of the manual. 2. Chapter 2.16: The default value in D1062 is K10. 3. API 15 in Chapter 3: The contents about S<D are deleted in program example 3. 4. API 148 and API 149 are added in Chapter 3. 5. The information related to DVP-SE is added. 6. The information related to DVP32ES-C is added. 7. The descriptions of the models are added in the contents. 8. Appendix A is added.	2012/07/01
5th	1. API 113 is added. 2. API150 is updated. 3. Chapter 7 is updated.	2012/09/01
6th	1. M1148, M1580, M1581, M1584, M1585, M1182, and M1183 are added to Chapter 2. 2. Chapter 3 is updated. API53, API 156, API 159, API69, API88, API143, API150, API155, API258, and API296-313 are added. 3. The description of API 178 is updated. 4. The description of the input/output mapping areas for DVP-ES2-C as a slave station is added to section 7.1.2. 5. C232, C249, and C250 are deleted from the description of the SE memory Map. 6. Appendix B is added.	2013/02/20

Issue	Description of Changes	Date
	7. Appendix C is added.	
7^{th}	1. The timer interrupts I805~I899 are added to Chapter2. The special auxiliary relays M1357~M1359, M1590, M1598, and M1599 are added to the table of special auxiliary relays. The descriptions of D1027 and D9998 in the table of special data registers are updated. D1056~D1059, D1150~D1153, D1246~D1247, and D9999 are added to the table of special data registers. The definitions of the pins in COM1 are added to the description of M1035. The new special auxiliary relays in the table of special data registers and the new special data registers in the table of special data registers are described in section 2.16. 2. API114, API115, API145, and API295 are added to Chapter 3. The descriptions of API17, API22, API23, API59, API78, API80, API81, API83, API101~API106, API112-API113, API150, API166, API179, and API197 are updated. 3. The information about M1040 is added to Chapter 5. 4. The description of the error code C450 is added to Chapter 6. 5. In Appendix C, the information about TP04P series text panels is changed to the information about TP series text panels. 6. Appendix D is added. It introduces the current consumption of slim PLCs/extension modules.	2014/07/04
8^{th}	1. In section B.1, the number of RTU modules onto which a DVP-SE series PLC can be mapped is updated. 2. In section B.2.2, the descriptions of CR#20~CR#86 are updated. 3. In section B.2.3, the descriptions of CR#17~CR#24 are updated, the description of	2014/08/29

Issue	Description of Changes	Date
	CR#27 is added, and the descriptions of CR#87~CR103 are updated. 4. In section B.2.4, the descriptions of CR#0 and CR#20~CR#26 are updated. 5. Section B.6 is added.	
9th	1. Chapter 2: add Ethernet descriptions in section 2.1, update M1119, M1334, M1338, M1700~M1731 in section 2.8, updated software/hardware high speed counter descriptions in section 2.12, add D1021 descriptions in section 2.13, add interrupt descriptions in section 2.15, add D1021, M1334, M1335 and update M1119 and PLC link descriptions in section 2.16. 2. Chapter 3: update instruction list. Update API113, API15, API17, API51, API59, API68, API76, API80, API123, API150, API158, API159, API206, and add new instruction descriptions API337. 3. Chapter 4: update Modbus address for SE series in section 4.4. 4. Chapter 6: delete error codes C430, C441, and C442. Add new error codes C430, C437 and C438 in section 6.2. 5. Appendix A: add descriptions of the USB installation in Windows 10. 6. Appendix B: add descriptions of ES2-E series. 7. Appendix C: update descriptions of program capacity for TP series. 8. Appendix D: add descriptions for 28SS2/28SA2/26SE.	2017/04/26
10th	1. Section 2.1: updated file register contents. 2. Section 2.2-2.4: updated external inputs X and Y 3. Section 2.8: added M1019, M1145, M1196-M1198, M1614-M1675 and updated M1119, M1183, M1334, M1335, and 1700-1731.	2018/10/30

Issue	Description of Changes	Date
	4. Section 2.12: updated supporting modules for hardware high-speed counters. 5. Section 2.13: updated D1021 and added D1175-D1177, D1227-D1231, D1400-D1403 and D6000-D6063. 6. Section 2.15: updated timer interrupt in API05. 7. Section 2.16: added M1019 as well as 1145 and update D1020, D1021, M1119, PLC Link, M1334, M1335. 8. Section 3.5: updated index register E and F. 9. Section 3.6: added API315, API316, API328-API336, API338-API340 and API342. 10. Section 3.8: updated API50, API53-API59, API68, API80, API85, API86, API113, API145, API148, API149, API158, API159, API198 and API337 and add API315, API316, API328-API336, API338-API340, API342. 11. Section 7.1.1: updated maximum number of PDO supported and updated standard Delta cable model names. 12. Appendix A.1: updated installation instruction for Windows 7. 13. Appendix B.1: updated Ethernet function. 14. Appendix B.5: updated object list. 15. Appendix C: updated D1114 and D1115. 16. Appendix C.4.3: TP04P-08TP1R does not support high-speed inputs. 17. Appendix E: added a new appendix for slim-type special modules. 18. Appendix F: added a new appendix for slim-type PLC specifications.	
11th	1. Section 2.4: updated 32bit high-speed count up/down hardware specification for SE series PLC. 2. Section 2.8: updated M1037, M1113, M1145, M1190-M1198, M1334, M1335, M1357-M1359, M1620-M1623, and M1672-M1675. Added M1077, M1116, M1117, M1550, M1551, and M1582.	2021.11.15

Issue	Description of Changes	Date
	3. Section 2.12: updated the software high speed counters and the maximum frequency of hardware high speed counters C24 and C250. 4. Section 2.13: updated D1021, D1037, D1038, D1056-D1059, D1246, and D1247. Added D1110 and D1111. 5. Section 2.16: updated the following contents: M1077 for real-time clock, D1021 for adjustment on input terminal response time, M1033 for output state latched in STOP mode, COM2 in COM port function, the availability of M1037 and D1037 for Enable SPD function, the availability of D1038 for communication response delay function, M1119 for enable 2-speed output function of DDRVI and DDRVA instructions, D1320 to D1327 for ID of right side modules on ES2/EX2/SS2/SA2/SX2/SE, D1386 to D1393 for ID of left side modules on SA2/SX2/SE, M1182 and D9800 to D9879 for Mapping function of SA2/SX2/SE for left-side high-speed special modules, M1183 and D9900 to D9979 for Mapping function for right-side high-speed special modules, added state codes of DVP02PU-E2, M1334 and M1335 for executing API59, PLSR/DPLSR instructions for ramp-down when the conditional contacts are closed, the availability of M1019 for if the PLC detects that the external 24V voltage is unstable, the error LED flashes, and the availability of M1145 for read MAC address from the left side network module EN01. 6. Chapter 3: updated API50, API51, API53, API54, API55, API57, API58, API59, API68, API78, API79, API80, API87, API88, API113, API148, API149, API158, API159, API207, API175-API280, API315, API337, and API340. Added API343-API354. 7. Section 3.2: updated the instructions OUT, SET, and RST. 8. Section 4.1: updated communication baud rate. Section 4.4: updated PLC device address. 9. Section 5.3: updated the contents of Repeated Usage of Output Coil. 10. Section 6.1: updated the contents of Common Problems and Solutions. 11. Section 6.2: updated the error codes C407, and	

Issue	Description of Changes	Date
	C430 to C438. Added new error codes C41C, C421 to C426, C451 to C453, and C461 to C465 12. Section 6.3: added error code 0E1C for device D1067. 13. Appendix A: added section A.4 Notes on Utilizing USB communication. 14. Appendix B: updated the descriptions for CR#26, and CR#88 to CR#93 in sections from B.2.2 to B.2.4. 15. Appendix C: updated the relay timer specification, counter C251 specification and serial ports specification in section C.1. Updated D1000 and D1022 in section C.2. Added C.2.1 PLC CPU Built-in Inputs and Outputs. Added C.5 Definitions of TP04P Communication Ports. 16. Appendix D: updated the contents in D.1.1 to D.1.4.	
12th	1. Section 2.8: updated M1084, M1356, M1360-M1439, M1582, and M1700-M1731. Added M1283. 2. Section 2.13: updated D1123, D1168, D1355-D1370, and D1415-D1991. Added D1248, D1254, D1385. 3. Section 2.16: updated Delta communication cable, PLC LINK description and added a new function group, Check the input/output points of the digital extension module or number of the special modules when power-on. 4. Chapter 3: updated K33 and K36 in DTM instruction (API68). Added descriptions as well as examples for M1263 and updated D1123, and D1168 in RS instruction (API80). Added new communication mode (S2) in ETHRS instruction (API 337). Updated the supporting firmware version of DVP-SS2 in $MOV (API343).	2022/3/22
13th	1. Added notes on the maximum frequency of various input counters for the following series, ES2, 12SA2, SX2, SS2, 12SE, 26SE, and 28SA2 in section 2.12. 2. Categorized the same groups of the instructions in the same sections in Chapter 3. 3. Updated the following instructions API113, API156, API158, API315, API337, and API344-API354 in Chapter 3. 4. Updated the general specification from slim type	2022/08/19

Issue	Description of Changes	Date
	PLC CPU to ES2 and EX2 series PLC CPU in section F1, added the general specification for SA2, SS2, SX2 and SE series PLC CPU in section F2 and added the general specification for TP series PLC CPU in section F3.	
14th	1. Added EC5 series product information in section 2.1 ES2/EX2/EC5 Memory Map. 2. Updated the contents in the specification of real time clock and special I/O modules in sections 2.2, 2.3 and 2.4. 3. Updated the followings in section 2.8: M1030, M1037, M1058, M1070, M1071, M1077, M1079, M1083, M1084, M1103, M1105, M1106, M1109, M1111, M1116, M1117, M1133, M1134, M1136, M1148, M1156, M1157, M1158, M1159, M1181, M1183, M1190-M1193, M1257, M1310, M1311, M1316-M1320, M1348, M1349,M1357-M1359, M1525, M1539, M1541, M1580-M1585, M1598, M1599, M1672, M1673, M1675, and M1700-M1731. Added the followings in sect9ion 2.8: M1133, M1134, M1310, M1311, M1321, M1322, M1520, M1521, M1526-M1527, M1530-M1533, M1536, M1537, M1542, M1543, M1576, M1577, M1586- M1589, M1604-M1607, M1676 and M1677. 4. Updated contents of Note for Applicable Software High Speed Counter and Hardware High Speed Counters in section 2.12. 5. Updated the followings in section 2.13: D1002, D1003, D1023, D1026, D1027, D1032, D1033, D1037, D1056-D1059, D1062, D1109, D1110-D1118, D1135, D1136, D1140, D1169, D1232-D1235, D1246-D1248, D1252-D1255, D1320-D1327, D1338, D1339, D1385, D1900-1931, D6000-D6063, and D9900-D9999. Added the followings in section 2.13: D1341, D1342, D1350, D1351, D1375-D1382, D1410-D1413, and D1988-D1991. 6. Added EC5 Series production information in the following Function Groups: Program Capacity, Communication Response Delay, Enable 2-speed output function of DDRVI/DDRVA instruction, PLC Link, When the conditional contacts are closed, execute the ramp-down on the outputs and updated Function Group: ID of left side modules on SA2/SX2/SE in section 2.16.	2023/02/10

Issue	Description of Changes	Date
	7. Added EC5 Series product information in sections from 3.1 to 3.5. 8. Added EC5 Series as an applicable model and updated the following instruction in section 3.6 API50, API53-API59, API143, API155-API159, API340; added output points for high speed instructions in section 3.6.6.1; added output pints for positioning instructions in section 3.6.14.1. 9. Added EC5 Series product information in chapters 4 and 5. 10. Added EC5 Series general specifications in section F.3. Updated other specifications for other series in Appendix F.	
15th	1. Updated M1262 in section 2.8. 2. Updated D1988, D1989, D1990, D1991 in section 2.13. 3. Updated COM port information in section 2.16. 4. Updated the information of applicable modules for basic instructions in section 3.2. 5. Updated the information of applicable modules for interrupts in section 3.4. 6. Updated the example 2 in APII02, the explanation #7 in API106, the example in API12, the applicable modules in API343, the parameter explanation in API68,the A2 wiring in API155, the special M for EC5 in API156-API159, the applicable modules in API195 in section 3.6 7. Updated the default communication settings for all COM ports in section 4.1. 8. Updated the error LED information and added low voltage handling process in Chapter 6. 9. Added an example of setting up the communication in Appendix C. 10. Updated the current consumption information for DVP01PU-S, DT01-S, 02TUR-S, DNET-SL, as well as EN01-SL and added current consumption information for new modules DVP08SM10N, 02TKL-S, 02TKN-S, 02TKR-S, 02PU-SL, PF02-SL, SCM12-SL and SCM52-SL in Appendix D. 11. Updated the specification for EC5 in section F.3.	2023/12/21

DVP-ES2/EX2/EC5/SS2/SA2/SX2/SE&TP Operation Manual Programming

Contents

3 Instruction Set

4 Communications

5 Sequential Function Chart

6 Troubleshooting

7 CANopen Function and Operation

Appendix A

Appendix B

Appendix C

Appendix D

Appendix E

Appendix F

The DVP series PLCs are listed below.

Series	Model name
DVP-ES2	DVP16ES200R, DVP16ES200T, DVP24ES200R, DVP24ES200T, DVP32ES200R, DVP32ES200T, DVP32ES211T, DVP40ES200R, DVP40ES200T, DVP60ES200R, DVP60ES200T, DVP40ES200RM, DVP58ES200R, DVP58ES200T, DVP80ES200R, DVP80ES200T
DVP-ES2-C	DVP32ES200RC, DVP32ES200TC
DVP-ES2-E	DVP20ES200RE, DVP20ES200TE, DVP32ES200RE, DVP32ES200TE, DVP40ES200RE, DVP40ES200TE, DVP60ES200RE, DVP60ES200TE
DVP-EX2	DVP20EX200R, DVP20EX200T, DVP30EX200R, DVP30EX200T
DVP-SS2	DVP14SS211R, DVP14SS211T, DVP28SS211R, DVP28SS211T, DVP12SS211S
DVP-SA2	DVP12SA211R, DVP12SA211T, DVP28SA211R, DVP28SA211T
DVP-SX2	DVP20SX211R, DVP20SX211S, DVP20SX211T
DVP-SE	DVP12SE11R, DVP12SE11T, DVP26SE11R, DVP26SE11T, DVP26SE11S
TP	TP04P-16TP1R, TP04P-32TP1R, TP04P-22XA1R, TP04P-21EX1R, TP04P-16TP1T, TP04P-32TP1T, TP04P-22XA1T, TP04P-21EX1T, TP70P-16TP1R, TP70P-32TP1R, TP70P-22XA1R, TP70P-21EX1R, TP70P-16TP1T, TP70P-32TP1T, TP70P-22XA1T, TP70P-21EX1T, TP04P-08TP1R

1

PLC Concepts

This chapter introduces basic and advanced concepts of ladder logic, which is the mostly adopted programming language of PLC. Users familiar with the PLC concepts can move to the next chapter for further programming concepts. However, for users not familiar with the operating principles of PLC, please refer to this chapter to get a full understanding of PLC concepts.

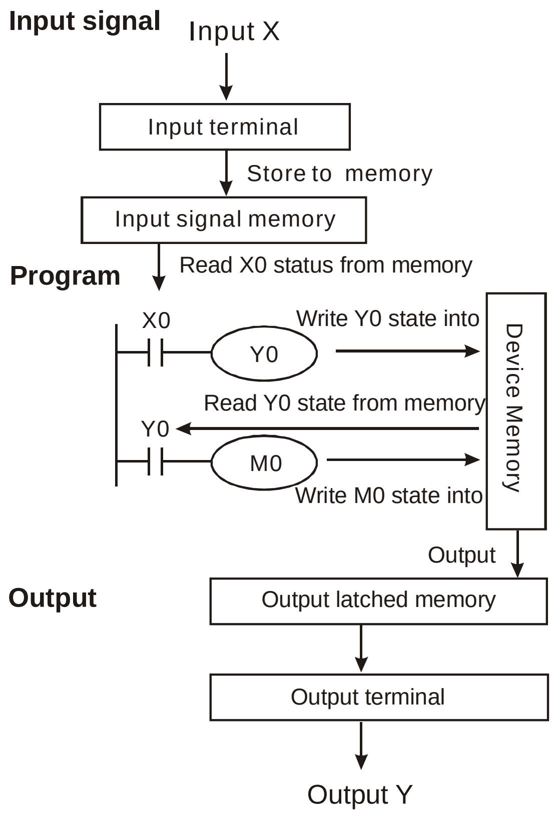
Table of Contents

1.1 PLC Scan Method

PLC utilizes a standard scan method when evaluating user program.

Scanning process:

Scan input status	Read the physical input status and store the data in internal memory.
Evaluate user program	Evaluate the user program with data stored in internal memory. Program scanning starts from up to down and left to right until reaching the end of the program.
Refresh the outputs	Write the evaluated data to the physical outputs



Input signal:
PLC reads the ON/OFF status of each input and stores the status into memory before evaluating the user program.
Once the external input status is stored into internal memory, any change at the external inputs will not be updated until next scan cycle starts.
Program:
PLC executes instructions in user program from top to down and left to right then stores the evaluated data into internal memory. Some of this memory is latched.
Output:
When END command is reached the program evaluation is complete. The output memory is transferred to the external physical outputs.

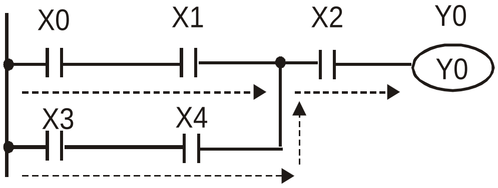
Scan time
The duration of the full scan cycle (read, evaluate, write) is called "scan time." With more I/O or longer program, scan time becomes longer.

Read scan time	PLC measures its own scan time and stores the value (0.1ms) in register D1010, minimum scan time in register D1011, and maximum scan time in register D1012.
Measure scan time	Scan time can also be measured by toggling an output every scan and then measuring the pulse width on the output being toggled.
Calculate scan time	Scan time can be calculated by adding the known time required for each instruction in the user program. For scan time information of individual instruction please refer to Ch3 in this manual.

Scan time exception
PLC can process certain items faster than the scan time. Some of these items interrupts and halt the scan time to process the interrupt subroutine program. A direct I/O refresh instruction REF allows the PLC to access I/O immediately during user program evaluation instead of waiting until the next scan cycle.

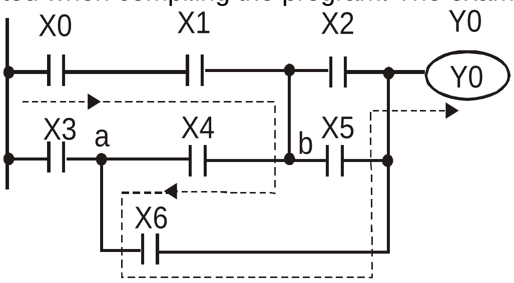
1.2 Current Flow

Ladder logic follows a left to right principle. In the example below, the current flows through paths started from either X0 or X3.

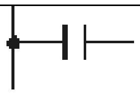
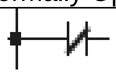


Reverse Current

When a current flows from right to left, which makes a reverse current logic, an error will be detected when compiling the program. The example below shows the reverse current flow.



1.3 NO Contact, NC Contact

NO contact	 Normally Open Contact, A contact
NC Contact	 Normally Closed Contact, B contact

1.4 PLC Registers and Relays

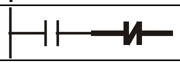
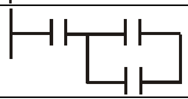
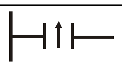
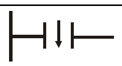
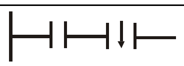
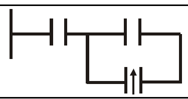
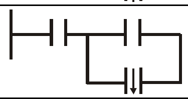
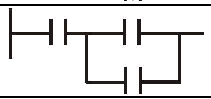
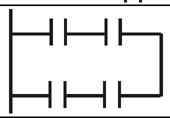
Introduction to the basic internal devices in a PLC

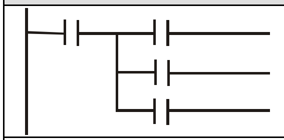
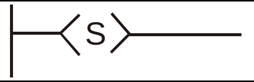
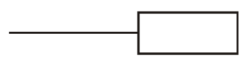
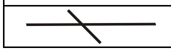
X (Input Relay)	Bit memory represents the physical input points and receives external input signals. ■ Device indication: Indicated as **X** and numbered in octal, e.g. X0~X7, X10~X17…X377
Y (Output Relay)	Bit memory represents the physical output points and saves the status to be refreshed to physical output devices. ■ Device indication: Indicated as **Y** and numbered in octal, e.g. Y0~Y7, Y10~Y17. ..Y377
M (Internal Relay)	Bit memory indicates PLC status. ■ Device indication: Indicated as **M** and numbered in decimal, e.g. M0, M1, M2…M4095
S (Step Relay)	Bit memory indicates PLC status in Step Function Control (SFC) mode. If no STL instruction is applied in program, step point S can be used as an internal relay M as well as an annunciator. ■ Device indication: Indicated as **S** and numbered in decimal, e.g. S0, S1, S2…S1023
T (Relay) (Word) (Dword)	Bit, word or double word memory used for timing and has coil, contact and register in it. When its coil is ON and the set time is reached, the associated contact will be energized. Every timer has its resolution (unit: 1ms/10ms/100ms). ■ Device indication: Indicated as **T** and numbered in decimal, e.g. T0, T1, T2…T255

C (Counter) (Relay) (Word) (Dword)	Bit, word or double word memory used for counting and has coil, contact and register in it. The counter count once (1 pulse) when the coil goes from OFF to ON. When the predefined counter value is reached, the associated contact will be energized. There are 16-bit and 32-bit high-speed counters available for users. ■ Device indication: Indicated as **C** and numbered in decimal, e.g. C0, C1, C2…C255
D (Data register) (Word)	Word memory stores values and parameters for data operations. Every register is able to store a word (16-bit binary value). A double word will occupy 2 consecutive data registers. ■ Device indication: Indicated as **D** and numbered in decimal, e.g. D0, D1, D2…D4999
E, F (Index register) (Word)	Word memory used as a modifier to indicate a specified device (word and double word) by defining an offset. Index registers not used as a modifier can be used as general purpose register. ■ Device indication: indicated as E0 ~ E7 and F0 ~ F7.

1.5 Ladder Logic Symbols

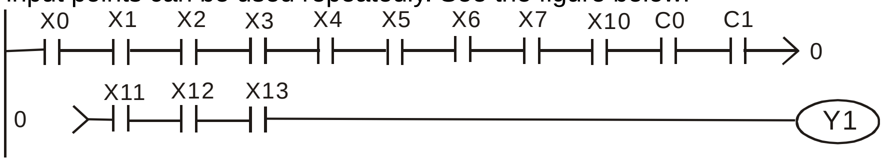
The following table displays list of WPLSoft symbols their description, command, and memory registers that are able to use the symbol.

Ladder Diagram Structure	Explanation	Instruction	Available Devices
	NO (Normally Open) contact / A contact	LD	X, Y, M, S, T, C
	NC (Normally Closed) contact / B contact	LDI	X, Y, M, S, T, C
	NO contact in series	AND	X, Y, M, S, T, C
	NC contact in series	ANI	X, Y, M, S, T, C
	NO contact in parallel	OR	X, Y, M, S, T, C
	NC contact in parallel	ORI	X, Y, M, S, T, C
	Rising-edge trigger switch	LDP	X, Y, M, S, T, C
	Falling-edge trigger switch	LDF	X, Y, M, S, T, C
	Rising-edge trigger in series	ANDP	X, Y, M, S, T, C
	Falling-edge trigger in series	ANDF	X, Y, M, S, T, C
	Rising-edge trigger in parallel	ORP	X, Y, M, S, T, C
	Falling-edge trigger in parallel	ORF	X, Y, M, S, T, C
	Block in series	ANB	None
	Block in parallel	ORB	None

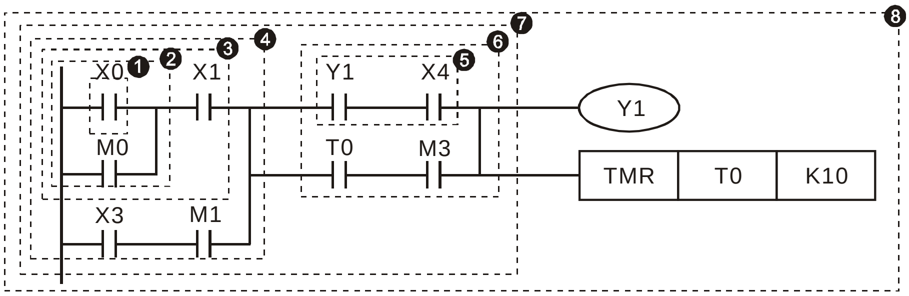
Ladder Diagram Structure	Explanation	Instruction	Available Devices
	Multiple output branches	MPS MRD MPP	None
	Output coil	OUT	Y, M, S
	Step ladder	STL	S
	Basic / Application instruction	-	Basic instructions and API instructions. Please refer to chapter 3 Instruction Set
	Inverse logic	INV	None

1.5.1 Creating a PLC Ladder Program

The editing of the program should start from the left side bus line to the right side bus line, and from up to down. However, the right side bus line is omitted when editing in WPLSoft. A single row can have maximum 11 contacts on it. If more than 11 contacts are connected, a continuous symbol “0” will be generated automatically and the 12th contact will be placed at the start of next row. The same input points can be used repeatedly. See the figure below:



When evaluating the user program, PLC scan starts from left to right and proceeds to next row down until the PLC reaches END instruction. Output coils and basic / application instructions belong to the output process and are placed at the right of ladder diagram. The sample program below explains the execution order of a ladder diagram. The numbers in the black circles indicate the execution order.

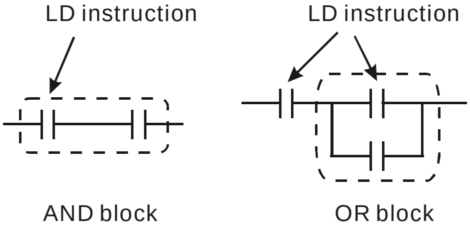


Execution order of the sample program:

1	LD	X0
2	OR	M0
3	AND	X1
4	LD	X3
	AND	M1
	ORB	
5	LD	Y1
	AND	X4
6	LD	T0
	AND	M3
	ORB	
7	ANB	
8	OUT	Y1
	TMR	T0 K10

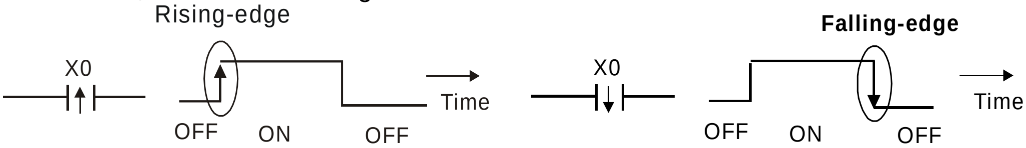
1.5.2 LD / LDI (Load NO contact / Load NC contact)

LD or LDI starts a row or block



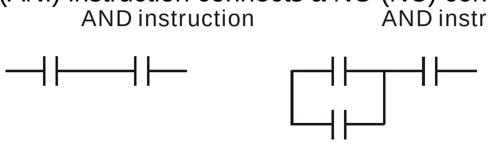
1.5.3 LDP / LDF (Load Rising edge trigger/ Load Falling edge trigger)

Similar to LD instruction, LDP and LDF instructions only act at the rising edge or falling edge when the contact is ON, as shown in the figure below.



1.5.4 AND / ANI (Connect NO contact in series / Connect NC contact in series)

AND (ANI) instruction connects a NO (NC) contact in series with another device or block.

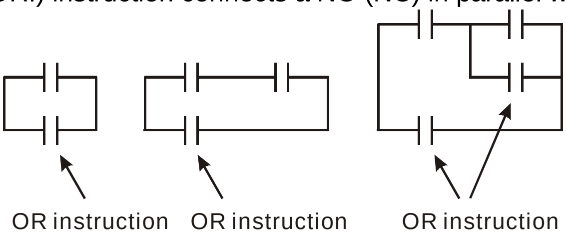


1.5.5 ANDP / ANDF (Connect Rising edge in series/ Connect Falling edge in series)

Similar to AND instruction, ANDP (ANDF) instruction connects rising (falling) edge triggers in series with another device or block.

1.5.6 OR / ORI (Connect NO contact in parallel / Connect NC contact in parallel)

OR (ORI) instruction connects a NO (NC) in parallel with another device or block.

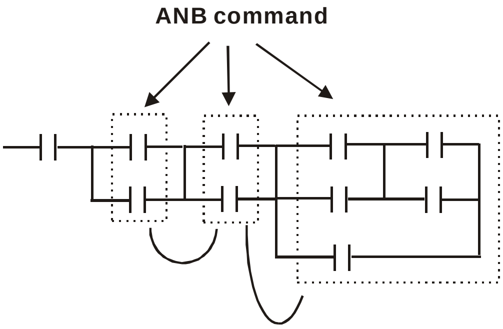


1.5.7 ORP / ORF (Connect Rising edge in parallel/ Connect Falling edge in parallel)

Similar to OR instruction, ORP (ORF) instruction connects rising (falling) edge triggers in parallel with another device or block

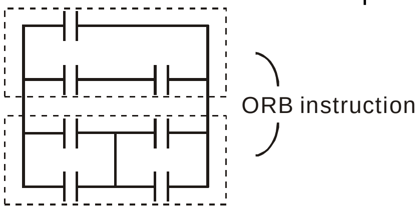
1.5.8 ANB (Connect block in series)

ANB instruction connects a block in series with another block



1.5.9 ORB (Connect block in parallel)

ORB instruction connects a block in parallel with another block



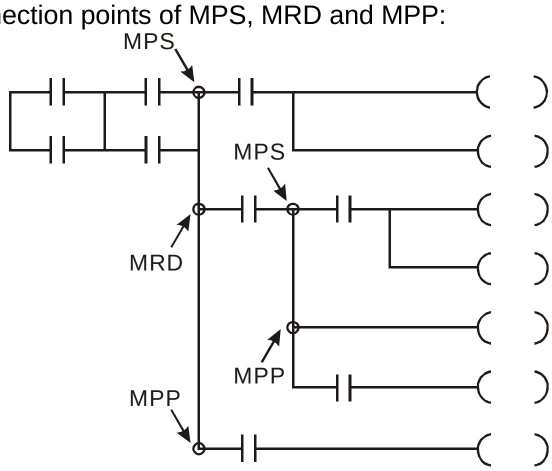
1.5.10 MPS / MRD / MPP (Branch instructions)

These instructions provide a method to create multiplexed output branches based on current result stored by MPS instruction.

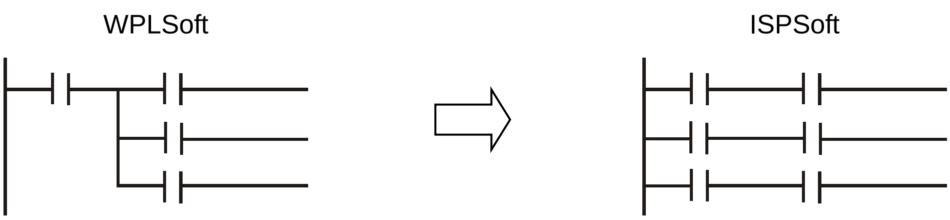
Branch instruction	Branch Symbol	Description
MPS	┬	Start of branches. Stores current result of program evaluation. Max. 8 MPS-MPP pairs can be applied
MRD	├	Reads the stored current result from previous MPS
MPP	└	End of branches. Pops (reads then resets) the stored result in previous MPS

Note: When compiling ladder diagram with WPLSoft, MPS, MRD and MPP could be automatically added to the compiled results in instruction format. However, sometimes the branch instructions are ignored by WPLSoft if not necessary. Users programming in instruction format can enter branch instructions as required.

Connection points of MPS, MRD and MPP:

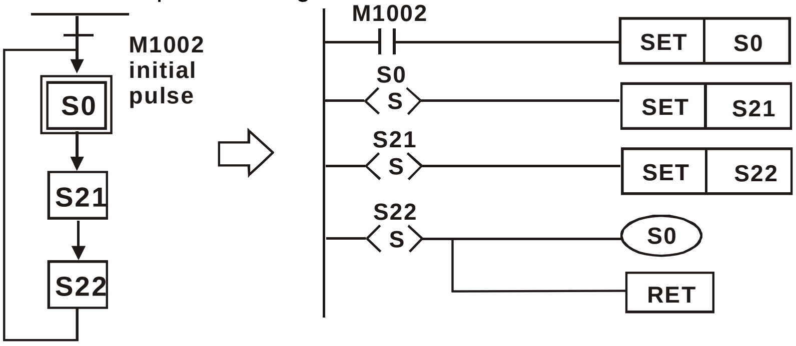


Note: Ladder diagram editor in ISPSoft does not support MPS, MRD and MPP instructions. To achieve the same results as branch instructions, users have to connect all branches to the left hand bus bar.



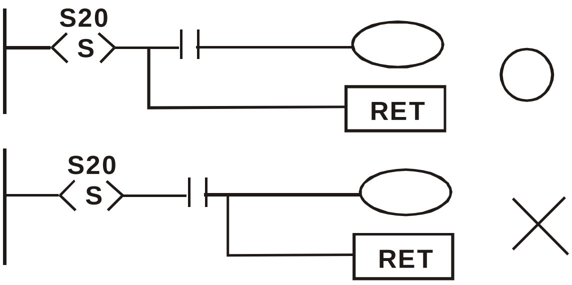
1.5.11 STL (Step Ladder Programming)

STL programming uses step points, e.g. S0 S21, S22, which allow users to program in a clearer and understandable way as drawing a flow chart. The program will proceed to next step only if the previous step is completed, therefore it forms a sequential control process similar to SFC (Sequential Function Chart) mode. The STL sequence can be converted into a PLC ladder diagram which is called "step ladder diagram" as below.



1.5.12 RET (Return)

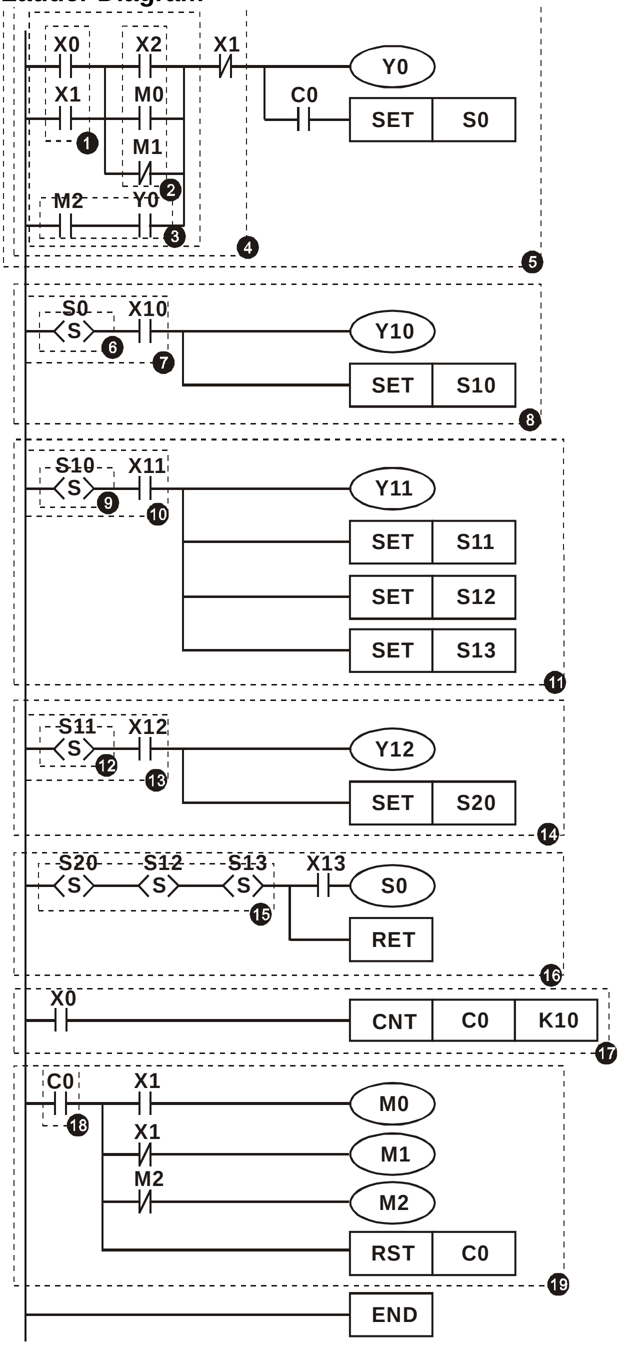
RET instruction has to be placed at the end of sequential control process to indicate the completion of STL flow.



Note: Always connect RET instruction immediately after the last step point indicated as the above diagram otherwise program error may occur.

1.6 Conversion between Ladder Diagram and Instruction List Mode

Ladder Diagram



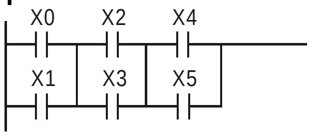
Instruction

Instruction	Operand	Comment	
LD	X0	❶ OR block	
OR	X1		
LD	X2	❷ OR block	
OR	M0		
ORI	M1		
ANB		Block in series	
LD	M2	❸ AND block	
AND	Y0		
ORB		Block in parallel	
ANI	X1	❹ ANI	❺ The output continues based on status of ❹
OUT	Y0	Multiple outputs	
AND	C0		
SET	S0		
STL	S0	Start of step ladder ❻	❼
LD	X10	S0 status operates with X10	❽
OUT	Y10	Output Y10 and transfer of step point	
SET	S10		
STL	S10	❾ Read S10 status	⓫
LD	X11	❿ S10 operates with X11	
OUT	Y11	Output Y11 and transfer of step points	
SET	S11		
SET	S12		
SET	S13		
STL	S11	⓬ Read S11 status	⓮
LD	X12	⓭ S11 operates with X12	
OUT	Y12	Output Y12 and transfer of step points	
SET	S20		
STL	S20	⓯ Convergence of multiple status	⓰ End of step ladder
STL	S12		
STL	S13		
LD	X13	Read X13 status and transfer of step point	
OUT	S0		
RET		Return	
LD	X0	⓱	
CNT	C0 K10		
LD	C0	⓲ Read C0	⓳
MPS		Multiple outputs	
AND	X1		
OUT	M0		
MRD			
ANI	X1		
OUT	M1		
MPP			
ANI	M2		
OUT	M2		
RST	C0		
END		End of program	

1.7 Fuzzy Syntax

Generally, the ladder diagram programming is conducted according to the “up to down and left to right” principle. However, some programming methods not following this principle still perform the same control results. Here are some examples explaining this kind of “fuzzy syntax.”

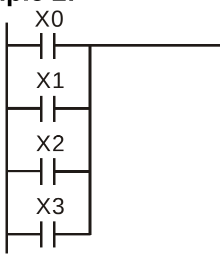
Example 1:



Better method		OK method	
LD	X0	LD	X0
OR	X1	OR	X1
LD	X2	LD	X2
OR	X3	OR	X3
ANB		LD	X4
LD	X4	OR	X5
OR	X5	ANB	
ANB		ANB	

The two instruction programs can be converted into the same ladder diagram. The difference between Better and OK method is the ANB operation conducted by MPU. ANB instruction cannot be used continuously for more than 8 times. If more than 8 ANB instructions are used continuously, program error will occur. Therefore, apply ANB instruction after a block is made is the better method to prevent the possible errors. In addition, it’s also the more logical and clearer programming method for general users.

Example 2:

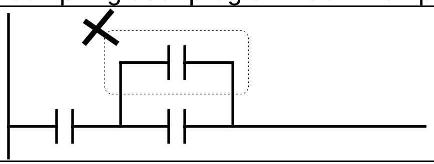
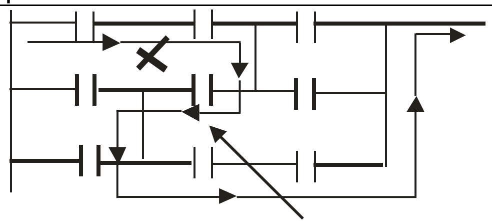
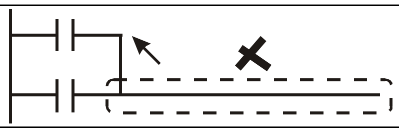


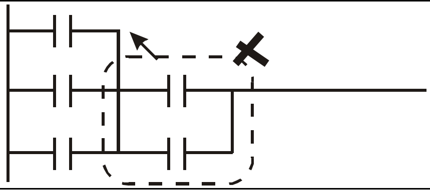
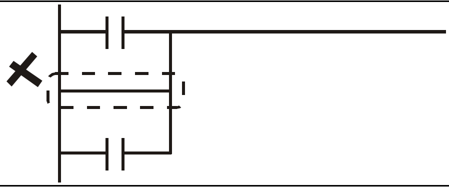
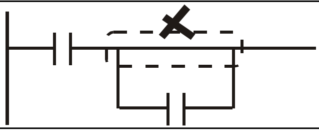
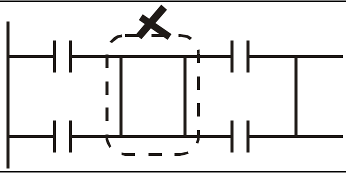
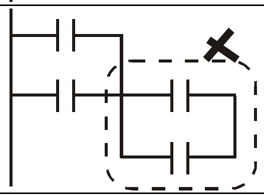
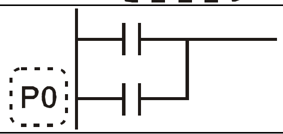
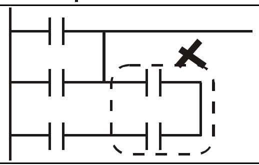
Good method		Bad method	
LD	X0	LD	X0
OR	X1	LD	X1
OR	X2	LD	X2
OR	X3	LD	X3
		ORB	
		ORB	
		ORB	

The difference between Good and Bad method is very clear. With longer program code, the required MPU operation memory increases in the Bad method. To sum up, following the general principle and applying good / better method when editing programs prevents possible errors and improves program execution speed as well.

Common Programming Errors

PLC processes the diagram program from up to down and left to right. When editing ladder diagram users should adopt this principle as well otherwise an error would be detected by WPLSoft when compiling user program. Common program errors are listed below:

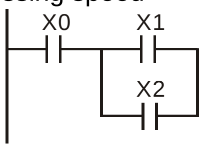
	OR operation upward is not allowed.
	“Reverse current” exists.
	Output should be connected on top of the circuit.

	Block combination should be made on top of the circuit.
	Parallel connection with empty device is not allowed..
	Parallel connection with empty device is not allowed.
	No device in the middle block.
	Devices and blocks in series should be horizontally aligned
	Label P0 should be at the first row of the complete network.
	“Reverse current” exists

1.8 Correcting Ladder Diagram

Example 1:

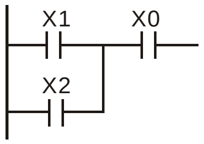
Connect the block to the front for omitting ANB instruction because simplified program improves processing speed



Instruction List

LD	X0
LD	X1
OR	X2
ANB	

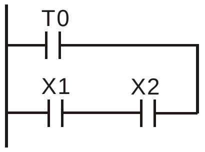
⇩



Instruction List

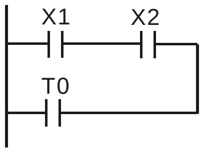
LD	X1
OR	X2
AND	X0

Example 2:
When a device is to be connected to a block, connect the device to upper row for omitting ORB instruction



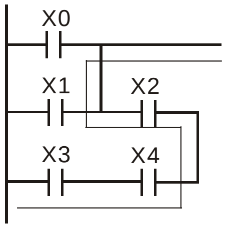
Instruction List	
LD	T0
LD	X1
AND	X2
ORB	

⇩



Instruction List	
LD	X1
AND	X2
OR	T0

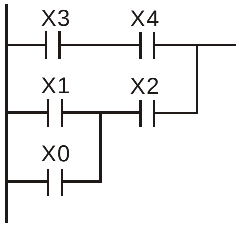
Example 3:
“Reverse current” existed in diagram (a) is not allowed for PLC processing principle.



(a)

Instruction List	
LD	X0
OR	X1
AND	X2
LD	X3
AND	X4
ORB	

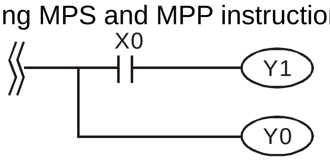
⇩



(b)

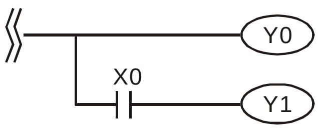
Instruction List	
LD	X3
AND	X4
LD	X1
OR	X0
AND	X2
ORB	

Example 4:
For multiple outputs, connect the output without additional input devices to the top of the circuit for omitting MPS and MPP instructions.



Instruction List	
MPS	
AND	X0
OUT	Y1
MPP	
OUT	Y0

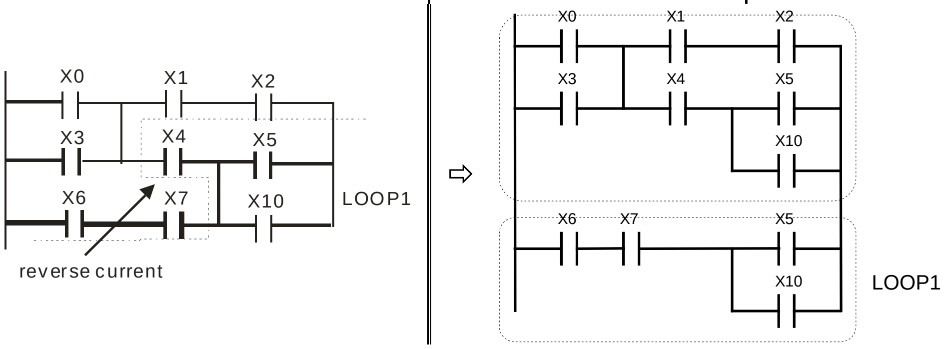
⇩



Instruction List	
OUT	Y0
AND	X0
OUT	Y1

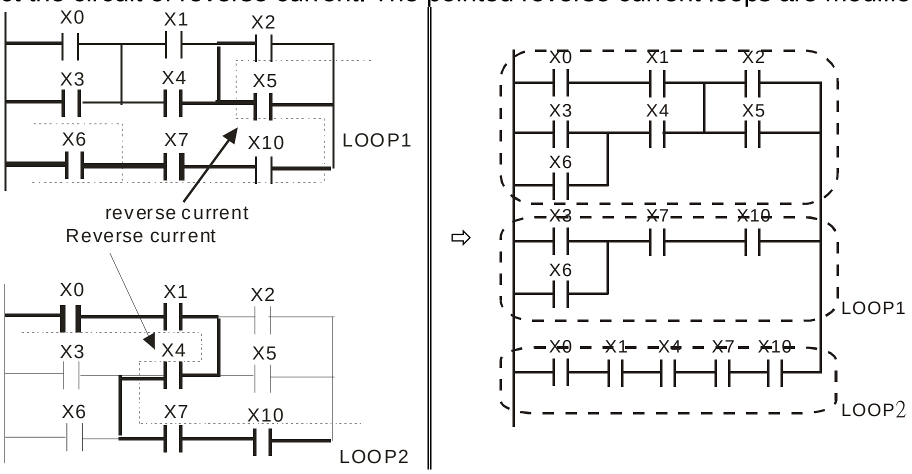
Example 5:

Correct the circuit of reverse current. The pointed reverse current loops are modified on the right.



Example 6:

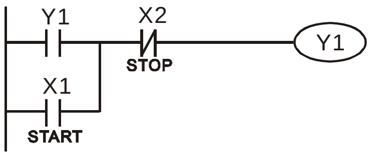
Correct the circuit of reverse current. The pointed reverse current loops are modified on the right.



1.9 Basic Program Design Examples

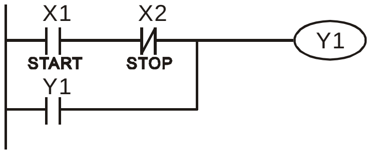
Example 1 - Stop First latched circuit

When X1 (START) = ON and X2 (STOP) = OFF, Y1 will be ON. If X2 is turned on, Y1 will be OFF. This is a Stop First circuit because STOP button has the control priority than START



Example 2 - Start First latched circuit

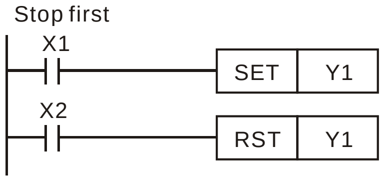
When X1 (START) = ON and X2 (STOP) = OFF, Y1 will be ON and latched. If X2 is turned ON, Y1 remains ON. This is a Start First circuit because START button has the control priority than STOP



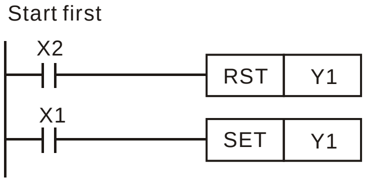
Example 3 - Latched circuit of SET and RST

The diagram opposite are latched circuits consist of RST and SET instructions.

In PLC processing principle, the instruction close to the end of the program determines the final output status of Y1. Therefore, if both X1 and X2 are ON, RST which is lower than SET forms a

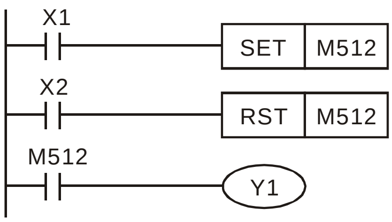


Stop First circuit while SET which is lower than RST forms a Start First circuit.

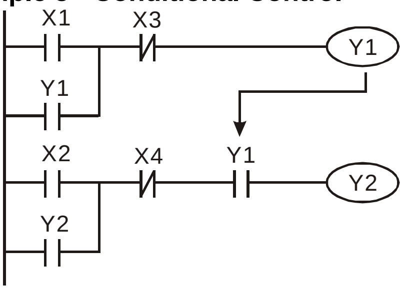


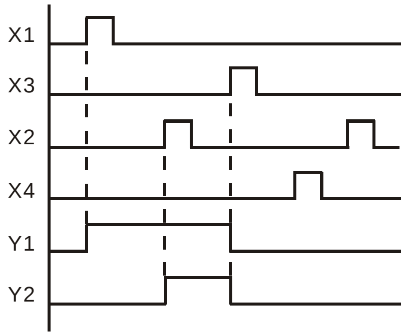
Example 4 - Power down latched circuit

The auxiliary relay M512 is a latched relay. Once X1 is ON, Y1 retains its status before power down and resumes after power up.



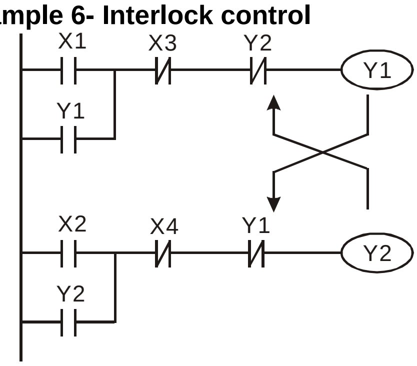
Example 5 - Conditional Control

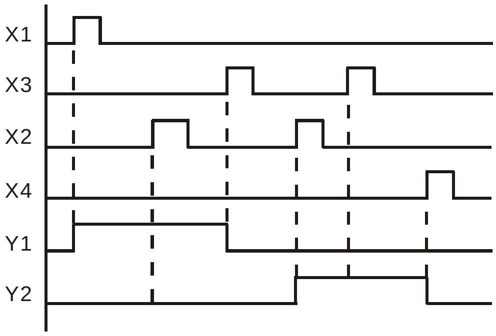




Because NO contact Y1 is connected to the circuit of Y2 output, Y1 becomes one of the conditions for enabling Y2, i.e. for turning on Y2, Y1 has to be ON

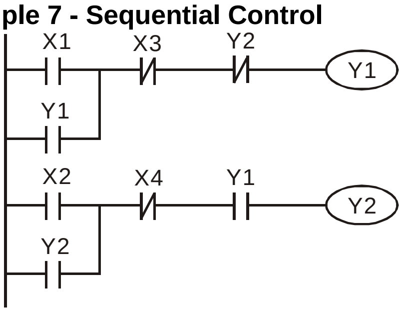
Example 6- Interlock control





NC contact Y1 is connected to Y2 output circuit and NC contact Y2 is connected Y1 output circuit. If Y1 is ON, Y2 will definitely be OFF and vice versa. This forms an Interlock circuit which prevents both outputs to be ON at the same time. Even if both X1 and X2 are ON, in this case only Y1 will be enabled.

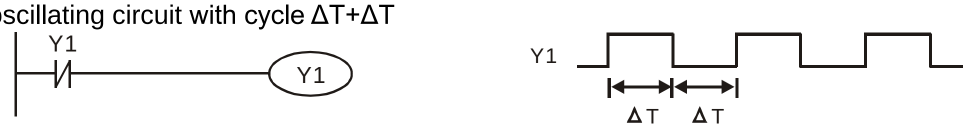
Example 7 - Sequential Control



Connect NC contact Y2 to Y1 output circuit and NO contact Y1 to Y2 output circuit. Y1 becomes one of the conditions to turn on Y2. In addition, Y1 will be OFF when Y2 is ON, which forms an sequential control process.

Example 8 - Oscillating Circuit

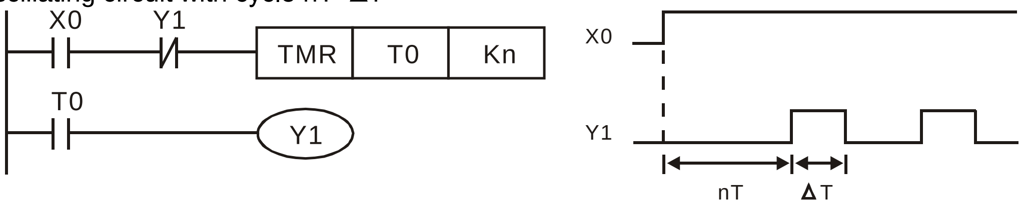
An oscillating circuit with cycle ΔT+ΔT



In the first scan, Y1 turns on. In the second scan, Y1 turns off due to the reversed state of contact Y1. Y1 output status changes in every scan and forms an oscillating circuit with output cycleΔT(ON)+ΔT(OFF)

Example 9 – Oscillating Circuit with Timer

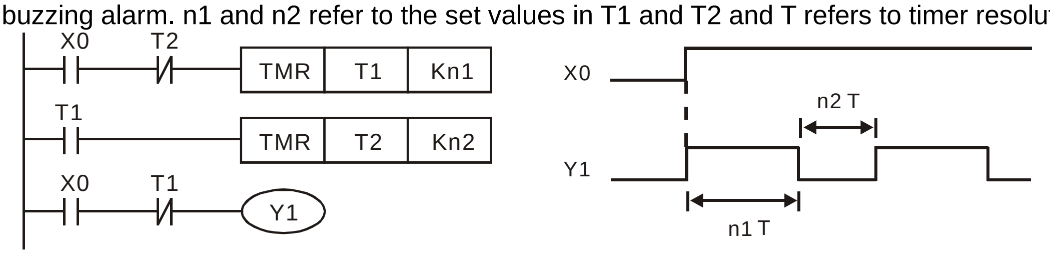
An oscillating circuit with cycle nT+ΔT



When X0 = ON, T0 starts timing (nT). Once the set time is reached, contact T0 = ON to enable Y1(ΔT). In next scan, Timer T0 is reset due to the reversed status of contact Y1. Therefore contact T0 is reset and Y1 = OFF. In next scan, T0 starts timing again. The process forms an oscillating circuit with output cycle nT+ΔT.

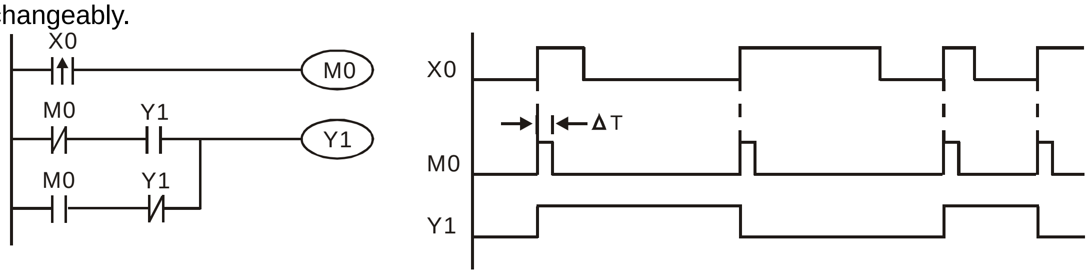
Example 10 - Flashing Circuit

The ladder diagram uses two timers to form an oscillating circuit which enables a flashing indicator or a buzzing alarm. n1 and n2 refer to the set values in T1 and T2 and T refers to timer resolution.



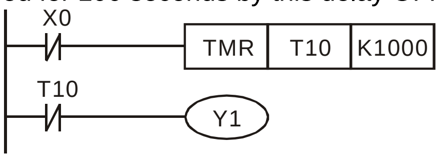
Example 11 - Trigger Circuit

In this diagram, rising-edge contact X0 generates trigger pulses to control two actions executing interchangeably.

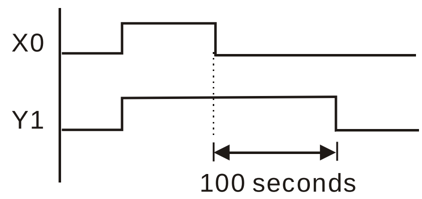


Example 12 - Delay OFF Circuit

If X0 = ON, timer T10 is not energized but coil Y1 is ON. When X0 is OFF, T10 is activated. After 100 seconds (K1000 × 0.1 sec = 100 sec), NC contact T10 is ON to turn off Y1. Turn-off action is delayed for 100 seconds by this delay OFF circuit.

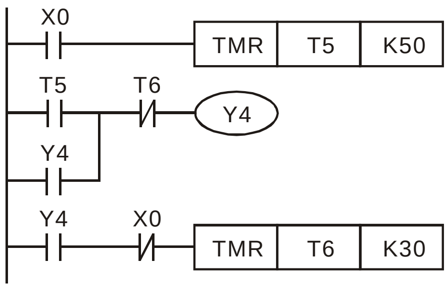


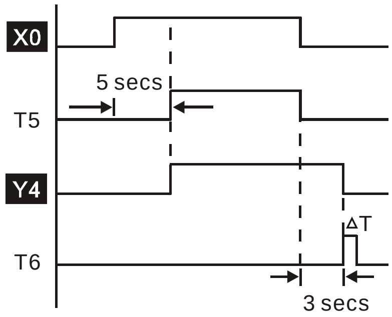
Timer Resolution: 0.1 sec



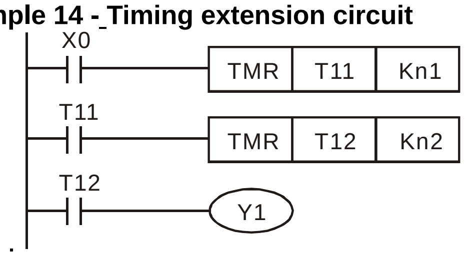
Example 13 - Output delay circuit

The output delay circuit is composed of two timers executing delay actions. No matter input X0 is ON or OFF, output Y4 will be delayed.



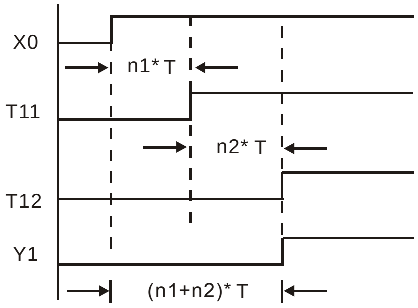


Example 14 - Timing extension circuit

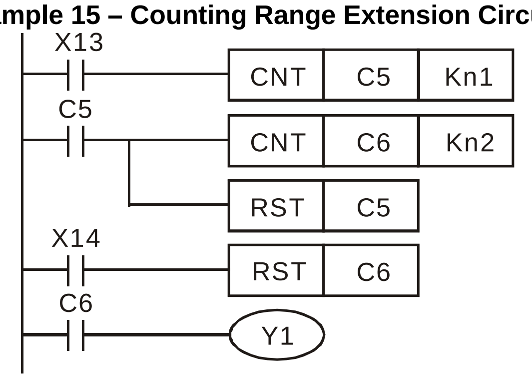


Timer = T11, T12
Timer resolution: T

The total delay time: (n1+n2)* T. T refers to the timer resolution.



Example 15 – Counting Range Extension Circuit

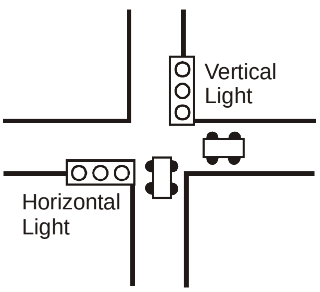


The counting range of a 16-bit counter is 0 ~ 32,767. The opposite circuit uses two counters to increase the counting range as n1*n2. When value in counter C6 reaches n2, The pulses counted from X13 will be n1*n2.

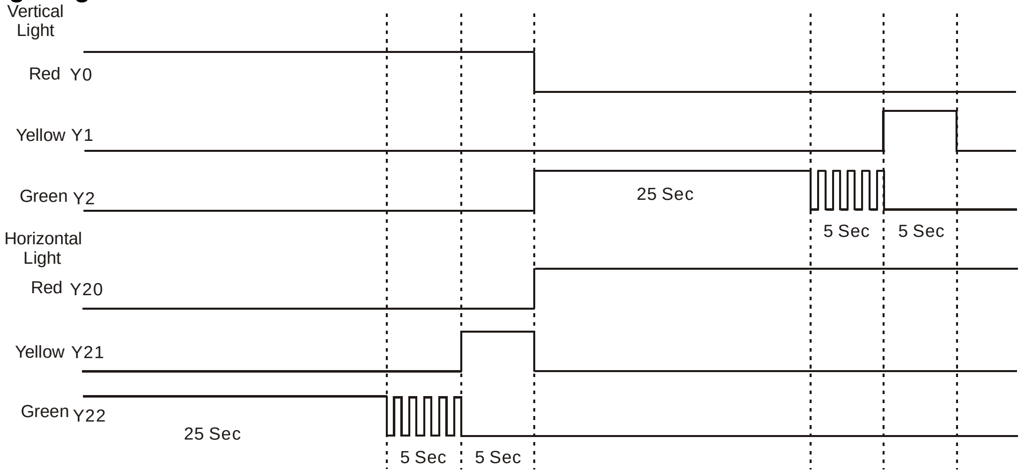
Example 16 - Traffic light control (Step Ladder Logic)

Traffic light control

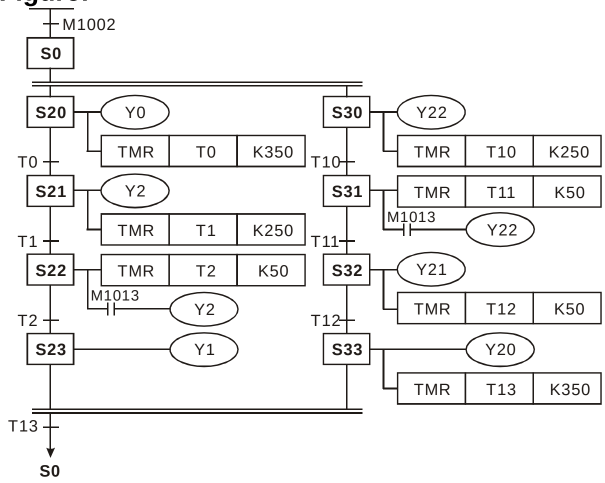
	Red light	Yellow light	Green light	Green light blinking
Vertical light	Y0	Y1	Y2	Y2
Horizontal light	Y20	Y21	Y22	Y22
Light Time	35 Sec	5 Sec	25 Sec	5 Sec



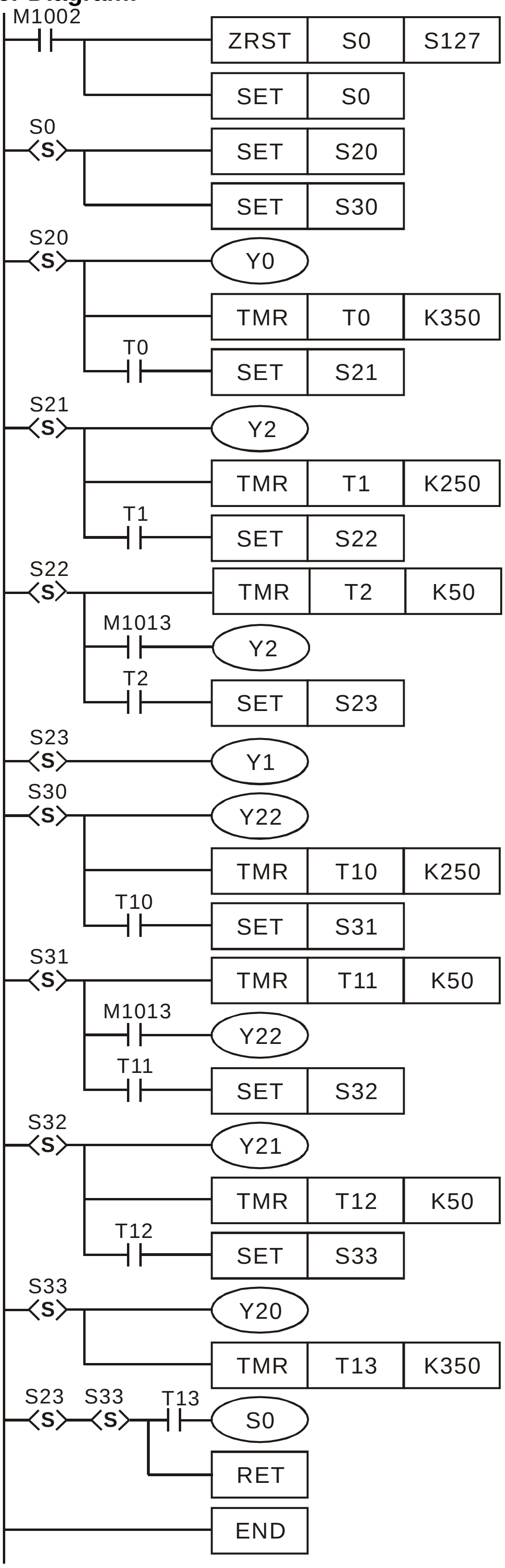
Timing Diagram:



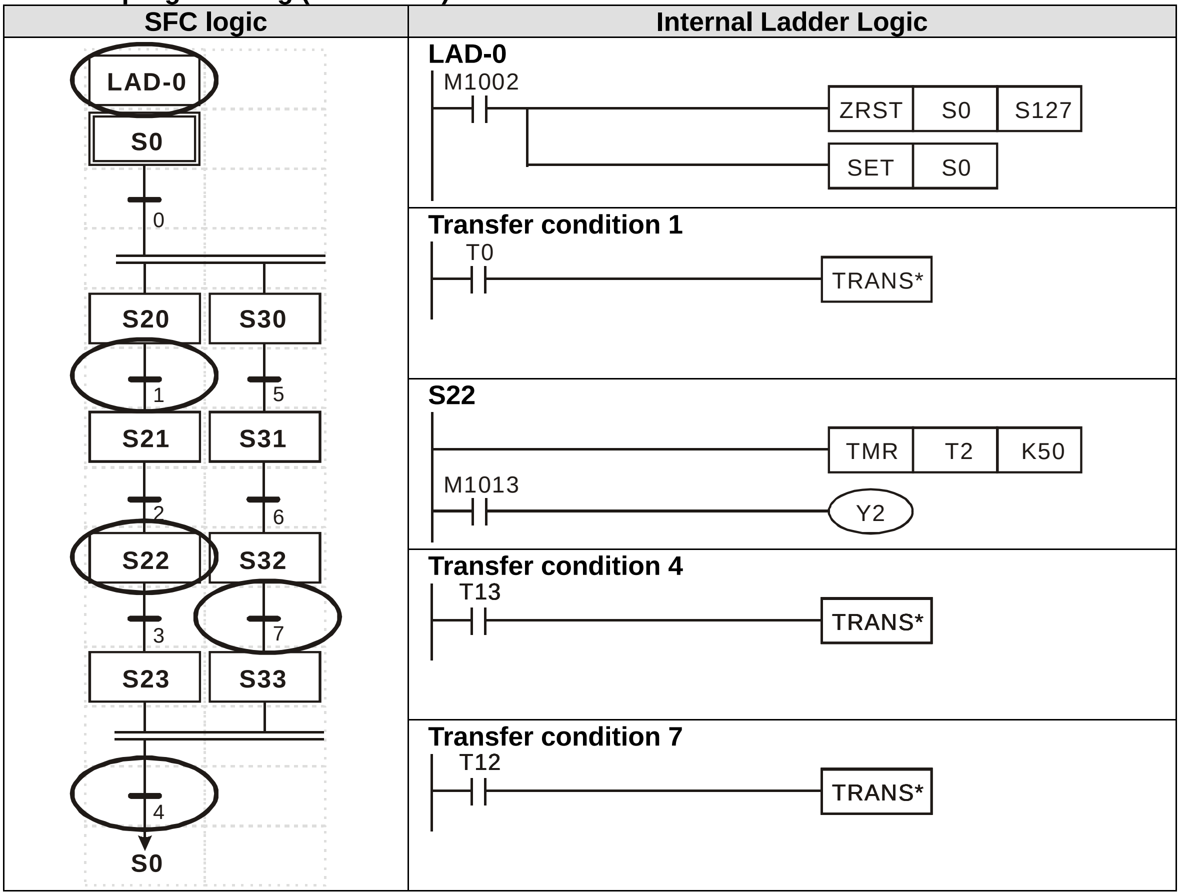
SFC Figure:



Ladder Diagram:



WPLSoft programming (SFC mode)



MEMO

Programming Concepts 2

DVP-ES2/EX2/EC5/SS/SA2/SX2/SE is a programmable logic controller spanning an I/O range of 10–256 I/O points (SS2/SA2/SX2/SE: 512 points). PLC can control a wide variety of devices to solve your automation needs. PLC monitors inputs and modifies outputs as controlled by the user program. User program provides features such as boolean logic, counting, timing, complex math operations, and communications to other communicating products.

Table of Contents

2.1 ES2/EX2/EC5 Memory Map

<table>
<tr><th colspan="7">Specifications</th></tr>
<tr><td colspan="5">Control Method</td><td colspan="2">Stored program, cyclic scan system</td></tr>
<tr><td colspan="5">I/O Processing Method</td><td colspan="2">Batch processing method (when END instruction is executed)</td></tr>
<tr><td colspan="5">Execution Speed</td><td colspan="2">LD instructions – 0.54µs, MOV instructions – 3.4µs</td></tr>
<tr><td colspan="5">Program language</td><td colspan="2">Instruction List + Ladder + SFC</td></tr>
<tr><td colspan="5">Program Capacity</td><td colspan="2">15872 steps</td></tr>
<tr><td rowspan="26">Bit Contacts</td><td>X</td><td colspan="3">External inputs</td><td>X0~X377, octal number system, 256 points max, (*4)</td><td rowspan="2">Total 256+16 I/O</td></tr>
<tr><td>Y</td><td colspan="3">External outputs</td><td>Y0~Y377, octal number system, 256 points max, (*4)</td></tr>
<tr><td rowspan="3">M</td><td rowspan="3">Auxiliary relay</td><td colspan="2">General</td><td>M0~M511, 512 points, (*1)
M768~M999, 232 points, (*1)
M2000~M2047, 48 points, (*1)</td><td rowspan="3">Total 4096 points</td></tr>
<tr><td colspan="2">Latched</td><td>M512~M767, 256 points, (*2)
M2048~M4095, 2048 points, (*2)</td></tr>
<tr><td colspan="2">Special</td><td>M1000~M1999, 1000 points, some are latched</td></tr>
<tr><td rowspan="6">T</td><td rowspan="6">Timer</td><td colspan="2" rowspan="3">100ms (M1028=ON, T64~T126: 10ms)</td><td>T0~T126, 127 points, (*1)
T128~T183, 56 points, (*1)</td><td rowspan="6">Total 256 points</td></tr>
<tr><td>T184~T199 for Subroutines, 16 points, (*1)</td></tr>
<tr><td>T250~T255(accumulative), 6 points (*1)</td></tr>
<tr><td colspan="2" rowspan="2">10ms (M1038=ON, T200~T245: 1ms)</td><td>T200~T239, 40 points, (*1)</td></tr>
<tr><td>T240~T245(accumulative), 6 points, (*1)</td></tr>
<tr><td colspan="2">1ms</td><td>T127, 1 points, (*1)
T246~T249(accumulative), 4 points, (*1)</td></tr>
<tr><td rowspan="9">C</td><td rowspan="9">Counter</td><td colspan="2" rowspan="2">16-bit count up</td><td>C0~C111, 112 points, (*1)
C128~C199,72 points, (*1)</td><td rowspan="4">Total 232 points</td></tr>
<tr><td>C112~C127,16 points, (*2)</td></tr>
<tr><td colspan="2" rowspan="2">32-bit count up/down</td><td>C200~C223, 24 points, (*1)</td></tr>
<tr><td>C224~C231, 8 points, (*2)</td></tr>
<tr><td rowspan="5">32bit high-speed count up/down</td><td rowspan="2">Soft-ware</td><td>C235~C242, 1 phase 1 input, 8 points, (*2)</td><td rowspan="5">Total 22 points</td></tr>
<tr><td>C232~C234, 2 phase 2 input, 3 points, (*2)</td></tr>
<tr><td rowspan="3">Hard-ware</td><td>C243~C244, 1 phase 1 input, 2 points, (*2)</td></tr>
<tr><td>C245~C250, 1 phase 2 input, 6 points, (*2)</td></tr>
<tr><td>C251~C254 2 phase 2 input, 4 points, (*2)</td></tr>
<tr><td rowspan="5">S</td><td rowspan="5">Step point</td><td colspan="2">Initial step point</td><td>S0~S9, 10 points, (*2)</td><td rowspan="5">Total 1024 points</td></tr>
<tr><td colspan="2">Zero point return</td><td>S10~S19, 10 points (use with IST instruction), (*2)</td></tr>
<tr><td colspan="2">Latched</td><td>S20~S127, 108 points, (*2)</td></tr>
<tr><td colspan="2">General</td><td>S128~S911, 784 points, (*1)</td></tr>
<tr><td colspan="2">Alarm</td><td>S912~S1023, 112 points, (*2)</td></tr>
<tr><td rowspan="3">Word Register</td><td>T</td><td colspan="3">Current value</td><td colspan="2">T0~T255, 256 words</td></tr>
<tr><td rowspan="2">C</td><td colspan="3" rowspan="2">Current value</td><td colspan="2">C0~C199, 16-bit counter, 200 words</td></tr>
<tr><td colspan="2">C200~C254, 32-bit counter, 55 words</td></tr>
</table>

Specifications					
	D	Data register	General	D0~D407, 408 words, (*1) D600~D999, 400 words, (*1) D3920~D9999, 6080 words, (*1)	Total 10000 points
			Latched	D408~D599, 192 words, (*2) D2000~D3919, 1920 words, (*2)	
			Special	D1000~D1999, 1000 words, some are latched	
			For Special mudules	D9900~D9999，100 words, (*1), (*5)	
			Index	E0~E7, F0~F7, 16 words, (*1)	
Pointer	N	Master control loop		N0~N7, 8 points	
	P	Pointer		P0~P255, 256 points	
	I	Interrupt Service	External interrupt	I000/I001(X0), I100/I101(X1), I200/I201(X2), I300/I301(X3), I400/I401(X4), I500/I501(X5), I600/I601(X6), I700/I701(X7), 8 points (01: rising-edge trigger , 00: falling-edge trigger)	
			Timer interrupt	I602~I699, I702~I799, 2 points (Timer resolution = 1ms) I805~I899, 1 point (Timer resolution = 0.1ms) (Supported by ES2/EX2: V2.00 and later versions)	
			High-speed counter interrupt	I010, I020, I030, I040, I050, I060, I070, I080,8 points	
			Communication interrupt	I140(COM1), I150(COM2), I160(COM3), 3 points, (*3)	
Constant	K	Decimal		K-32,768 ~ K32,767 (16-bit operation), K-2,147,483,648 ~ K2,147,483,647 (32-bit operation)	
	H	Hexadecimal		H0000 ~ HFFFF (16-bit operation), H00000000 ~HFFFFFFFF (32-bit operation)	
Serial ports				COM1: built-in RS-232 ((Master/Slave); COM2: built-in RS-485 (Master/Slave); COM3: built-in RS-485 (Master/Slave), NOT available for EC5; COM1 is typically the programming port. Ethernet(*8): built-in Ethernet, NOT available for EC5, refer to appendix B for more details on operation	
Real Time Clock(*6)				Year, Month, Day, Week, Hours, Minutes, Seconds	
Special I/O Modules (NOT available for EC5)				Up to 8 special I/O modules can be connected	
File Register(*2) (NOT available for EC5)				K0~K4999, 5000 points K0~K7999, 8000 points (*9)	

Notes:

1. Non-latched area cannot be modified
2. Latched area cannot be modified
3. COM1: built-in RS232 port. COM2: built-in RS485 port. COM3: built-in RS485 port.
4. When input points(X) are expanded to 256 points, only 16 output points(Y) are applicable. Also, when ouput points(Y) are expanded to 256 points, only 16 input points(X) are applicable.
5. This area is applicable only when the ES2/EX2 MPU is connected with special I/O modules. Every special I/O module occupies 10 points.
6. ES2/EX2 with firmware version 2.00 or later support the function of keeping track of the time even after the power is off. When the power is off, this function can go on for about 1 week. This function is available for EC5, but after the power goes off, EC5 stopps keeping track of the time.
7. ES2/EX2 with firmware version 2.00 or later versions support the function of file register. Refer to the instructions MEMR/MEMW for more details on operation.

8. Ethernet: this function is only available for DVP-EX2-E series PLC.
   This function is available for ES2 and EX2 series with firmware V3.46 or later and for EX2-E series with firmware V1.08 or later.

2.2 SS2 Memory Map

Specifications						
Control Method					Stored program, cyclic scan system	
I/O Processing Method					Batch processing method (when END instruction is executed)	
Execution Speed					LD instructions – 0.54μs, MOV instructions – 3.4μs	
Program language					Instruction List + Ladder + SFC	
Program Capacity					7920 steps	
Bit Contacts	X	External inputs			X0~X377, octal number system, 256 points max.	Total 480+14 I/O(*4)
	Y	External outputs			Y0~Y377, octal number system, 256 points max.	
	M	Auxiliary relay	General		M0~M511, 512 points, (*1) M768~M999, 232 points, (*1) M2000~M2047, 48 points, (*1)	Total 4096 points
			Latched		M512~M767, 256 points, (*2) M2048~M4095, 2048 points, (*2)	
			Special		M1000~M1999, 1000 points, some are latched	
	T	Timer	100ms (M1028=ON, T64~T126: 10ms)		T0~T126, 127 points, (*1) T128~T183, 56 points, (*1)	Total 256 points
					T184~T199 for Subroutines, 16 points, (*1)	
					T250~T255(accumulative), 6 points (*1)	
			10ms (M1038=ON, T200~T245: 1ms)		T200~T239, 40 points, (*1)	
					T240~T245(accumulative), 6 points, (*1)	
			1ms		T127, 1 points, (*1) T246~T249(accumulative), 4 points, (*1)	
	C	Counter	16-bit count up		C0~C111, 112 points, (*1) C128~C199, 72 points, (*1)	Total 233 points
					C112~C127, 16 points, (*2)	
			32-bit count up/down		C200~C223, 24 points, (*1)	
					C224~C232, 9 points, (*2)	
			32bit high-speed count up/down	Soft-ware	C235~C242, 1 phase 1 input, 8 points, (*2)	Total 22 points
					C233~C234, 2 phase 2 input, 2 points, (*2)	
				Hard-ware	C243~C244, 1 phase 1 input, 2 points, (*2)	
					C245~C250, 1 phase 2 input, 6 points, (*2)	
					C251~C254 2 phase 2 input, 4 points, (*2)	
	S	Step point	Initial step point		S0~S9, 10 points, (*2)	Total 1024 points
			Zero point return		S10~S19, 10 points (use with IST instruction), (*2)	
			Latched		S20~S127, 108 points, (*2)	
			General		S128~S911, 784 points, (*1)	
			Alarm		S912~S1023, 112 points, (*2)	
Word Register	T	Current value			T0~T255, 256 words	
	C	Current value			C0~C199, 16-bit counter, 200 words	
					C200~C254, 32-bit counter, 55 words	

Specifications					
	D	Data register	General	D0~D407, 408 words, (*1) D600~D999, 400 words, (*1) D3920~D4999, 1080 words, (*1)	Total 5000 points
			Latched	D408~D599, 192 words, (*2) D2000~D3919, 1920 words, (*2)	
			Special	D1000~D1999, 1000 words, some are latched	
			Index	E0~E7, F0~F7, 16 words, (*1)	
Pointer	N	Master control loop		N0~N7, 8 points	
	P	Pointer		P0~P255, 256 points	
	I	Interrupt Service	External interrupt	I000/I001(X0), I100/I101(X1), I200/I201(X2), I300/I301(X3), I400/I401(X4), I500/I501(X5), I600/I601(X6), I700/I701(X7), 8 points (01: rising-edge trigger , 00: falling-edge trigger)	
			Timer interrupt	I602~I699, I702~I799, 2 points (Timer resolution = 1ms) I805~I899, 1 point (Timer resolution = 0.1ms) (Supported by V2.00 and above)	
			High-speed counter interrupt	I010, I020, I030, I040, I050, I060, I070, I080, 8 points	
			Communication interrupt	I140(COM1), I150(COM2), 2 points, (*3)	
Constant	K	Decimal		K-32,768 ~ K32,767 (16-bit operation), K-2,147,483,648 ~ K2,147,483,647 (32-bit operation)	
	H	Hexadecimal		H0000 ~ HFFFF (16-bit operation), H00000000 ~HFFFFFFFF (32-bit operation)	
Serial ports				COM1: built-in RS-232 ((Master/Slave) COM2: built-in RS-485 (Master/Slave) COM1 is typically the programming port.	
Real Time Clock				Year, Month, Day, Week, Hours, Minutes, Seconds; NOT available when the power is off	
Special I/O Modules				Up to 8 special I/O modules can be connected on the right-side of the PLC.	

Notes:

1. Non-latched area cannot be modified
2. Latched area cannot be modified
3. COM1: built-in RS232 port. COM2: built-in RS485 port.
4. The PLC occupies 16 input points (X0~X17) and 16 output points (Y0~Y17). The extension input point starts from X20 and extension output point from Y20.

2.3 SA2/SX2 Memory Map

Specifications						
Control Method					Stored program, cyclic scan system	
I/O Processing Method					Batch processing method (when END instruction is executed)	
Execution Speed					LD instructions – 0.54μs, MOV instructions – 3.4μs	
Program language					Instruction List + Ladder + SFC	
Program Capacity					15872 steps	
Bit Contacts	X	External inputs			X0~X377, octal number system, 256 points max.	Total 480 + points for PLC (*4)
	Y	External outputs			Y0~Y377, octal number system, 256 points max.	
	M	Auxiliary relay	General		M0~M511, 512 points, (*1) M768~M999, 232 points, (*1) M2000~M2047, 48 points, (*1)	Total 4096 points
			Latched		M512~M767, 256 points, (*2) M2048~M4095, 2048 points, (*2)	
			Special		M1000~M1999, 1000 points, some are latched	
	T	Timer	100ms (M1028=ON, T64~T126: 10ms)		T0~T126, 127 points, (*1) T128~T183, 56 points, (*1)	Total 256 points
					T184~T199 for Subroutines, 16 points (*1)	
					T250~T255(accumulative), 6 points (*1)	
			10ms (M1038=ON, T200~T245: 1ms)		T200~T239, 40 points, (*1)	
					T240~T245(accumulative), 6 points, (*1)	
			1ms		T127, 1 points, (*1) T246~T249(accumulative), 4 points, (*1)	
	C	Counter	16-bit count up		C0~C111, 112 points, (*1) C128~C199, 72 points, (*1)	Total 233 points
					C112~C127, 16 points, (*2)	
			32-bit count up/down		C200~C223, 24 points, (*1)	
					C224~C232, 9 points, (*2)	
			32bit high-speed count up/down	Soft-ware	C235~C242, 1 phase 1 input, 8 points, (*2)	Total 22 points
					C233~C234, 2 phase 2 input, 2 points, (*2)	
				Hard-ware	C243~C244, 1 phase 1 input, 2 points, (*2)	
					C245~C250, 1 phase 2 input, 6 points, (*2)	
					C251~C254 2 phase 2 input, 4 points, (*2)	
	S	Step point	Initial step point		S0~S9, 10 points, (*2)	Total 1024 points
			Zero point return		S10~S19, 10 points (use with IST instruction), (*2)	
			Latched		S20~S127, 108 points, (*2)	
			General		S128~S911, 784 points, (*1)	
			Alarm		S912~S1023, 112 points, (*2)	
Word Register	T	Current value			T0~T255, 256 words	
	C	Current value			C0~C199, 16-bit counter, 200 words	
					C200~C254, 32-bit counter, 55 words	

Specifications					
	D	Data register	General	D0~D407, 408 words, (*1) D600~D999, 400 words, (*1) D3920~D9799, 5880 words, (*1)	Total 10000 points
			Latched	D408~D599, 192 words, (*2) D2000~D3919, 1920 words, (*2)	
			Special	D1000~D1999, 1000 words, some are latched	
			Righ-side special module	D9900~D9999, 100 words (*1) (*6)	
			Left-side special module	D9800~D9899, 100 words (*1) (*7) (NOT applicable to DVP28SA2)	
			Index	E0~E7, F0~F7, 16 words, (*1)	
Pointer	N	Master control loop		N0~N7, 8 points	
	P	Pointer		P0~P255, 256 points	
	I	Interrupt Service	External interrupt	I000/I001(X0), I100/I101(X1), I200/I201(X2), I300/I301(X3), I400/I401(X4), I500/I501(X5), I600/I601(X6), I700/I701(X7), 8 points (01: rising-edge trigger , 00: falling-edge trigger)	
			Timer interrupt	I602~I699, I702~I799, 2 points (Timer resolution = 1ms) I805~I899, 1 point (Timer resolution = 0.1ms) (Supported by V2.00 and above)	
			High-speed counter interrupt	I010, I020, I030, I040, I050, I060, I070, I080, 8 points	
			Communication interrupt	I140(COM1), I150(COM2), I160(COM3), 3 points, (*3)	
Constant	K	Decimal		K-32,768 ~ K32,767 (16-bit operation), K-2,147,483,648 ~ K2,147,483,647 (32-bit operation)	
	H	Hexadecimal		H0000 ~ HFFFF (16-bit operation), H00000000 ~HFFFFFFFF (32-bit operation)	
Serial Ports			SA2	COM1: built-in RS-232 ((Master/Slave) COM2: built-in RS-485 (Master/Slave) COM3: built-in RS-485 (Master/Slave) COM1 is typically the programming port.	
			SX2	COM1: built-in RS-232 ((Master/Slave) COM2: built-in RS-485 (Master/Slave) COM3: built-in USB (Slave) COM1 is typically the programming port.	
Real Time Clock*8				Year, Month, Day, Week, Hours, Minutes, Seconds	
Special I/O Modules				Right side: Up to 8 I/O modules can be connected Left side: Up to 8 I/O module can be connected (NOT applicable to DVP28SA2)	
File Register*5				K0~K4999, 5000 points (*2)	

Notes:

1. Non-latched area cannot be modified
2. Latched area cannot be modified
3. Please refer to the table above for more information about serial ports. SX2 does not support I160.
4. The PLC occupies 16 input points (X0~X17) and 16 output points (Y0~Y17). The extension input point starts from X20 and extension output point from Y20.
5. If the firmware version of an MPU is 2.0 or above, the MPU support the use of file registers. Please refer to the instruction MEMR/MEMW for more information about the reading/writing of data.

6. If an SA2/SX2 series MPU is connected to a right-side special module, and M1183 is Off, the range of data registers can be used. Every special module connected to an SA2/SX2 series MPU occupies ten data registers.
7. If an SA2/SX2 series MPU is connected to a left-side special module, and M1182 is Off, the range of data registers can be used. Every special module connected to an SA2/SX2 series MPU occupies ten data registers.
8. SA22/SX2 with firmware version 2.00 or later support the function of keeping track of the time even after the power is off. When the power is off, this function can go on for about 1 week.

2.4 SE Memory Map

Specifications						
Control Method					Stored program, cyclic scan system	
I/O Processing Method					Batch processing method (when END instruction is executed)	
Execution Speed					LD instructions – 0.64μs, MOV instructions – 2μs, 1000 steps – approximately 1ms	
Program language					Instruction List + Ladder diagram+ SFC	
Program Capacity					15872 steps	
Bit Contacts	X	External inputs			X0~X377, octal number system, 256 points max.	Total 480+ points for PLC (*4)
Bit Contacts	Y	External outputs			Y0~Y377, octal number system, 256 points max.	Total 480+ points for PLC (*4)
Bit Contacts	M	Auxiliary relay	General		M0~M511, 512 points, (*1) M768~M999, 232 points, (*1) M2000~M2047, 48 points, (*1)	Total 4096 points
Bit Contacts	M	Auxiliary relay	Latched		M512~M767, 256 points, (*2) M2048~M4095, 2048 points, (*2)	Total 4096 points
Bit Contacts	M	Auxiliary relay	Special		M1000~M1999, 1000 points, some are latched	Total 4096 points
Bit Contacts	T	Timer	100ms (M1028=ON, T64~T126: 10ms)		T0~T126, 127 points, (*1) T128~T183, 56 points, (*1)	Total 256 points
Bit Contacts	T	Timer	100ms (M1028=ON, T64~T126: 10ms)		T184~T199 for Subroutines, 16 points, (*1)	Total 256 points
Bit Contacts	T	Timer	100ms (M1028=ON, T64~T126: 10ms)		T250~T255(accumulative), 6 points (*1)	Total 256 points
Bit Contacts	T	Timer	10ms (M1038=ON, T200~T245: 1ms)		T200~T239, 40 points, (*1)	Total 256 points
Bit Contacts	T	Timer	10ms (M1038=ON, T200~T245: 1ms)		T240~T245(accumulative), 6 points, (*1)	Total 256 points
Bit Contacts	T	Timer	1ms		T127, 1 points, (*1) T246~T249(accumulative), 4 points, (*1)	Total 256 points
Bit Contacts	C	Counter	16-bit count up		C0~C111, 112 points, (*1) C128~C199, 72 points, (*1)	Total 232 points
Bit Contacts	C	Counter	16-bit count up		C112~C127, 16 points, (*2)	Total 232 points
Bit Contacts	C	Counter	32-bit count up/down		C200~C223, 24 points, (*1)	Total 232 points
Bit Contacts	C	Counter	32-bit count up/down		C224~C231, 8 points, (*2)	Total 232 points
Bit Contacts	C	Counter	32bit high-speed count up/down	Soft-ware	C235~C242, 1 phase 1 input, 8 points, (*2)	Total 20 points
Bit Contacts	C	Counter	32bit high-speed count up/down	Soft-ware	C233~C234, 2 phase 2 input, 2 points, (*2)	Total 20 points
Bit Contacts	C	Counter	32bit high-speed count up/down	Hard-ware	C243~C244, 1 phase 1 input, 2 points, (*2)	Total 20 points
Bit Contacts	C	Counter	32bit high-speed count up/down	Hard-ware	C245~C248, 1 phase 2 input, 4 points, (*2)	Total 20 points
Bit Contacts	C	Counter	32bit high-speed count up/down	Hard-ware	C251~C254 2 phase 2 input, 4 points, (*2)	Total 20 points
Bit Contacts	S	Step point	Initial step point		S0~S9, 10 points, (*2)	Total 1024 points
Bit Contacts	S	Step point	Zero point return		S10~S19, 10 points (use with IST instruction), (*2)	Total 1024 points
Bit Contacts	S	Step point	Latched		S20~S127, 108 points, (*2)	Total 1024 points
Bit Contacts	S	Step point	General		S128~S911, 784 points, (*1)	Total 1024 points
Bit Contacts	S	Step point	Alarm		S912~S1023, 112 points, (*2)	Total 1024 points
Word Register	T	Current value			T0~T255, 256 words	
Word Register	C	Current value			C0~C199, 16-bit counter, 200 words	
Word Register	C	Current value			C200~C254, 32-bit counter, 55 words	

Specifications					
	D	Data register	General	D0~D407, 408 words, (*1) D600~D999, 400 words, (*1) D3920~D9799, 5880 words, (*1) D10000~D11999, 2000 words, (*1)	Total 12000 points
	D	Data register	Latched	D408~D599, 192 words, (*2) D2000~D3919, 1920 words, (*2)	Total 12000 points
	D	Data register	Special	D1000~D1999, 1000 words, some are latched	Total 12000 points
	D	Data register	Right-side special module	D9900~D9999, 100 words, (*1) (*5)	Total 12000 points
	D	Data register	Left-side special module	D9800~D9899, 100 words, (*1) (*6) (NOT applicable to DVP26SE)	Total 12000 points
	D	Data register	Index	E0~E7, F0~F7, 16 words, (*1)	Total 12000 points
Pointer	N	Master control loop		N0~N7, 8 points	
Pointer	P	Pointer		P0~P255, 256 points	
Pointer	I	Interrupt Service	External interrupt	I000/I001(X0), I100/I101(X1), I200/I201(X2), I300/I301(X3), I400/I401(X4), I500/I501(X5), I600/I601(X6), I700/I701(X7), 8 points (01: rising-edge trigger , 00: falling-edge trigger)	
Pointer	I	Interrupt Service	Timer interrupt	I602~I699, I702~I799, 2 points (Timer resolution = 1ms) I805~I899, 1 point (Timer resolution = 0.1ms) (Supported by V1.60 and above)	
Pointer	I	Interrupt Service	High-speed counter interrupt	I010, I020, I030, I040, I050, I060, I070, I080, 8 points	
Pointer	I	Interrupt Service	Communication interrupt	I150 (COM2), I160 (COM3), 2 points, (*3)	
Constant	K	Decimal		K-32,768 ~ K32,767 (16-bit operation), K-2,147,483,648 ~ K2,147,483,647 (32-bit operation)	
Constant	H	Hexadecimal		H0000 ~ HFFFF (16-bit operation), H00000000 ~HFFFFFFFF (32-bit operation)	
Serial Ports				COM1: built-in USB (Slave) COM2: built-in RS-485 (Master/Slave) COM3: built-in RS-485 (Master/Slave) Ethernet: built-in Ethernet (Please refer to Appendix B for more information.) COM1 is typically the programming port.	
Real Time Clock				Year, Month, Day, Week, Hours, Minutes, Seconds; When the power is off, this function can still go on for about 1 week.	
Special I/O Modules				Right side: Up to 8 I/O modules can be connected. Left side: Up to 8 high-speed I/O modules can be connected.	

Notes:

1. Non-latched area cannot be modified.
2. Latched area cannot be modified.
3. COM2: built-in RS485 port. COM3: built-in RS485 port.
4. The PLC occupies 16 input points (X0~X17) and 16 output points (Y0~Y17). The extension input point starts from X20 and extension output point from Y20.
5. If an SE series MPU is connected to a right-side special module, and M1183 is Off, the data registers can be used. Each connected special module occupies ten data registers.
6. If an SE series MPU is connected to a left-side special module, and M1182 is Off, the data registers can be used. Each connected special module occupies ten data registers.

2.5 Status and Allocation of Latched Memory

<table>
<tr><th>Memory type</th><th>Power OFF=>ON</th><th>STOP=>RUN</th><th>RUN=>STOP</th><th>Clear all non-latched areas (M1031=ON)</th><th>Clear all latched areas (M1032=ON)</th><th>Factory setting</th></tr>
<tr><td rowspan="2">Non-latched</td><td rowspan="2">Clear</td><td rowspan="2">Unchanged</td><td>When M1033=OFF, clear</td><td rowspan="2">Clear</td><td rowspan="2">Unchanged</td><td rowspan="2">0</td></tr>
<tr><td>When M1033=ON, No change</td></tr>
<tr><td>Latched</td><td colspan="3">Unchanged</td><td>Unchanged</td><td>Clear</td><td>0</td></tr>
<tr><td>Special M, Special D, Index register</td><td>Initial</td><td colspan="2">Unchanged</td><td colspan="2">Unchanged</td><td>Initial setting</td></tr>
<tr><td>File register</td><td colspan="5">Unchanged</td><td>HFFFF</td></tr>
</table>

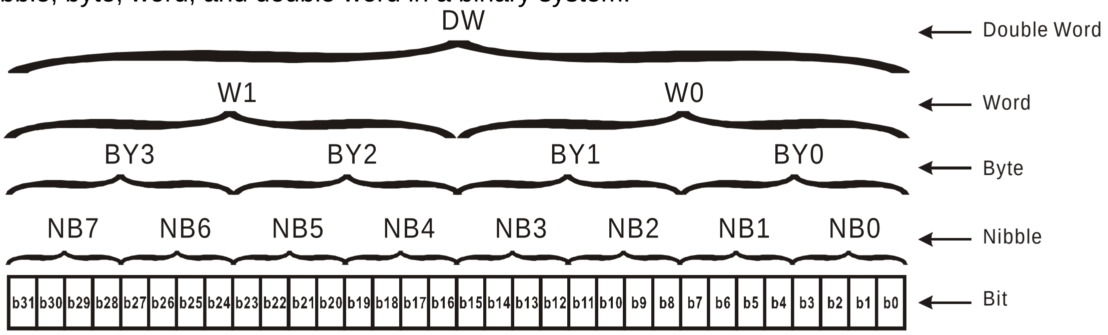
<table>
<tr><td rowspan="3">M
Auxiliary relay</td><td colspan="3">General</td><td colspan="5">Latched</td><td colspan="5">Special auxiliary relay</td></tr>
<tr><td colspan="3">M0~M511
M768~M999
M2000~M2047</td><td colspan="5">M512~M999
M2048~M4095</td><td colspan="5">M1000~M1999</td></tr>
<tr><td colspan="3">Not latched</td><td colspan="5">Latched</td><td colspan="5">Some are latched and can't be changed.</td></tr>
<tr><td rowspan="4">T
Timer</td><td>100 ms</td><td colspan="2">100 ms</td><td colspan="2">1 ms</td><td colspan="2">10 ms</td><td colspan="2">10ms</td><td colspan="3">1 ms</td><td>100 ms</td></tr>
<tr><td>T0 ~T126
T128~T183</td><td colspan="2">T184~T199</td><td colspan="2">T127</td><td colspan="2">T200~T239</td><td colspan="2">T240~T245</td><td colspan="3">T246~T249</td><td>T250~T255</td></tr>
<tr><td>M1028=1,T64~T126:10ms</td><td colspan="2">For subroutine</td><td colspan="2">-</td><td colspan="4">M1038=1,T200~T245: 1ms</td><td colspan="4">-</td></tr>
<tr><td>non-latched</td><td colspan="6">non-latched</td><td colspan="6">Accumulative non-latched</td></tr>
<tr><td rowspan="3">C
Counter</td><td colspan="4">16-bit count up</td><td colspan="6">32-bit count up/down</td><td colspan="3">32-bit high-speed count up/down</td></tr>
<tr><td>C0~C111
C128~C199</td><td colspan="3">C112~C127</td><td colspan="3">C200~C223</td><td colspan="3">C224~C232</td><td colspan="3">C233~C254</td></tr>
<tr><td>Non-latched</td><td colspan="3">Latched</td><td colspan="3">Non-latched</td><td colspan="3">Latched</td><td colspan="3">Latched</td></tr>
<tr><td rowspan="3">S
Step relay</td><td>Initial</td><td colspan="3">Zero return</td><td colspan="3">Latched</td><td colspan="3">General</td><td colspan="3">Step alarm</td></tr>
<tr><td>S0~S9</td><td colspan="3">S10~S19</td><td colspan="3">S20~S127</td><td colspan="3">S128~S911</td><td colspan="3">S912~S1023</td></tr>
<tr><td colspan="7">Latched</td><td colspan="3">Non-latched</td><td colspan="3">Latched</td></tr>
<tr><td rowspan="3">D
Register</td><td colspan="2">General</td><td colspan="4">Latched</td><td colspan="5">Special register</td><td colspan="2">For AIO</td></tr>
<tr><td colspan="2">D0~D407
D600~D999
D3920~D11999</td><td colspan="4">D408~D599
D2000~D3919</td><td colspan="5">D1000~D1999</td><td colspan="2">D9800~D9999</td></tr>
<tr><td colspan="2">Non-latched</td><td colspan="4">Latched</td><td colspan="5">Some are latched, and can't be changed</td><td colspan="2">Non-latched</td></tr>
</table>

2.6 PLC Bits, Nibbles, Bytes, Words, etc

For different control purposes, there are five types of values inside DVP-PLC for executing the operations.

Numeric	Description
Bit	Bit is the basic unit of a binary number system. Range is 0 or 1
Nibble	Consists of 4 consecutive bits, e.g. b3~b0. Range 0 ~ 9 in Decimal or 0~F in Hex
Byte	Consists of 2 consecutive nibbles, e.g. b7~b0. Range 00 ~ FF in Hex
Word	Consists of 2 consecutive bytes, e.g. b15~b0. Range 0000 ~ FFFF in Hex
Double Word	Consists of 2 consecutive words, e.g. b31~b1. Range 00000000 - FFFFFFFF in Hex

Bit, nibble, byte, word, and double word in a binary system:



2.7 Binary, Octal, Decimal, BCD, Hex

For fulllfilling different kinds of internal manipulation, DVP-PLC appies 5 foramts of number systems. Each number system has its specific purpose and function described as below.

1. Binary Number, (BIN)
   PLC internally calculates, operates, and stores the value in Binary format.
2. Octal Number, (OCT)
   The external I/O points of DVP-PLC are numbered in octal format.
   e.g.
   External inputs: X0～X7, X10～X17, …, X377. (No. of device)
   External outputs: Y0～Y7, Y10～Y17, …, Y377. (No. of device)
3. Decimal Number, (DEC)
   DVP-PLC appies decimal operation in situations below:
   - Set value for timers and counters, e.g. TMR C0 K50. (K value)
   - No. of S, M, T, C, D, E, F, P, I devices, e.g. M10, T30. (No. of device)
   - For use of operand in API instructions, e.g. MOV K123 D0. (K value)
4. BCD (Binary Coded Decimal)
   BCD format takes 1 digit or 4 bits to indicate a Decimal value, so that data of consecutive 16 bits indicates a 4-digit decimal value. Used mainly for reading values from DIP switches or sending data to 7-segement displays
5. Hexadecimal Number, HEX
   DVP-PLC appies Hexadecimal operation in situations below:
   - For use of operand in API instructions, e.g. MOV H1A2B D0。(H value)

   Constant (K): A decimal number in a PLC is generally preceded by K. For example, K100 represents the decimal number 100.

Exception:
If K is used with an X/Y/M/S device, a nibble device, a byte device, a word device, or a double word device will be formed.
Example:
K1Y10 represents a device composed of 4 bits, K2Y10 represents a device composed of 8 bits, K3Y10 represents a device composed of 12 bit, and K4Y10 represents a device composed of 16 bits. K1M100 represents a device composed of 4 bits, K2M100 represents a device composed of 8 bits, K3M100 represents a device composed of 12 bit, and K4M100 represents a device composed of 16 bits.

Constant (H): A hexadecimal number in a PLC is generally preceded by H. For example, the hexadecimal number H100 represents the decimal number 256.

Reference Table:

Binary (BIN)	**Octal (OCT)**	**Decimal (K) (DEC)**	**BCD (Binary Code Decimal)**	**Hexadecimal (H) (HEX)**
For PLC internal operation	No. of X, Y relay	Costant K, No. of registers M, S, T, C, D, E, F, P, I devices	For DIP Switch and 7-segment display	Constant H
0000	0	0	0000	0
0001	1	1	0001	1
0010	2	2	0010	2
0011	3	3	0011	3
0100	4	4	0100	4
0101	5	5	0101	5
0110	6	6	0110	6
0111	7	7	0111	7
1000	10	8	1000	8
1001	11	9	1001	9
1010	12	10	0000	A
1011	13	11	0001	B
1100	14	12	0010	C
1101	15	13	0011	D
1110	16	14	0100	E
1111	17	15	0101	F
10000	20	16	0110	10
10001	21	17	0111	11

2.8 M Relay

The types and functions of special auxiliary relays (special M) are listed in the table below. Care should be taken that some devices of the same No. may bear different meanings in different series MPUs. Special M and special D marked with "*" will be further illustrated in 2.13. Columns marked with "R" refers to "read only", "R/W" refers to "read and write", "-" refers to the status remains unchanged and "#" refers to that system will set it up according to the status of the PLC.

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1000*	Monitor normally open contact	○	○	○	○	OFF	ON	OFF	R	NO	OFF
M1001*	Monitor normally closed contact	○	○	○	○	ON	OFF	ON	R	NO	ON
M1002*	Enable single positive pulse at the moment when RUN is activate (Normally OFF)	○	○	○	○	OFF	ON	OFF	R	NO	OFF
M1003*	Enable single negative pulse at the moment when RUN is activate (Normally ON)	○	○	○	○	ON	OFF	ON	R	NO	ON
M1004*	ON when syntax errors occur	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1008*	Watchdog timer (ON: PLC WDT time out)	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1009	Indicate LV signal due to 24VDC insufficiency	○	○	○	○	OFF	-	-	R	NO	OFF
M1011*	10ms clock pulse, 5ms ON/5ms OFF	○	○	○	○	OFF	-	-	R	NO	OFF
M1012*	100ms clock pulse, 50ms ON / 50ms OFF	○	○	○	○	OFF	-	-	R	NO	OFF
M1013*	1s clock pulse, 0.5s ON / 0.5s OFF	○	○	○	○	OFF	-	-	R	NO	OFF
M1014*	1 min clock pulse, 30s ON / 30s OFF	○	○	○	○	OFF	-	-	R	NO	OFF
M1015*	Enable high-speed timer	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1016*	Indicate Year display mode of RTC.	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1017*	±30 seconds correction on real time clock	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1018	Flag for Radian/Degree, ON for degree	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1019*	If the PLC detects the external 24V voltage is unstable; OFF: the PLC runs after the power is stabilized, ON: the error LED keeps flashing	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1020	Zero flag	○	○	○	○	OFF	-	-	R	NO	OFF
M1021	Borrow flag	○	○	○	○	OFF	-	-	R	NO	OFF
M1022	Carry flag	○	○	○	○	OFF	-	-	R	NO	OFF
M1024	COM1 monitor request	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1025*	Indicate incorrect request for communication	○	○	○	○	OFF	-	-	R	NO	OFF
M1026	RAMP mode selection	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1027	PR output mode selection (8/16 bytes)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1028	Switch T64~T126 timer resulotion (10ms/100ms). ON =10ms	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1029*	CH0 (Y0, Y1) pulse output execution completed.	○	○	○	○	OFF	-	-	R	NO	OFF
M1030*	Pulse output Y1 execution completed	ES2/ EX2	○	○	○	OFF	-	-	R	NO	OFF
M1031*	Clear all non-latched memory	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1032*	Clear all latched memory	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1033*	Output state latched at STOP	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1034*	Disable all Y outputs	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1035*	Enable X7 input point as RUN/STOP switch	○	○	○	○	-	-	-	R/W	YES	OFF
M1037*	Enable 8-sets SPD function (Has to be used with D1037) Refer to section 2.16 the description on M1037 for more information on availability.	ES2/ EX2	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1038	Switch T200~T255 timer resulotion (10ms/1ms). ON = 1ms	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1039*	Fix scan time	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1040	Disable step transition	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1041	Step transition start	○	○	○	○	OFF	-	OFF	R/W	NO	OFF
M1042	Enable pulse operation	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1043	Zero return completed	○	○	○	○	OFF	-	OFF	R/W	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1044	Zero point condition	○	○	○	○	OFF	-	OFF	R/W	NO	OFF
M1045	Disable "all output reset" function	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1046	Indicate STL status	○	○	○	○	OFF	-	-	R	NO	OFF
M1047	Enable STL monitoring	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1048	Indicate alarm status	○	○	○	○	OFF	-	-	R	NO	OFF
M1049	Enable alarm monitoring	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1050	Disable interruption I000 / I001	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1051	Disable interruption I100 / I101	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1052	Disable interruption I200 / I201	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1053	Disable interruption I300 / I301	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1054	Disable interruption I400 / I401	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1055	Disable interruption I500 / I501	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1056	Disable interruption I600~I699	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1057	Disable interruption I700~I799 Disable interruption I805~I899 (V2.00 and above are supported.)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1058	COM3 monitor request	ES2/ EX2	╳	○	○	OFF	-	-	R/W	NO	OFF
M1059	Disable high-speed counter interruptions I010~I080	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1060	System error message 1	○	○	○	○	OFF	-	-	R	NO	OFF
M1061	System error message 2	○	○	○	○	OFF	-	-	R	NO	OFF
M1062	System error message 3	○	○	○	○	OFF	-	-	R	NO	OFF
M1063	System error message 4	○	○	○	○	OFF	-	-	R	NO	OFF
M1064	Incorrect use of operands	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1065	Syntax error	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1066	Loop error	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1067*	Program execution error	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1068*	Execution error locked (D1068)	○	○	○	○	OFF	-	-	R	NO	OFF
M1070	Switching clock pulse of Y1 for PWM instruction (ON: 100us; OFF: 1ms)	ES2/ EX2	○	○	○	OFF	-	-	R/W	NO	OFF
M1071	Switching clock pulse of Y3 for PWM instruction (ON: 100us; OFF: 1ms)	ES2/ EX2	○	○	○	OFF	-	-	R/W	NO	OFF
M1072	PLC status (RUN/STOP), ON = RUN	○	○	○	○	OFF	ON	OFF	R/W	NO	OFF
M1075	Error occurring when write in Flash ROM	○	○	○	○	OFF	-	-	R	NO	OFF
M1077*	Real-time clock is out of order.	ES2/ EX2	○	○	○	OFF	-	-	R	NO	OFF
M1078	Y0/CH0(Y0, Y1) pulse output pause (immediate)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1079	Y1 pulse output pause (immediate)	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1080	COM2 monitor request	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1081	Changing conversion mode for FLT instruction	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1083*	Selecting X6 pulse-width detecting mode. M1083 = ON, detecting pulse-width when X6 = ON; M1083 = OFF, detecting pulse-width when X6 = OFF.	ES2/ EX2	○	○	○	OFF	-	-	R/W	NO	OFF
M1084*	Enabling X6 Pulse width detecting function. (has to be used with M1083 and D1023)	ES2/ EX2	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1085	Selecting DVP-PCC01 duplicating function	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1086	Enabling password function for DVP-PCC01	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1088	Matrix comparison. Comparing between equivalent values (M1088 = ON) or different values (M1088 = OFF).	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1089	Indicating the end of matrix comparison. When the comparison reaches the last bit, M1089 = ON.	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1090	Indicating start of matrix comparison. When the comparison starts from the first bit, M1090 = ON.	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1091	Indicating matrix searching results. When the comparison has matched results,	○	○	○	○	OFF	OFF	-	R	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
	comparison will stop immediately and M1091 = ON.										
M1092	Indicating pointer error. When the pointer Pr exceeds the comparison range, M1092 = ON	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1093	Matrix pointer increasing flag. Adding 1 to the current value of the Pr.	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1094	Matrix pointer clear flag. Clear the current value of the Pr to 0	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1095	Carry flag for matrix rotation / shift / output.	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1096	Borrow flag for matrix rotation/shift/input	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1097	Direction flag for matrix rotation/displacement	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1098	Counting the number of bits which are “1” or “0”	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1099	ON when the bits counting result is “0”	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1102*	Y2/CH1 (Y2, Y3) pulse output execution completed	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1103*	Y3 pulse output completed	ES2/ EX2	○	○	○	OFF	-	-	R/W	NO	OFF
M1104	Y2/CH1 (Y2, Y3) pulse output pause (immediate)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1105	Y3 pulse output pause (immediate)	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1106	When M1106=ON, set the zero point for the first DZRN auto-reset (Y0, Y1) to the position wher DOG point stops on the positive direction. (If M1106 is OFF, it stops on the negative direction.)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1107	When M1107=ON, set the zero point for the first DZRN auto-reset (Y0, Y1) to the position wher DOG point stops on the positive direction. (If M1107 is OFF, it stops on the negative direction.)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1108	Y0/CH0 (Y0, Y1) pulse output pause (ramp down)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1109	Y1 pulse output pause (ramp down)	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1110	Y2/CH1 (Y2, Y3) pulse output pause (ramp down)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1111	Y3 pulse output pause (ramp down)	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1112	Switching clock pulse of Y0 for PWM instruction (ON: 100us; OFF: 1ms)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1113	Switching clock pulse of Y2 for PWM instruction (ON: 100us; OFF: 1ms)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1116	Enable high-speed output of Y0/Y2 for PWM instruction (unit: 1 us) available for ES2: V3.00, SS2: V2.80, SA2: V2.60, 12SE: V2.02, 26SE: V1.02, SX2: V2.40 or later	ES2/ EX2	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1117	Enable high-speed output of Y1/Y3 for PWM instruction (unit: 1 us) available for ES2: V3.00, SS2: V2.80, SA2: V2.60, 12SE: V2.02, 26SE: V1.02, SX2: V2.40 or later	ES2/ EX2	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1119*	Enable 2-speed output function of DDRVI/DDRVA instructions; refer to section 2.16 for the usage	○	○	SA2 26SE	○	OFF	OFF	OFF	R/W	NO	OFF
M1120*	Retaining the communication setting of COM2 (RS-485), modifying D1120 will be invalid when M1120 is set.	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1121	For COM2(RS-485), data transmission ready	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1122	For COM2(RS-485), sending request	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1123	For COM2(RS-485), data receiving completed	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1124	For COM2(RS-485), data receiving ready	○	○	○	○	OFF	OFF	-	R/W	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
M1125	For COM2(RS-485), communication ready status reset	○	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1126	For COM2(RS-485), set STX/ETX as user defined or system defined	○	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1127	For COM2(RS-485), data sending / receiving / converting completed. (RS instruction is not supported)	○	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1128	For COM2(RS-485), Transmitting/Receiving status Indication	○	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1129	For COM2(RS-485), receiving time out	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1130	For COM2(RS-485), STX/ETX selection	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1131	For COM2(RS-485), ON when MODRD/RDST/MODRW data is being converted from ASCII to Hex	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1132	ON when there are no communication related instructions in the program	○	○	○	○	OFF	-	-	R	NO	OFF
M1133	Pulse output ramp-down stop for Y4/CH2 (Y4, Y5)	EC5	╳	╳	╳	OFF	OFF	-	R/W	NO	OFF
M1134	Pulse output ramp-down stop for Y6/CH3 (Y6, Y7)	EC5	╳	╳	╳	OFF	OFF	-	R/W	NO	OFF
M1136*	For COM3(RS-485/USB), retaining communication setting	○	╳	○	○	OFF	-	-	R/W	NO	OFF
M1137	Retain DNET mapping data during non-executing period	╳	╳	○	○	-	-	-	R/W	NO	OFF
M1138*	For COM1 (RS-232), retaining communication setting. Modifying D1036 will be invalid when M1138 is set.	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1139*	For COM1(RS-232), ASCII/RTU mode selection (OFF: ASCII; ON: RTU)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1140	For COM2 (RS-485), MODRD / MODWR / MODRW data receiving error	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1141	For COM2 (RS-485), MODRD / MODWR / MODRW parameter error	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1142	Data receiving error of VFD-A handy instructions	○	○	○	○	OFF	OFF	-	R	NO	OFF
M1143*	For COM2(RS-485), ASCII/RTU mode selection (OFF: ASCII; ON: RTU)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1145*	Read MAC address from the left side network module (should work with D1400~1403); available for 12SA2: V3.00, 12SE: V2.00, 20SX2: V3.00	╳	╳	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1148	After the instruction DELAY is executed, the execution of the program following DELAY is delayed for 5 us. (ES2/EX2: V3.2; EC5: V1.00)	○	V3.0	V2.6 V1.4	V2.4	OFF	OFF	-	R/W	NO	OFF
M1156*	Enabling the mask and alignment mark function on I400/I401(X4) corresponding to Y0	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1157*	Enabling the mask and alignment mark function on I500/I501(X5) corresponding to Y1 (NOT applicable to ES2/EX2: V3.41; EC5)	ES2/ EX2	╳	╳	╳	OFF	OFF	-	R/W	NO	OFF
M1158*	Enabling the mask and alignment mark function on I600/I601(X6) corresponding to Y2	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1159*	Enabling the mask and alignment mark function on I700/I701(X7) corresponding to Y3 (NOT applicable to ES2/EX2: V3.41; EC5)	ES2/ EX2	╳	╳	╳	OFF	OFF	-	R/W	NO	OFF
M1161	8/16 bit mode (ON = 8 bit mode)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1162	Switching between decimal integer and binary floating point for SCLP instruction. ON: binary floating point; OFF: decimal integer	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1167	16-bit mode for HKY input	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1168	Designating work mode of SMOV	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1177	Enable the communication instruction for Delta VFD series inverter.	○	○	○	○	OFF	-	-	R/W	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
	ON: VFD-A (Default), OFF: other models of VFD										
M1178	Enable knob VR0	×	×	○	○	OFF	-	-	R/W	NO	OFF
M1179	Enable knob VR1	×	×	○	○	OFF	-	-	R/W	NO	OFF
M1180	The EX2/SX2 model reads analog-to-digital values immediately.	EX2	×	×	○	OFF	-	-	R/W	NO	OFF
M1181	The EX2/SX2 model outputs digital-to-analog values immediately.	EX2	×	×	○	OFF	-	-	R/W	NO	OFF
M1182*	M1182 = ON, disable auto-mapping function when connected with left-side modules. ■ For SA2 /SX2/SE models, values of AIO modules will be auto-mapped to D9800 and above. ■ If the left side is connected with a communication module, additional 10 words will be occupied. Ex: 04AD-SL + EN01-SL + SA2, average value of Ch1~Ch4 of 04AD-SL maps to D9810~D9813.	×	×	○	○	ON	-	-	R/W	NO	ON
M1183*	M1183 = ON, disable auto mapping function when connected with special modules #: ES2/EX2: OFF; SE/SA2/SX2: ON (maps to D9900 and later)	ES2/ EX2	×	○	○	#	-	-	R/W	NO	#
M1190	Set Y0 high speed output as 0.01 ~ 10Hz; not available for 12SE	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1191	Set Y1 high speed output as 0.01 ~ 10Hz; not available for 12SE	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1192	Set Y2 high speed output as 0.01 ~ 10Hz; not available for 12SE	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1193	Set Y3 high speed output as 0.01 ~ 10Hz; not available for 12SE	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1196	Keep connectivity flag for ETHRS instrucitons; available for ES2-E: V1.08, 26SE: V1.0, 12SE: V1.88 or later	ES2-E	×	SE	×	OFF	OFF	-	R/W	NO	OFF
M1197	In execution flag for ETHRS instrucitons; available for ES2-E: V1.08, 26SE: V1.0, 12SE: V1.88 or later	ES2-E	×	SE	×	OFF	OFF	-	R/W	NO	OFF
M1198	Error flag for ETHRS instrucitons; available for ES2-E: V1.08, 26SE: V1.0, 12SE: V1.88 or later	ES2-E	×	SE	×	OFF	OFF	-	R/W	NO	OFF
M1200	C200 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1201	C201 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1202	C202 counting mode ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1203	C203 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1204	C204 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1205	C205 counting mode (ON :count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1206	C206 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1207	C207 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1208	C208 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1209	C209 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1210	C210 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1211	C211 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1212	C212 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1213	C213 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1214	C214 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1215	C215 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1216	C216 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1217	C217 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1218	C218 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1219	C219 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1220	C220 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1221	C221 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1222	C222 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
M1223	C223 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1224	C224 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1225	C225 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1226	C226 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1227	C227 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1228	C228 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1229	C229 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1230	C230 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1231	C231 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1232	C232 counting mode (ON: count down)	╳	○	╳	╳	OFF	-	-	R/W	NO	OFF
	C232 counter monitor (ON: count down)	○	╳	○	○	OFF	-	-	R	NO	OFF
M1233	C233 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1234	C234 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1235	C235 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1236	C236 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1237	C237 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1238	C238 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1239	C239 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1240	C240 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1241	C241 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1242	C242 counting mode (ON: count down)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1243	C243 Reset function control. ON = R function disabled	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1244	C244 Reset function control. ON = R function disabled	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1245	C245 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1246	C246 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1247	C247 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1248	C248 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1249	C249 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1250	C250 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1251	C251 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1252	C252 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1253	C253 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1254	C254 counter monitor (ON: count down)	○	○	○	○	OFF	-	-	R	NO	OFF
M1257	Set the ramp up/down of Y0, Y2 to be "S curve." ON = S curve.	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1260	Set up X7 as the reset signal for software counters C235 ~ C241	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1262	Enable cyclic output for table output function of DPTPO instruction. ON = enable.	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1263	Execute RS instruction with M1263 to set the data receiving as completed, when the data receiving stops for a period of time that is longer than what D1168 was set. Refer to RS instruction for more details.	○	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1270	C235 counting mode (ON: falling-edge count)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1271	C236 counting mode ON: falling-edge count)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1272	C237 counting mode (ON: falling-edge count)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1273	C238 counting mode (ON: falling-edge count)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1274	C239 counting mode (ON: falling-edge count)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1275	C240 counting mode (ON: falling-edge count)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1276	C241 counting mode (ON: falling-edge count)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1277	C242 counting mode (ON: falling-edge count)	○	○	○	○	OFF	-	-	R/W	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
M1280*	For I000 / I001, reverse interrupt trigger pulse direction (Rising/Falling)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1284*	For I400 / I401, reverse interrupt trigger pulse direction (Rising/Falling)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1286*	For I600 / I601, reverse interrupt trigger pulse direction (Rising/Falling)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1303	High / low bits exchange for XCH instruction	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1304*	Enable force-ON/OFF of input point X	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1305	Reverse Y1 pulse output direction in high speed pulse output instructions	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1306	Reverse Y3 pulse output direction in high speed pulse output instructions	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1307	For ZRN instruction, enable left limit switch	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1308*	Output specified pulses or seek Z phase signal when zero point is achieved.	○	○	○	○	OFF	OFF	OFF	R/W	NO	OFF
M1310	Pulse output immediate pause for Y4/CH2 (Y4, Y5)	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1311	Pulse output immediate pause for Y6/CH3 (Y6, Y7)	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1312	For COM1(RS-232), sending request (Only applicable for MODRW and RS instruction)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1313	For COM1(RS-232), ready for data receiving (Only applicable for MODRW and RS instruction)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1314	For COM1(RS-232), data receiving completed (Only applicable for MODRW and RS instruction)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1315	For COM1(RS-232), data receiving error (Only applicable for MODRW and RS instruction)	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1316	For COM3(RS-485), sending request (Only applicable for MODRW and RS instruction)	ES2/ EX2	×	○	×	OFF	OFF	-	R/W	NO	OFF
M1317	For COM3(RS-485), ready for data receiving (Only applicable for MODRW and RS instruction)	ES2/ EX2	×	○	×	OFF	OFF	-	R/W	NO	OFF
M1318	For COM3(RS-485), data receiving completed (Only applicable for MODRW and RS instruction)	ES2/ EX2	×	○	×	OFF	OFF	-	R/W	NO	OFF
M1319	For COM3(RS-485), data receiving error (Only applicable for MODRW and RS instruction)	ES2/ EX2	×	○	×	OFF	OFF	-	R/W	NO	OFF
M1320*	For COM3 (RS-485), ASCII/RTU mode selection. (OFF: ASCII; ON: RTU)	ES2/ EX2	×	○	×	OFF	-	-	R/W	NO	OFF
M1321*	pulse output execution completed for Y4/CH2 (Y4, Y5)	EC5	×	×	×	OFF	-	-	R/W	NO	OFF
M1322*	pulse output execution completed for Y6/CH3 (Y6, Y7)	EC5	×	×	×	OFF	-	-	R/W	NO	OFF
M1334*	Close the conditional contact and to enable the instructions PLSR, DPLSR Y0, DDRVI, DDRVA CH0(Y0/Y1) to execute ramp-down (ON: Enable; OFF: Disable) (available for ES2/EX2: V3.42, ES2-C: V3.48, ES2-E: V1.00, SS2: V3.28, 12SA2: V2.86, 26SE: V1.0, 12SE: V2.02, SX2: V2.86, 28SA2: V3.0 or later)	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1335*	Close the conditional contact and to enable the instructions PLSR/DPLSR Y2/DDRVI/DDRVA CH1(Y2/Y3) (ON: Enable; OFF: Disable) (available for ES2/EX2: V3.42, ES2-C: V3.48, ES2-E:	○	○	○	○	OFF	-	-	R/W	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
	V1.00, SS2: V3.28, 12SA2: V2.86, 26SE: V1.0, 12SE: V2.02, SX2: V2.86, 28SA2: V3.0 or later)										
M1346*	Output clear signals when ZRN is completed	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1347	Auto-reset Y0 when high speed pulse output is completed	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1348	Auto-reset Y1 when high speed pulse output is completed	ES2/ EX2	○	○	○	OFF	-	-	R/W	NO	OFF
M1349	When M1349 is ON, the CANopen function is enabled. (Only for DVP-ES2-C)	ES2-C	╳	╳	╳	On	-	-	R/W	NO	On
M1350*	Enable PLC LINK	○	○	○	○	OFF	-	OFF	R/W	NO	OFF
M1351*	Enable auto mode on PLC LINK	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1352*	Enable manual mode on PLC LINK	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1353*	Enable access up to 50 words through PLC LINK (If M1353 is ON, D1480~D1511 are latched devices.)	○	○	○	○	OFF	-	-	R/W	YES	OFF
M1354*	Enable simultaneous data read/write in a polling of PLC LINK	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1355*	Select Slave linking mode in PLC LINK (ON: manual; OFF: auto-detection)	○	○	○	○	-	-	-	R/W	YES	OFF
M1356*	Enable station number selection function. When both M1353 and M1356 are ON, the user can specify the station number for D1900~D1931 according to their contents. No need to use the default consecutive station number specified in D1399.	○	╳	○	○	-	-	-	R/W	YES	OFF
M1357*	Enabling the detection of X0's input pulse frequency (ON: Enable; OFF: Disable); available for ES2/EX2: V3.22, ES2-C: V3.68, SA2: V3.02, SX2: V2.66 or later	ES2/ EX2	╳	SA2	○	OFF	OFF	-	R/W	NO	OFF
M1358*	Enablling the detection of X1's input pulse frequency (ON: Enable; OFF: Disable) ; available for ES2/EX2: V3.22, ES2-C: V3.68, SA2: V3.02, SX2: V2.66 or later	ES2/ EX2	╳	SA2	○	OFF	OFF	-	R/W	NO	OFF
M1359*	Enablling the detection of X2's input pulse frequency (ON: Enable; OFF: Disable) ; available for ES2/EX2: V3.22, ES2-C: V3.68, SA2: V3.02, SX2: V2.66 or later	ES2/ EX2	╳	SA2	○	OFF	OFF	-	R/W	NO	OFF
M1360*	Activation status of connection ID#1 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1361*	Activation status of connection ID#2 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1362*	Activation status of connection ID#3 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1363*	Activation status of connection ID#4 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1364*	Activation status of connection ID#5 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1365*	Activation status of connection ID#6 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1366*	Activation status of connection ID#7 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1367*	Activation status of connection ID#8 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1368*	Activation status of connection ID#9 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1369*	Activation status of connection ID#10 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1370*	Activation status of connection ID#11 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1371*	Activation status of connection ID#12 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1372*	Activation status of connection ID#13 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
M1373*	Activation status of connection ID#14 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1374*	Activation status of connection ID#15 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1375*	Activation status of connection ID#16 on PLC LINK (ON: activated)	○	○	○	○	-	-	-	R/W	YES	OFF
M1376*	Communication status of connection ID#1 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1377*	Communication status of connection ID#2 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1378*	Communication status of connection ID#3 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1379*	Communication status of connection ID#4 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1380*	Communication status of connection ID#5 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1381*	Communication status of connection ID#6 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1382*	Communication status of connection ID#7 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1383*	Communication status of connection ID#8 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1384*	Communication status of connection ID#9 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1385*	Communication status of connection ID#10 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1386*	Communication status of connection ID#11 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1387*	Communication status of connection ID#12 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1388*	Communication status of connection ID#13 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1389*	Communication status of connection ID#14 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1390*	Communication status of connection ID#15 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1391*	Communication status of connection ID#16 on PLC LINK (ON: communicating)	○	○	○	○	OFF	-	-	R	NO	OFF
M1392*	Connecting error on PLC LINK connection ID#1	○	○	○	○	OFF	-	-	R	NO	OFF
M1393*	Connecting error on PLC LINK connection ID#2	○	○	○	○	OFF	-	-	R	NO	OFF
M1394*	Connecting error on PLC LINK connection ID#3	○	○	○	○	OFF	-	-	R	NO	OFF
M1395*	Connecting error on PLC LINK connection ID#4	○	○	○	○	OFF	-	-	R	NO	OFF
M1396*	Connecting error on PLC LINK connection ID#5	○	○	○	○	OFF	-	-	R	NO	OFF
M1397*	Connecting error on PLC LINK connection ID#6	○	○	○	○	OFF	-	-	R	NO	OFF
M1398*	Connecting error on PLC LINK connection ID#7	○	○	○	○	OFF	-	-	R	NO	OFF
M1399*	Connecting error on PLC LINK connection ID#8	○	○	○	○	OFF	-	-	R	NO	OFF
M1400*	Connecting error on PLC LINK connection ID#9	○	○	○	○	OFF	-	-	R	NO	OFF
M1401*	Connecting error on PLC LINK connection ID#10	○	○	○	○	OFF	-	-	R	NO	OFF
M1402*	Connecting error on PLC LINK connection ID#11	○	○	○	○	OFF	-	-	R	NO	OFF
M1403*	Connecting error on PLC LINK connection ID#12	○	○	○	○	OFF	-	-	R	NO	OFF
M1404*	Connecting error on PLC LINK connection ID#13	○	○	○	○	OFF	-	-	R	NO	OFF
M1405*	Connecting error on PLC LINK connection ID#14	○	○	○	○	OFF	-	-	R	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
M1406*	Connecting error on PLC LINK connection ID#15	○	○	○	○	OFF	-	-	R	NO	OFF
M1407*	Connecting error on PLC LINK connection ID#16	○	○	○	○	OFF	-	-	R	NO	OFF
M1408*	Reading complete on PLC LINK connection ID#1	○	○	○	○	OFF	-	-	R	NO	OFF
M1409*	Reading complete on PLC LINK connection ID#2	○	○	○	○	OFF	-	-	R	NO	OFF
M1410*	Reading complete on PLC LINK connection ID#3	○	○	○	○	OFF	-	-	R	NO	OFF
M1411*	Reading complete on PLC LINK connection ID#4	○	○	○	○	OFF	-	-	R	NO	OFF
M1412*	Reading complete on PLC LINK connection ID#5	○	○	○	○	OFF	-	-	R	NO	OFF
M1413*	Reading complete on PLC LINK connection ID#6	○	○	○	○	OFF	-	-	R	NO	OFF
M1414*	Reading complete on PLC LINK connection ID#7	○	○	○	○	OFF	-	-	R	NO	OFF
M1415*	Reading complete on PLC LINK connection ID#8	○	○	○	○	OFF	-	-	R	NO	OFF
M1416*	Reading complete on PLC LINK connection ID#9	○	○	○	○	OFF	-	-	R	NO	OFF
M1417*	Reading complete on PLC LINK connection ID#10	○	○	○	○	OFF	-	-	R	NO	OFF
M1418*	Reading complete on PLC LINK connection ID#11	○	○	○	○	OFF	-	-	R	NO	OFF
M1419*	Reading complete on PLC LINK connection ID#12	○	○	○	○	OFF	-	-	R	NO	OFF
M1420*	Reading complete on PLC LINK connection ID#13	○	○	○	○	OFF	-	-	R	NO	OFF
M1421*	Reading complete on PLC LINK connection ID#14	○	○	○	○	OFF	-	-	R	NO	OFF
M1422*	Reading complete on PLC LINK connection ID#15	○	○	○	○	OFF	-	-	R	NO	OFF
M1423*	Reading complete on PLC LINK connection ID#16	○	○	○	○	OFF	-	-	R	NO	OFF
M1424*	Writing complete on PLC LINK connection ID#1	○	○	○	○	OFF	-	-	R	NO	OFF
M1425*	Writing complete on PLC LINK connection ID#2	○	○	○	○	OFF	-	-	R	NO	OFF
M1426*	Writing complete on PLC LINK connection ID#3	○	○	○	○	OFF	-	-	R	NO	OFF
M1427*	Writing complete on PLC LINK connection ID#4	○	○	○	○	OFF	-	-	R	NO	OFF
M1428*	Writing complete on PLC LINK connection ID#5	○	○	○	○	OFF	-	-	R	NO	OFF
M1429*	Writing complete on PLC LINK connection ID#6	○	○	○	○	OFF	-	-	R	NO	OFF
M1430*	Writing complete on PLC LINK connection ID#7	○	○	○	○	OFF	-	-	R	NO	OFF
M1431*	Writing complete on PLC LINK connection ID#8	○	○	○	○	OFF	-	-	R	NO	OFF
M1432*	Writing complete on PLC LINK connection ID#9	○	○	○	○	OFF	-	-	R	NO	OFF
M1433*	Writing complete on PLC LINK connection ID#10	○	○	○	○	OFF	-	-	R	NO	OFF
M1434*	Writing complete on PLC LINK connection ID#11	○	○	○	○	OFF	-	-	R	NO	OFF
M1435*	Writing complete on PLC LINK connection ID#12	○	○	○	○	OFF	-	-	R	NO	OFF
M1436*	Writing complete on PLC LINK connection ID#13	○	○	○	○	OFF	-	-	R	NO	OFF
M1437*	Writing complete on PLC LINK connection ID#14	○	○	○	○	OFF	-	-	R	NO	OFF
M1438*	Writing complete on PLC LINK connection ID#15	○	○	○	○	OFF	-	-	R	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
M1439*	Writing complete on PLC LINK connection ID#16	○	○	○	○	OFF	-	-	R	NO	OFF
M1520*	Close the conditional contact and to enable the instructions PLSR, DPLSR Y0, DDRVI, DDRVA CH2 (Y4/Y5) to execute ramp-down (ON: Enable; OFF: Disable) (available for EC5: V1.00 or later)	EC5	×	×	×	OFF	-	-	R/W	NO	OFF
M1521*	Close the conditional contact and to enable the instructions PLSR, DPLSR Y0, DDRVI, DDRVA CH3 (Y6/Y7) to execute ramp-down (ON: Enable; OFF: Disable) (available for EC5: V1.00 or later)	EC5	×	×	×	OFF	-	-	R/W	NO	OFF
M1524	Auto-reset Y2 when high speed pulse output is completed	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1525	Auto-reset Y3 when high speed pulse output is completed	ES2/ EX2	○	○	○	OFF	-	-	R/W	NO	OFF
M1526	Auto-reset Y4 when high speed pulse output is completed	EC5	×	×	×	OFF	-	-	R/W	NO	OFF
M1527	Auto-reset Y6 when high speed pulse output is completed	EC5	×	×	×	OFF	-	-	R/W	NO	OFF
M1530	Switching clock pulse of Y4 for PWM instruction (ON: 10 us; OFF: 100 us)	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1531	Switching clock pulse of Y6 for PWM instruction (ON: 10 us; OFF: 100 us)	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1532	Reverse Y5 pulse output direction in high speed pulse output instructions	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1533	Reverse Y7 pulse output direction in high speed pulse output instructions	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1534	Enable ramp-down time setting on Y0. Has to be used with D1348.	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1535	Enable ramp-down time setting on Y2. Has to be used with D1349.	○	○	○	○	OFF	-	-	R/W	NO	OFF
M1536	Enable ramp-down time setting on Y4. Has to be used with D1350.	EC5	×	×	×	OFF	-	-	R/W	NO	OFF
M1537	Enable ramp-down time setting on Y6. Has to be used with D1351.	EC5	×	×	×	OFF	-	-	R/W	NO	OFF
M1538	Indicate pause status of Y0	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1539	Indicate pause status of Y1	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1540	Indicate pause status of Y2	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1541	Indicate pause status of Y3	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1542	Indicate pause status of Y4	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1543	Indicate pause status of Y6	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1550	Modbus TCP Port conflicts; should be used with D1110	ES2-E v1.46	×	SE v1.46	×	OFF	OFF	-	R	NO	OFF
M1551	EtherNet/IP upper/lower byte arrangement , OFF: Little-endian; ON: Big-endian	ES2-E	×	26SE	×	OFF	OFF	-	R/W	NO	OFF
M1580	The absolute position of Delta ASDA-A2 servo is read successfully by means of the instruction DABSR.	ES2/ EX2: V3.2	×	V2.6 V1.4	V2.4	OFF	OFF	OFF	R/W	NO	OFF
M1581	The absolute position of Delta ASDA-A2 servo is not read successfully by means of the instruction DABSR.	ES2/ EX2: V3.2	×	V2.6 V1.4	V2.4	OFF	OFF	OFF	R/W	NO	OFF
M1582	Check the input/output points of the digital extension module or number of the special modules when power-on. (ON: enable; OFF: disable; should be used with D1248, D1254, and D1385) available for ES2/EX2/ES2C: V3.62, ES2E: V1.48 or later	ES2/ EX2	×	×	×	-	-	-	R/W	YES	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
M1584	If the left limit switch of CH0 is enabled, it can be triggered either by a rising-edge signal or by a falling-edge signal. (OFF: Rising-edge signal; ON: Falling-edge signal); available for ES2/EX2: V3.2, EC5: V1.00 or later	○	V3.0	V2.8 V1.4	V2.6	OFF	OFF	-	R/W	NO	OFF
M1585	If the left limit switch of CH1 is enabled, it can be triggered either by a rising-edge signal or by a falling-edge signal. (OFF: Rising-edge signal; ON: Falling-edge signal); available for ES2/EX2: V3.2, EC5: V1.00 or later	○	V3.0	V2.8 V1.4	V2.6	OFF	OFF	-	R/W	NO	OFF
M1586	If the left limit switch of CH2 is enabled, it can be triggered either by a rising-edge signal or by a falling-edge signal. (OFF: Rising-edge signal; ON: Falling-edge signal);	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1587	If the left limit switch of CH3 is enabled, it can be triggered either by a rising-edge signal or by a falling-edge signal. (OFF: Rising-edge signal; ON: Falling-edge signal);	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1588	When M1588=ON, set the zero point for the first DZRN auto-reset (Y4, Y6) to the position wher DOG point stops on the positive direction. (If M1588 is OFF, it stops on the negative direction.)	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1589	When M1589=ON, set the zero point for the first DZRN auto-reset (Y4, Y6) to the position wher DOG point stops on the positive direction. (If M1589 is OFF, it stops on the negative direction.)	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1590	Enabling the acceleration of the Ethernet data exchange (ON: Enable; OFF: Disable)	×	×	V2.66 V1.4	V2.66	OFF	OFF	-	R/W	NO	OFF
M1598*	Enabling the fetching of the value in the hardware counter C243/C245/C246/C247/C248/C251/C252, and using X6 as a fetching signal (ON: Enable; OFF: Disable)	ES2/ EX2: V3.28	V 3.28	SA2: V2.82	V2.82	OFF	-	-	R/W	NO	OFF
M1599*	Enabling the fetching of the value in the hardware counter C244/C249/C250/C253/C254, and using X7 as a fetching signal (ON: Enable; OFF: Disable)	ES2/ EX2: V3.28	V 3.28	SA2: V2.82	V2.82	OFF	-	-	R/W	NO	OFF
M1604	Enable pulse output at the maximum frequency of the ramp up/down set in D1410 and D1411 for the fixed slope of Y0 or CH0 (Y0/Y1). (ON: Enable; OFF: Disable) Available for ES2/EX2: V3.28, ES2-E: V1.00, EC5: V1.00, 12SA2/SX2: V2.82, SS2: V3.24, SE: V2.00, 28SA2: V2.90.	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1605	Enable pulse output at the maximum frequency of the ramp up/down set in D1412 and D1413 for the fixed slope of Y2 or CH1 (Y2/Y3). (ON: Enable; OFF: Disable) Available for ES2/EX2: V3.28, ES2-E: V1.00, EC5: V1.00, 12SA2/SX2: V2.82, SS2: V3.24, SE: V2.00, 28SA2: V2.90.	○	○	○	○	OFF	OFF	-	R/W	NO	OFF
M1606	Enable pulse output at the maximum frequency of the ramp up/down set in D1988 and D1989 for the fixed slope of Y1. (ON: Enable; OFF: Disable) Available for ES2/EX2: V3.28, ES2-E: V1.00, 12SA2/SX2: V2.82, SS2: V3.24, SE: V2.00, 28SA2: V2.90.	ES2/ EX2	×	○	○	OFF	OFF	-	R/W	NO	OFF
M1607	Enable pulse output at the maximum frequency of the ramp up/down set in	ES2/ EX2	×	○	○	OFF	OFF	-	R/W	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
	D1990 and D1991 for the fixed slope of Y3. (ON: Enable; OFF: Disable) Available for ES2/EX2: V3.28, ES2-E: V1.00, 12SA2/SX2: V2.82, SS2: V3.24, SE: V2.00, 28SA2: V2.90.										
M1610	Enable pulse output at the maximum frequency of the ramp up/down set in D1970 and D1971 for the fixed slope of CH2 (Y4/Y5). (ON: Enable; OFF: Disable)	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1611	Enable pulse output at the maximum frequency of the ramp up/down set in D1972 and D1973 for the fixed slope of CH3 (Y6/Y7) (ON: Enable; OFF: Disable)	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1614	Enabling the drive function	ES2-C V3.48	×	×	×	OFF	-	-	R/W	NO	OFF
M1615	Drive initializaiton complete	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1616	Drive error	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1617	Independent heartbeat mode (ON: independent mode, OFF: linking mode	ES2-C V3.49	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1620	Communication protocol for CANRS instruction; OFF: 2.0B, ON: 2.0A); available for ES2-C: V3.49, SA2/SX2: V3.02, 12SE: V1.87, 26SE: V2.02 or later	ES2-C	×	○	○	OFF	-	-	R/W	NO	OFF
M1621	Communication mode (Master/Slave) for CANRS instruction; OFF: master, ON: slave) ; available for ES2-C: V3.49, SA2/SX2: V3.02, 12SE: V1.87, 26SE: V2.02 or later	ES2-C	×	○	○	OFF	-	-	R/W	NO	OFF
M1622	Single or two ways communication for CANRS instruction; OFF: two ways, ON: single way; available for ES2-C: V3.49, SA2/SX2: V3.02, 12SE: V1.87, 26SE: V2.02 or later	ES2-C	×	○	○	OFF	-	-	R/W	NO	OFF
M1623	Communication error for CANRS instruction; available for ES2-C: V3.49, SA2/SX2: V3.02, 12SE: V1.87, 26SE: V2.02 or later	ES2-C	×	○	○	OFF	-	-	R/W	NO	OFF
M1624	For drive instructions, drive #1 pulse outputting complete	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1625	For drive instructions, drive #2 pulse outputting complete	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1626	For drive instructions, drive #3 pulse outputting complete	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1627	For drive instructions, drive #4 pulse outputting complete	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1628	For drive instructions, drive #5 pulse outputting complete	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1629	For drive instructions, drive #6 pulse outputting complete	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1630	For drive instructions, drive #7 pulse outputting complete	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1631	For drive instructions, drive #8 pulse outputting complete	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1632	For drive instructions, drive #1 ramp-down stop	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1633	For drive instructions, drive #2 ramp-down stop	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1634	For drive instructions, drive #3 ramp-down stop	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1635	For drive instructions, drive #4 ramp-down stop	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1636	For drive instructions, drive #5 ramp-down stop	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
M1637	For drive instructions, drive #6 ramp-down stop	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1638	For drive instructions, drive #7 ramp-down stop	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1639	For drive instructions, drive #8 ramp-down stop	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1640	For drive instructions, drive #1 enabling drive	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1641	For drive instructions, drive #2 enabling drive	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1642	For drive instructions, drive #3 enabling drive	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1643	For drive instructions, drive #4 enabling drive	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1644	For drive instructions, drive #5 enabling drive	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1645	For drive instructions, drive #6 enabling drive	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1646	For drive instructions, drive #7 enabling drive	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1647	For drive instructions, drive #8 enabling drive	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1648	For drive instructions, drive #1 enabling go back and forth function	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1649	For drive instructions, drive #2 enabling go back and forth function	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1650	For drive instructions, drive #3 enabling go back and forth function	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1651	For drive instructions, drive #4 enabling go back and forth function	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1652	For drive instructions, drive #5 enabling go back and forth function	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1653	For drive instructions, drive #6 enabling go back and forth function	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1654	For drive instructions, drive #7 enabling go back and forth function	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1655	For drive instructions, drive #8 enabling go back and forth function	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1656	For drive instructions, drive #1 directional indication	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1657	For drive instructions, drive #2 directional indication	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1658	For drive instructions, drive #3 directional indication	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1659	For drive instructions, drive #4 directional indication	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1660	For drive instructions, drive #5 directional indication	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1661	For drive instructions, drive #6 directional indication	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1662	For drive instructions, drive #7 directional indication	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1663	For drive instructions, drive #8 directional indication	ES2-C V3.48	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1664	Drive #1 heartbeat error	ES2-C V3.49	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1665	Drive #2 heartbeat error	ES2-C V3.49	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1666	Drive #3 heartbeat error	ES2-C V3.49	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1667	Drive #4 heartbeat error	ES2-C V3.49	×	×	×	OFF	-	OFF	R/W	NO	OFF

Special M	Function	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
M1668	Drive #5 heartbeat error	ES2-C V3.49	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1669	Drive #6 heartbeat error	ES2-C V3.49	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1670	Drive #7 heartbeat error	ES2-C V3.49	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1671	Drive #8 heartbeat error	ES2-C V3.49	×	×	×	OFF	-	OFF	R/W	NO	OFF
M1672	Use REF instruction to refresh current position of high-speed output Y0; available for ES2/EX2/ES2-C: V3.60, ES2-E: V1.40, EC5: V1.00, 28SA2/12SA2/SX2: V3.0, 26SE: V1.92; EC5: V1.00 and later	○	×	○	○	OFF	OFF	-	R/W	NO	OFF
M1673	Use REF instruction to refresh current position of high-speed output Y1; available for ES2/EX2/ES2-C: V3.60, ES2-E: V1.40, 28SA2/12SA2/SX2: V3.0, 26SE: V1.92 and later	ES2/ EX2	×	○	○	OFF	OFF	-	R/W	NO	OFF
M1674	Use REF instruction to refresh current position of high-speed output Y2; available for ES2/EX2/ES2-C: V3.60, ES2-E: V1.40, 28SA2/12SA2/SX2: V3.0, 26SE: V1.92	○	×	○	○	OFF	OFF	-	R/W	NO	OFF
M1675	Use REF instruction to refresh current position of high-speed output Y3; available for ES2/EX2/ES2-C: V3.60, ES2-E: V1.40, 28SA2/12SA2/SX2: V3.0, 26SE: V1.92	ES2/ EX2	×	○	○	OFF	OFF	-	R/W	NO	OFF
M1676	Use REF instruction to refresh current position of high-speed output Y4;	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1677	Use REF instruction to refresh current position of high-speed output Y6;	EC5	×	×	×	OFF	OFF	-	R/W	NO	OFF
M1700* ~1731*	Enabling to read the code 0X04 of the connection ID1 from PLC Link (available for ES2/EX2/ES2-C: V3.48, ES2-E : V1.00, 12SA2: V3.0, SS2: V3.60, 2: V3.0, 26SE: V.0, 28SA2: V3.0 and later)	ES2/ EX2	○	○	○	OFF	OFF	-	R/W	NO	OFF

2.9 S Relay

Initial step relay Starting instruction in Sequential Function Chart (SFC).
S0~S9, total 10 points.

Zero return step relay Returns to zero point when using IST instruction in program. Zero return step relays not used for IST instruction can be used as general step relays.
S10~S19, total 10 ponits.

Latched step relay In sequential function chart (SFC), latched step relay will be saved when power loss after running. The state of power on after power loss will be the same as the sate before power loss.
S20 ~ S127, total 108 points.

General purpose step relay General relays in sequential function chart (SFC). They will be cleared when power loss after running.
S128 ~ S911, total 784 points.

Alarm step relay Used with alarm driving instruction API 46 ANS as an alarm contact for recording the alarm messages or eliminating external malfunctions.
S912 ~ S1023, total 112 points.

2.10 T (Timer)

The units of the timer are 1ms, 10ms and 100ms and the counting method is counting up. When the present value in the timer equals the set value, the associated output coil will be ON. The set value should be a K value in decimal and can be specified by the content of data register D.

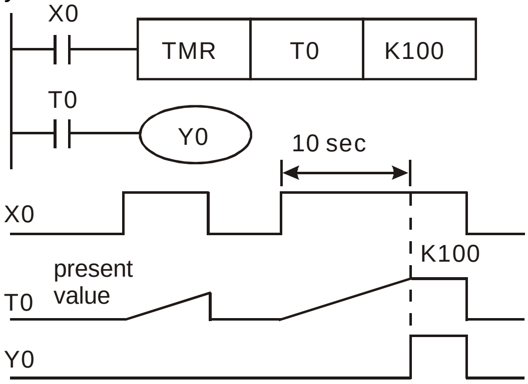
The actual set time in the timer = timer resolution× set value

Ex: If set value is K200 and timer resolution is 10ms, the actual set time in timer will be 10ms*200 = 2000ms = 2 sec.

General Timer

The timer executes once when the program reaches END instruction. When TMR instruction is executed, the timer coil will be ON when the current value reaches its preset value.

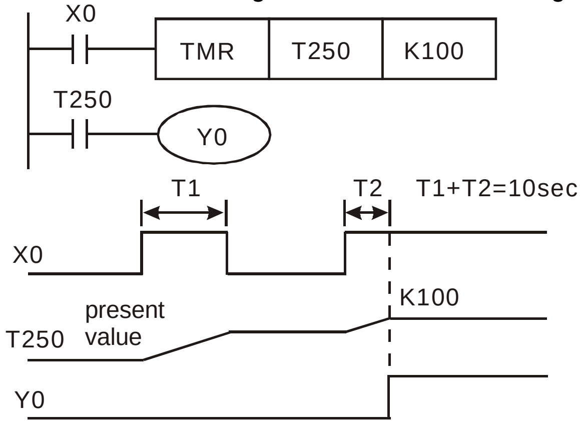
When X0 = ON, TMR instruction is driven. When current value achieves K100, the assocailte timer contact T0 is ON to drive Y0. If X0 = OFFor the power is off, the current value in T0 will be cleared as 0 and output Y0 driven by contact T0 will be OFF.



Accumulative Timer

The timer executes once when the program reaches END instruction. When TMR instruction is executed, the timer coil will be ON when the current value reaches its preset value. For accumulative timers, current value will not be cleared when timing is interrupted.

Timer T250 will be driven when X0 = ON. When X0 = OFFor the power is off, timer T250 will pause and retain the current value. When X0 is ON again, T250 resumes timing from where it was paused.



Timers for Subroutines and Interrupts

Timers for subroutines and interrupts count once when END instruction is met. The associated output coils will be ON if the set value is achieved when End instruction executes. T184~T199 are the only timers that can be used in subroutines or interrupts. Generals timers used in subroutines and interrupts will not work if the subroutines or interrupts are not executing.

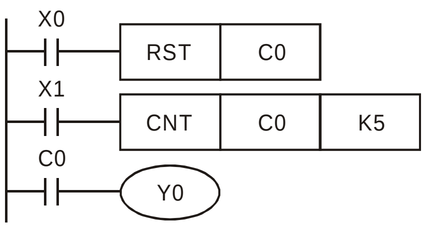
2.11 C (Counter)

Counters will increment their present count value when input signals are triggered from OFF→ON.

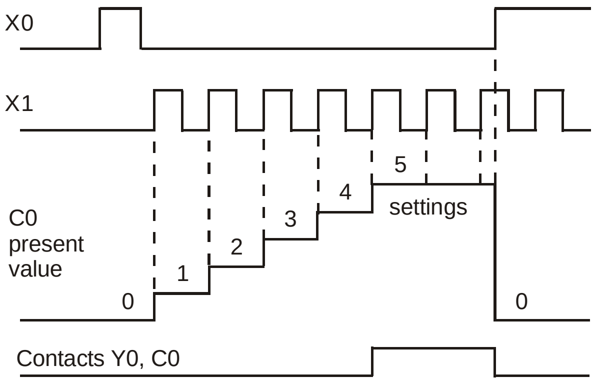
<table>
<tr><th></th><th>16 bits counters</th><th colspan="3">32 bits counters</th></tr>
<tr><td>Type</td><td>General</td><td>General</td><td colspan="2">High speed</td></tr>
<tr><td>Counters</td><td>C0~C199</td><td>C200~C231(C232)</td><td>C232(C233)~C242, C245~C254</td><td>C243, C244</td></tr>
<tr><td>Count direction</td><td>Count up</td><td colspan="2">Count up/down</td><td>Count up</td></tr>
<tr><td>Range</td><td>0~32,767</td><td colspan="2">-2,147,483,648~+2,147,483,647</td><td>0~2,147,483,647</td></tr>
<tr><td>Preset value register</td><td>Constant K or data register D (Word)</td><td colspan="3">Constant K or data register D (Dword)</td></tr>
<tr><td>Output operation</td><td>Counter will stop when preset value reached</td><td colspan="2">Counter will keep on counting when preset value reached. The count value will become -2,147,483,648 if one more count is added to +2,147,483,647</td><td>Counter will keep on counting when preset value is reached. The count value will become 0 if one more count is added to +2,147,483,647</td></tr>
<tr><td>Output contact function</td><td>Ouptut Coil will be ON when counter reaches preset value.</td><td colspan="2">Output coil is ON when counter reaches or is above preset value.
Output coil is OFF when counter is below preset value.</td><td>Output coil is ON when counter reaches or is above preset value</td></tr>
<tr><td>High speed conparison</td><td>-</td><td>Associated devices are activated immediately when preset value is reached, i.e. independant of scan time.</td><td colspan="2">-</td></tr>
<tr><td>Reset action</td><td colspan="4">The present value will reset to 0 when RST instruction is executed, output coil will be OFF.</td></tr>
</table>

Example:

LD	X0
RST	C0
LD	X1
CNT	C0 K5
LD	C0
OUT	Y0



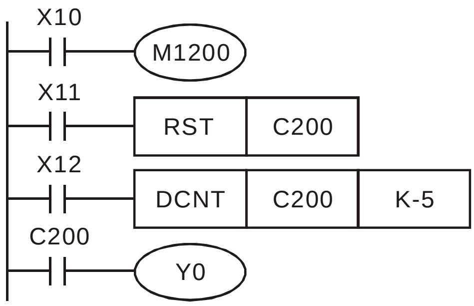
When X0 = ON, RST instruction resets C0. Every time When X1 is driven, C0 will count up (add 1).
When C0 reaches the preset value K5, output coil Y0 will be ON and C0 will stop counting and ignore the signals from input X1.



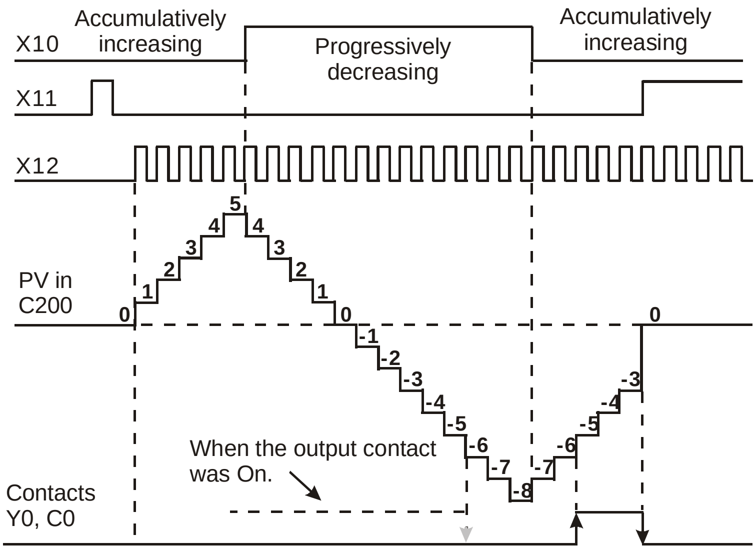
M relays M1200~M1254 are used to set the up/down counting direction for C200~C254 respectively. Setting the corresponding M relay ON will set the counter to count down.

Example:

```
LD    X10
OUT   M1200
LD    X11
RST   C200
LD    X12
CNT   C200  K-5
LD    C200
OUT   Y0
```



a) X10 drives M1200 to determine counting direction (up / down) of C200
b) When X11 goes from OFF to ON, RST instsruction will be executed and the PV (present value) in C200 will be cleared and contact C200 is OFF.
c) When X12 goes from Off to On, PV of C200 will count up (plus 1) or count down (minus 1).
d) When PV in C200 changes from K-6 to K-5, the contact C200 will be energized. When PV in C200 changes from K-5 to K-6, the contact of C200 will be reset.
e) If MOV instruction is applied through WPLSoft or HPP to designate a value bigger than SV to the PV register of C0, next time when X1 goes from OFF to ON, the contact C0 will be ON and PV of C0 will equal SV.



2.12 High-speed Counters

There are two types of high speed counters provided including Software High Speed Counter (SHSC) and Hardware High Speed Counter (HHSC). The same Input point (X) can be designated with only one high speed counter. Double designation on the same input or the same counter will result in syntax error when executing DCNT instruction.

Applicable Software High Speed Counters:

C / X	1-phase input								2 phase 2 input		
	C235	C236	C237	C238	C239	C240	C241	C242	C232[#2]	C233	C234
X0	U/D								A		
X1		U/D									
X2			U/D						B		
X3				U/D							
X4					U/D					A	
X5						U/D				B	
X6							U/D				A
X7								U/D			B
R/F	M1270	M1271	M1272	M1273	M1274	M1275	M1276	M1277	-	-	-
U/D	M1235	M1236	M1237	M1238	M1239	M1240	M1241	M1242	-	-	-

U: Count up D: Count down A: Phase A input B: Phase B input

Note:

1. SHSC supports max 10kHz input pulse on single point. Max 8 counters are applicable in the same time.
2. An SS2, SA2, SX2, SE model does not support a two-phase two-input counter (C232 with the input points X0 and X2).
3. For 2-phase 2-input conuting, (X4, X5) (C233) and (X6, X7) (C234), max 5kHz. (X0,X2) (C232), max 15kHz.
4. 2-phase 2-input counting supports double and quadruple frequency, which is selected in D1022 as the table shown below.
5. R/F (Rising edge trigger/ Falling edge trigger) can also be specified by special M. OFF = Rising; ON = Falling.
6. U/D (Count up/Count down) can be specified by special M. OFF = count up; ON = count down.

Applicable Hardware High Speed Counters:

C / **X**	1-phase input		1-phase 2-input						2-phase 2-input			
	C243	C244	C245	C246	C247	C248	C249[#2]	C250[#2]	C251	C252	C253	C254
X0	U		U/D	U/D	U	U			A	A		
X1	R		Dir	Dir	D	D			B	B		
X2		U					U/D	U/D			A	A
X3		R					Dir	Dir			B	B
X4				R		R				R		
X5								R				R

U: Count up A: Phase A input Dir: Directoin signal input
D: Count down B: Phase B input R: Reset signal input

Note:

1. Refer to the following table for the maximum frequency of the 1-phase input counters X0 (C243) and X2 (C244).

Model	ES2 Series	EC5	12SA2	SX2	SS2	12SE	26SE	28SA2
Maximum frequency of the counter (Hz)	100K	50K	100K	100K	20K	100K	100K	100K

2. Refer to the following table for the maximum frequency of the 1-phase 2-input counters (X0, X1) (C245, C246) and (X2, X3) (C249, C250).

Model	ES2 Series	EC5	12SA2	SX2	SS2	12SE*	26SE*	28SA2
Maximum frequency of the counter (Hz)	100K	50K	100K	100K	20K	100K	100K	100K

* 12SE with firmware V2.xx and 26SE with firmware V1.xx support C249 and C250.

3. Refer to the following table for the maximum frequency of the 1-phase 2-input counters (X0, X1) (C247, C248).

Model	ES2 Series	32ES211T	EC5	12SA2	SX2	SS2	12SE	26SE	28SA2
Maximum frequency of the counter (Hz)	10K / 100K*	100K	50K	100K	100K	20K	100K	100K	100K

* The maximum frequency of the 1-phase 2-input counters (X0, X1) (C247, C248) is 100 kHz for ES2 series manufactured after year 2013.

4. Refer to the following table for the maximum frequency of the 2-phase 2-input counter (X0, X1) (C251, C252).

Model	ES2 Series	32ES211T	EC5	12SA2	SX2	SS2	12SE	26SE	28SA2
Maximum frequency of the counter (Hz)	5K / 50K*	50K	25K	50K	10K	10K	50K	50K	50K

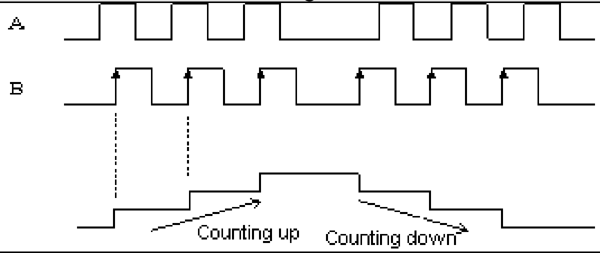
*The maximum frequency of the 2-phase 2-input counter (X0, X1) (C251, C252) is 50 kHz for ES2 series manufactured after year 2013.

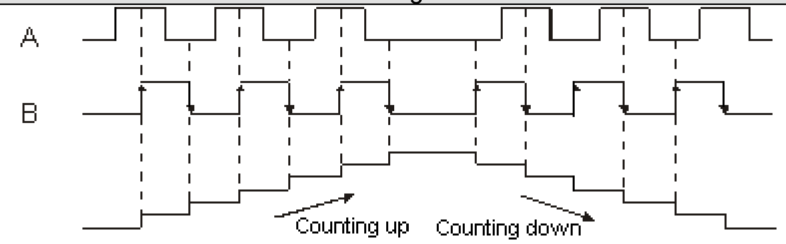
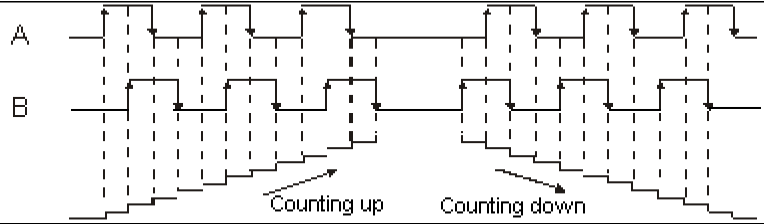
5. Refer to the following table for the maximum frequency of the 2-phase 2-input counter (X2, X3) (C253, C254).

Model	ES2 Series	32ES211T	EC5	12SA2	SX2	SS2	12SE	26SE	28SA2
Maximum frequency of the counter (Hz)	5K / 50K*	50K	25K	50K	10K	10K	50K	50K	50K

* The maximum frequency of the 2-phase 2-input counter (X2, X3) (C253, C254) is 50 kHz for ES2 series manufactured after year 2013.

6. 2-phase 2-input counting supports double and 4 times frequency, which is selected in D1022 as the table in next page. Please refer to the below table for detailed counting wave form.

D1022	Counting mode
K1	

D1022	Counting mode
K2 (Double Frequency)	
K4 or other value (Quadruple frequency) (Default)	

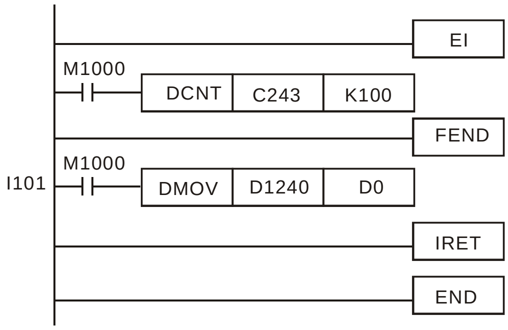
7. DVP-ES2/DVP-SS2 series PLCs whose firmware version is 2.80 or above support the single frequency mode. DVP-SA2/DVP-SX2 series PLCs whose firmware version is 2.00 support the single frequency mode. The other PLCs support the three modes.
8. C243 and C244 support count-up mode only and occupy the associate input points X1 and X3 as reset ("R") function. If users do not need to apply reset function, set ON the associated special M relays (M1243 and M1244) to disable the reset function.
9. "Dir" refers to direction control. OFF indicates counting up; ON indicates counting down.
10. When X1, X3, X4 and X5 is applied for reset function and associated external interrupts are disabled, users can define the reset function as Rising/Falling-edge triggered by special M relays

Reset Function	X1	X3	X4	X5
R/F	M1271	M1273	M1274	M1275

11. When X1, X3, X4 and X5 is applied for reset function and external interrupts are applied, the interrupt instructions have the priority in using the input points. In addition, PLC will move the current data in the counters to the associated data registers below then reset the counters.

Special D	D1241, D1240				D1243, D1242		
Counter	C243	C246	C248	C252	C244	C250	C254
External Interrupt	X1 (I100/I101)	X4(I400/I401)			X3 (I300/I301)	X5(I500/I501)	

Example:

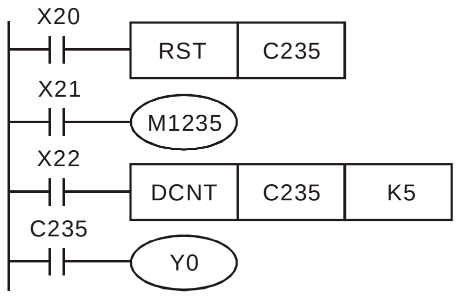


When C243 is counting and external interrupt is triggerred from X1(I101), counted value in C243 will be move to (D1241, D1240) immediately then C243 is reset. After this interrupt I101 executes.

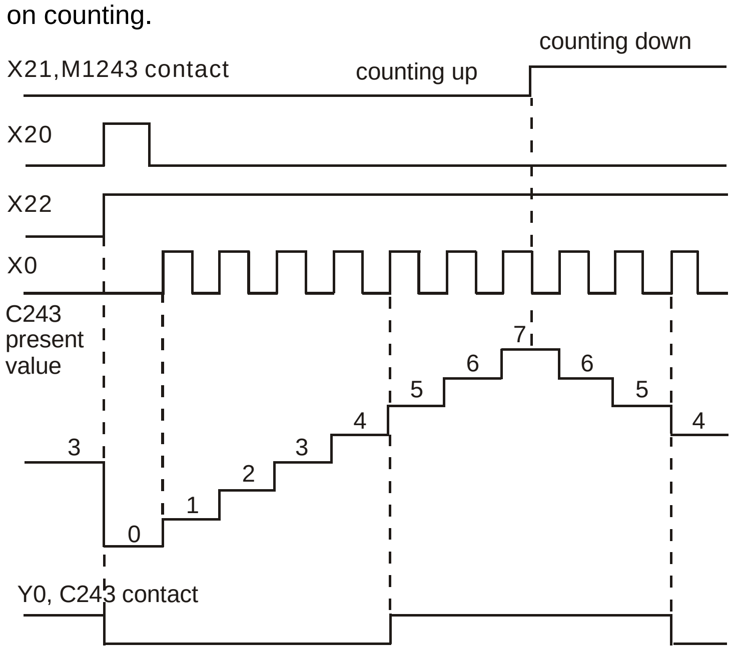
1-phase 1 input high-speed counter:

Example:

```
LD      X20
RST     C235
LD      X21
OUT     M1235
LD      X22
DCNT    C235  K5
LD      C235
OUT     Y0
```



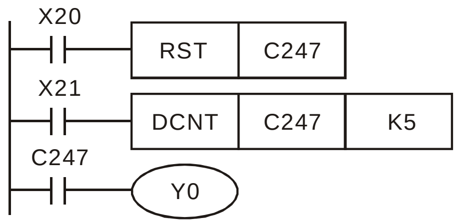
1. X21 drives M1235 to determine counting direction (Up/Down) of C235.
2. When X20 = ON, RST instsruction executes and the current value in C235 will be cleared. Contact C235 will be OFF
3. When X22 = ON, C235 receives signals from X0 and counter will count up (+1) or count down (-1).
4. When counter C235 reaches K5, contact C235 will be ON. If there is still input signal input for X0, it will keep on counting.



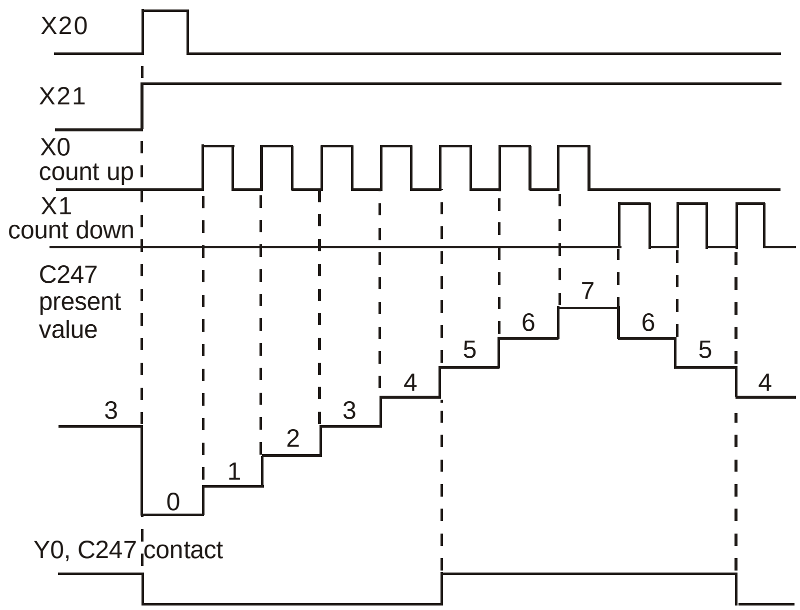
1-phase 2 inputs high-speed counter:

Example:

```
LD      X20
RST     C247
LD      X21
DCNT    C247 K5
LD      C247
OUT     Y0
```



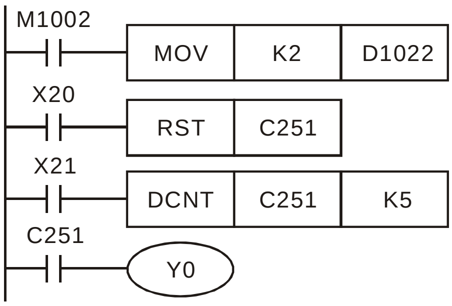
1. When X20 is ON, RST instsruction executes and the current value in C247 will be cleared. Contact C247 will be OFF.
2. When X21=ON, C247 receives count signals from X0 and counter counts up (+1), or C247 receives count signal from X1 and counter counts down (-1)
3. When C247 reaches K5, contact C247 will be ON. If there is still input signal from X0 or X1, C247 will keep on counting



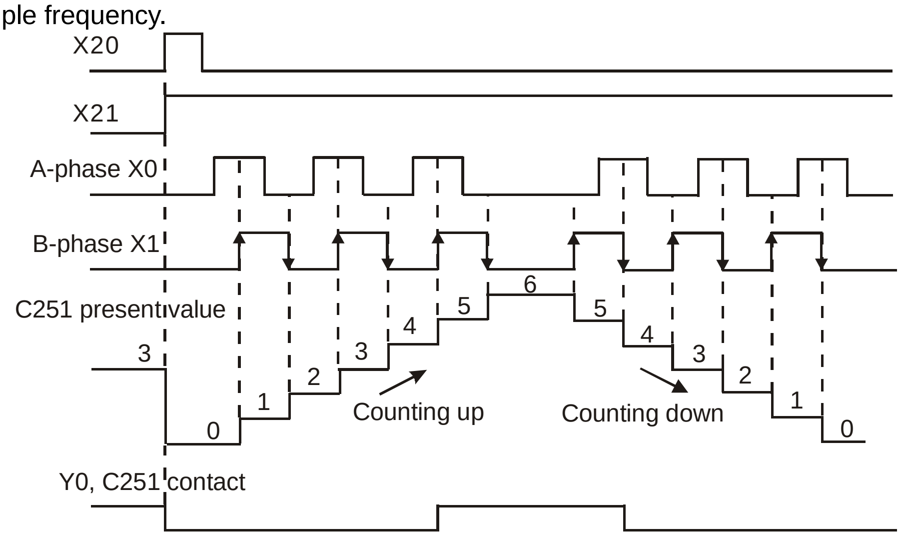
AB-phase input high-speed counter:

Example:

```
LD      M1002
MOV     K2 D1022
LD      X20
RST     C251
LD      X21
DCNT    C251 K5
LD      C251
OUT     Y0
```



1. When X20 is ON, RST instsruction executes and the current value in C251 will be cleared. Contact C251 will be OFF.
2. When X21 is ON, C251 receives A phase counting signal of X0 input terminal and B phase counting signal of X1 input terminal and executes count up or count down
3. When counter C251 reaches K5, contact C251 will be ON. If there is still input signal from X0 or X1, C251 will keep on counting
4. Counting mode can be specified as double frequency or 4-times frequency by D1022. Default: quadruple frequency.



2.13 Special Data Register

The types and functions of special registers (special D) are listed in the table below. Care should be taken that some registers of the same No. may bear different meanings in different series MPUs. Special M and special D marked with “*” will be further illustrated in 2.13. Columns marked with “R” refers to “read only”, “R/W” refers to “read and write”, “-“ refers to the status remains unchanged and “#” refers to that system will set it up according to the status of the PLC. For detailed explanation please also refer to 2.13 in this chapter.

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1000*	Setting value of the watchdog timer (WDT) (Unit: 1ms)	○	○	○	○	200	-	-	R/W	NO	200
D1001	Displaying the firmware version of DVP-PLC (For example, the firmware version is 1.0 if the value in D1001 is HXX10.)	○	○	○	○	-	-	-	R	NO	#
D1002*	Program capacity (ES2/EX2/EC5/SA2/SX2: 15872; SS2: 7920)	○	○	○	○	#	-	-	R	NO	#
D1003	Sum of the PLC internal program memory (ES2/EX2/EC5/SA2/SX2: -15872; SS2: -7920)	○	○	○	○	-	-	-	R	YES	#
D1004*	Syntax check error code	○	○	○	○	0	0	-	R	NO	0
D1008*	Step address when WDT is ON	○	○	○	○	0	-	-	R	NO	0
D1009	Number of LV (Low voltage) signal occurrence	○	○	○	○	-	-	-	R	YES	0
D1010*	Current scan time (Unit: 0.1ms)	○	○	○	○	#	#	#	R	NO	0
D1011*	Minimum scan time (Unit: 0.1ms)	○	○	○	○	#	#	#	R	NO	0
D1012*	Maximum scan time (Unit: 0.1ms)	○	○	○	○	#	#	#	R	NO	0
D1015*	Value of accumulative high-speed timer (0~32,767, unit: 0.1ms)	○	○	○	○	0	-	-	R/W	NO	0
D1018*	π PI (Low byte)	○	○	○	○	H’ 0FDB	H’ 0FDB	H’ 0FDB	R/W	NO	H’ 0FDB
D1019*	π PI(High byte)	○	○	○	○	H’ 4049	H’ 4049	H’ 4049	R/W	NO	H’ 4049
D1020*	X0~X7 input filter (unit: ms) 0~20ms adjustable	○	○	○	○	10	-	-	R/W	NO	10
D1021*	X10~X17 X7 input filter (unit: ms) 0~20ms adjustable (available for 28SS2: V3.42, 28SA2: V3.0, 26SE: V1.0 and later)	×	○	○	×	10	-	-	R/W	NO	10
D1022	Counting mode selection (Double frequency/ 4 times frequency) for AB phase counter (From X0, X1 input)	○	○	○	○	4	-	-	R/W	NO	4
D1023*	Register for Storing detected pulse width (unit: 0.1ms)	○	○	○	○	0	-	-	R/W	NO	0
D1025*	Code for communication request error	○	○	○	○	0	-	-	R	NO	0
D1026*	The pulse number for masking Y0 is set when M1156 = ON (Low word)	ES2/ EX2	○	○	○	0	0	-	R/W	NO	0
D1027*	The pulse number for masking Y0 is set when M1156 = ON (High word) If the value in the 32-bit register (D1027, D1026) is less than or equal to 0, the function will not be enabled. (Default value: 0)	ES2/ EX2	○	○	○	0	0	-	R/W	NO	0
D1028	Index register E0	○	○	○	○	0	-	-	R/W	NO	0
D1029	Index register F0	○	○	○	○	0	-	-	R/W	NO	0
D1030	PV of Y0 pulse output (Low word)	○	○	○	○	-	-	-	R/W	YES	0
D1031	PV of Y0 pulse output (High word)	○	○	○	○	-	-	-	R/W	YES	0
D1032	PV of Y1 pulse output (Low word)	ES2/ EX2	○	○	○	0	-	-	R/W	NO	0
D1033	PV of Y1 pulse output (High word)	ES2/ EX2	○	○	○	0	-	-	R/W	NO	0
D1036*	COM1 (RS-232) communication protocol	○	○	○	○	H’86	-	-	R/W	NO	H’86
D1037*	Register for setting 8-sets SPD function (has to be used with M1037) Refer to section 2.16 the description on M1037 for more information.	ES2/ EX2	○	○	○	0	-	-	R/W	NO	0

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1038	1. Delay time setting for data response when PLC is SLAVE in COM2 / COM3 RS-485 communication. Range: 0 ~ 20 (unit: 0.1ms). Refer to section 2.16 the description on D1038 for more information. 2. By using PLC LINK in COM2 (RS-485), D1038 can be set to send next communication data with delay. Range: 0 ~ 10,000 (Unit: one scan cycle)	○	○	○	○	-	-	-	R/W	NO	0
D1039*	Fixed scan time (ms)	○	○	○	○	0	-	-	R/W	NO	0
D1040	No. of the 1st step point which is ON.	○	○	○	○	0	-	-	R	NO	0
D1041	No. of the 2nd step point which is ON	○	○	○	○	0	-	-	R	NO	0
D1042	No. of the 3rd step point which is ON.	○	○	○	○	0	-	-	R	NO	0
D1043	No. of the 4th step point which is ON	○	○	○	○	0	-	-	R	NO	0
D1044	No. of the 5th step point which is ON.	○	○	○	○	0	-	-	R	NO	0
D1045	No. of the 6th step point which is ON	○	○	○	○	0	-	-	R	NO	0
D1046	No. of the 7th step point which is ON.	○	○	○	○	0	-	-	R	NO	0
D1047	No. of the 8th step point which is ON	○	○	○	○	0	-	-	R	NO	0
D1049	No. of alarm which is ON	○	○	○	○	0	-	-	R	NO	0
D1050 ↓ D1055	Processing MODRD communication data The PLC automatically converts the data in D1070~D1085 in the ASCII mode into hexadecimal values, or combines two lower 8 bits in the RTU mode into 16 bits in the RTU mode.	○	○	○	○	0	-	-	R	NO	0
D1056*	Low word of X0's input pulse frequency (Unit: 0.001Hz) It is used with M1357, available for ES2/EX2: V3.22, ES2-C: V3.68, SA2: V3.02, SX2: V2.66 or later.	ES2/ EX2	×	SA2	○	0	0	-	R	NO	0
D1057*	High word of X0's input pulse frequency (Unit: 0.001Hz) It is used with M1357, available for ES2/EX2: V3.22, ES2-C: V3.68, SA2: V3.02, SX2: V2.66 or later.	ES2/ EX2	×	SA2	○	0	0	-	R	NO	0
D1058*	Low word of X1's input pulse frequency (Unit: 0.001Hz) It is used with M1358, available for ES2/EX2: V3.22, ES2-C: V3.68, SA2: V3.02, SX2: V2.66 or later.	ES2/ EX2	×	SA2	○	0	0	-	R	NO	0
D1059*	High word of X1's input pulse frequency (Unit: 0.001Hz) It is used with M1358, available for ES2/EX2: V3.22, ES2-C: V3.68, SA2: V3.02, SX2: V2.66 or later.	ES2/ EX2	×	SA2	○	0	0	-	R	NO	0
D1062*	Average number of times an analog signal is input to the EX2/SX2 series PLC The default value is K10 for EX2 version 2.6 and version 2.8.	ES2/ EX2	×	×	○	2	-	-	R/W	YES	2
D1067*	Error code for program execution error	○	○	○	○	0	0	-	R	NO	0
D1068*	Address of program execution error	○	○	○	○	0	-	-	R	NO	0
D1070 ↓ D1085	Feedback data (ASCII) of Modbus communication. When PLC's RS-485 communication instruction receives feedback signals, the data will be saved in the registers D1070~D1085. Usres can check the received data in these registers.	○	○	○	○	0	-	-	R	NO	0
D1086	High word of the password in DVP-PCC01 (displayed in hex according to its ASCII codes)	○	○	○	○	0	-	-	R/W	NO	0
D1087	Low word of the password in DVP-PCC01 (displayed in hex according to its ASCII codes)	○	○	○	○	0	-	-	R/W	NO	0
D1089 ↓ D1099	Sent data of Modbus communication. When PLC's RS-485 communication instruction sends out data, the data will be stored in D1089~D1099. Users can check the sent data in these registers.	○	○	○	○	0	-	-	R	NO	0
D1109*	COM3 (RS-485) Communication protocol	ES2/ EX2	×	○	○	H'86	-	-	R/W	NO	H'86

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1110*	Average value of EX2/SX2 analog input channel 0 (AD 0) When average times in D1062 is set to 1, D1110 indicates present value.	EX2	×	×	○	0	-	-	R	NO	0
D1110	Modbus TCP Server port number can be modified.	ES2-E v1.46	×	SE v2.04	×	0	-	-	R/W	NO	502
D1111*	Average value of EX2/SX2 analog input channel 1 (AD 1) When average times in D1062 is set to 1, D1111 indicates present value	EX2	×	×	○	0	-	-	R	NO	0
D1111	ETHRW instruction is used to set the port number of the remote connection.	ES2-E v1.46	×	SE v2.04	×	0	-	-	R/W	NO	502
D1112*	Average value of EX2/SX2 analog input channel 2 (AD 2) Whenaverage times in D1062 is set to 1, D1112 indicates present value	EX2	×	×	○	0	-	-	R	NO	0
D1113*	Average value of 20EX2/SX2 analog input channel 3 (AD 3) Whenaverage times in D1062 is set to 1, D1113 indicates present value	EX2	×	×	○	0	-	-	R	NO	0
	Displaying the status of the analog input channel of 30EX2	EX2	×	×	×	0	-	-	R	NO	0
D1114*	Enable/disable 20EX2/SX2 AD channels (0: enable (default) / 1: disable) bit0~bit3 sets AD0~AD3. P.S. 30EX2 does not support this function.	EX2	×	×	○	0	-	-	R/W	YES	0
D1115*	20EX2/SX2 analog input/output mode setting	EX2	×	×	○	0	0	0	R/W	YES	0
	30EX2 analog input/output mode setting	EX2	×	×	×	-	-	-	R/W	YES	H'FFFF
D1116*	Output value of analog output channel 0 (DA 0) of EX2/SX2	EX2	×	×	○	0	0	0	R/W	NO	0
D1117*	Output value of analog output channel 1 (DA 0) of 20EX2/SX2 P.S. 30EX2 does not support this function.	EX2	×	×	○	0	0	0	R/W	NO	0
D1118*	EX2/SX2 sampling time of analog/digital converstion. Default: 2. Unit: 1ms. Sampling time will be regarded as 2ms if D1118≦2	EX2	×	×	○	2	-	-	R/W	YES	2
D1120*	COM2 (RS-485) communication protocol	○	○	○	○	H'86	-	-	R/W	NO	H'86
D1121*	COM1(RS-232) and COM2(RS-485) PLC communication address	○	○	○	○	-	-	-	R/W	Yes	1
D1122	COM2(RS-485) Residual number of words of transmitting data	○	○	○	○	0	0	-	R	NO	0
D1123	COM2(RS-485) When M1263 is OFF, the value in D1123 indicates the remaining bytes of data to be received. When M1263 is ON, the value in D1123 indicates the bytes of data that are received.	○	○	○	○	0	0	-	R	NO	0
D1124	COM2(RS-485) Definition of start character (STX)	○	○	○	○	H'3A	-	-	R/W	NO	H'3A
D1125	COM2(RS-485) Definition of first ending character (ETX1)	○	○	○	○	H'0D	-	-	R/W	NO	H'0D
D1126	COM2(RS-485) Definition of second ending character (ETX2)	○	○	○	○	H'0A	-	-	R/W	NO	H'0A
D1127	Number of pulses for ramp-up operation of positioning instruction (Low word)	○	○	○	○	0	-	-	R/W	NO	0
D1128	Number of pulses for ramp-up operation of positioning instruction (High word)	○	○	○	○						
D1129	COM2 (RS-485) Communication time-out setting (ms)	○	○	○	○	0	-	-	R/W	NO	0
D1130	COM2 (RS-485) Error code returning from Modbus	○	○	○	○	0	-	-	R	NO	0
D1131	Input/output percentage value of CH0(Y0,Y1) close loop control	○	○	○	○	100	-	-	R/W	NO	100
D1132	Input/output percentage value of CH1(Y2,Y3) close loop control	○	○	○	○	100	-	-	R/W	NO	100

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1133	Number of pulses for ramp-down operation of positioning instruction (Low word)	○	○	○	○	0	-	-	R	NO	0
D1134	Number of pulses for ramp-down operation of positioning instruction (High word)	○	○	○	○	0	-	-	R	NO	0
D1135*	Pulse number for masking Y2 when M1158 = ON (Low word)	ES2/ EX2	○	○	○	0	0	-	R/W	NO	0
D1136*	Pulse number for masking Y2 when M1158 = ON (High word)	ES2/ EX2	○	○	○	0	0	-	R/W	NO	0
D1137*	Address where incorrect use of operand occurs	○	○	○	○	0	0	-	R	NO	0
D1140*	Number of I/O modules (max. 8)	ES2/ EX2	○	○	○	0	-	-	R	NO	0
D1142*	Number of input points (X) on DIO modules	○	○	○	○	0	-	-	R	NO	0
D1143*	Number of output points (Y) on DIO modules	○	○	○	○	0	-	-	R	NO	0
D1145*	Number of the connected let-side modules	×	×	○	○	0	-	-	R	NO	0
D1150*	Vale fetched from the hardware counter C243/C245/C246/C247/C248/C251/C252 (Low word)	V 3.28	V 3.28	SA2: V2.82	V2.82	0	-	-	R/W	NO	0
D1151*	Value fetched from the hardware counter C243/C245/C246/C247/C248/C251/C252 (High word)	V 3.28	V 3.28	SA2: V2.82	V2.82	0	-	-	R/W	NO	0
D1152*	Value fetched from the hardware counter C244/C249/C250/C253/C254 (Low word)	V 3.28	V 3.28	SA2: V2.82	V2.82	0	-	-	R/W	NO	0
D1153*	Value fetched from the hardware conter C244/C249/C250/C253/C254 (High word)	V 3.28	V 3.28	SA2: V2.82	V2.82	0	-	-	R/W	NO	0
D1167	The specific end word to be detected for RS instruction to execute an interruption request (I140) on COM1 (RS-232).	○	○	○	○	0	-	-	R/W	NO	0
D1168	For RS instruction COM2 (RS-485): When M1263 is ON, the value in D1168 indicates a specific end word to be seen as an interrupt request; unit: ms. It is suggested to set this value shorter than what the timeout value is set in D1129. When M1263 is OFF, and the received data is low-byte of a specific word, the interrupt (I150) is triggered.	○	○	○	○	0	-	-	R/W	NO	0
D1169	The specific end word to be detected for RS instruction to execute an interruption request (I160) on COM3 (RS-485)	ES2/ EX2	×	○	×	0	-	-	R/W	NO	0
D1175	Number of communication packets received via broadcasting (number of slaves) by executing CANRS instruction	ES2-C	×	○	○	0	0	-	R	NO	0
D1176	Error code for ETHRS instruction (available for ES2-E: V1.2, 12SE: V1.92, 26SE: V2.00 and later)	ES2-C	×	SE	×	0	0	-	R	NO	0
D1177	Communication timeout setting for CANRS instruction (available for ES2-C: V3.48, SA2/SX2: V2.89, SE: V1.83 and later)	ES2-C	×	○	○	200	-	-	R/W	NO	200
D1178	VR0 value	×	×	○	○	0	-	-	R	NO	0
D1179	VR1 value	×	×	○	○	0	-	-	R	NO	0
D1182	Index register E1	○	○	○	○	0	-	-	R/W	NO	0
D1183	Index register F1	○	○	○	○	0	-	-	R/W	NO	0
D1184	Index register E2	○	○	○	○	0	-	-	R/W	NO	0
D1185	Index register F2	○	○	○	○	0	-	-	R/W	NO	0
D1186	Index register E3	○	○	○	○	0	-	-	R/W	NO	0
D1187	Index register F3	○	○	○	○	0	-	-	R/W	NO	0
D1188	Index register E4	○	○	○	○	0	-	-	R/W	NO	0
D1189	Index register F4	○	○	○	○	0	-	-	R/W	NO	0
D1190	Index register E5	○	○	○	○	0	-	-	R/W	NO	0
D1191	Index register F5	○	○	○	○	0	-	-	R/W	NO	0
D1192	Index register E6	○	○	○	○	0	-	-	R/W	NO	0
D1193	Index register F6	○	○	○	○	0	-	-	R/W	NO	0
D1194	Index register E7	○	○	○	○	0	-	-	R/W	NO	0
D1195	Index register F7	○	○	○	○	0	-	-	R/W	NO	0

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1220	Output mode setting for the 1st group pulse CH0 (Y0, Y1)	○	○	○	○	0	-	-	R/W	NO	0
D1221	Output mode setting for the 2nd group pulse CH1 (Y2, Y3)	○	○	○	○	0	-	-	R/W	NO	0
D1222	Output mode setting for the 3rd group pulse CH2 (Y4, Y5)	EC5	×	×	×	0	-	-	R/W	NO	0
D1223	Output mode setting for the 4th group pulse CH3 (Y6, Y7)	EC5	×	×	×	0	-	-	R/W	NO	0
D1227 D1228	Sender's IP address when executing ETHRS instruction in receiving mode (available for ES2-E: V1.2, 12SE: V1.92, 26SE: V2.00)	ES2-C	×	SE	×	0	0	-	R	NO	0
D1229~ D1231	Network MAC address (hexadecimal format ex: 12:34:56:78:9A:BC => D1229=H'1234, D1230=H'5678, D1231=H'9ABC) (available for ES2-E: V1.2, 12SE: V1.92, 26SE: V2.00)	ES2-C	×	SE	×	0	0	-	R	YES	0
D1232*	Number of output pulses for CH0 (Y0, Y1) ramp-down stop when mark sensor receives signals. (Low word).	ES2/ EX2	○	○	○	0	0	--	R/W	NO	0
D1233*	Number of output pulses for CH0 (Y0, Y1) ramp-down stop when mark sensor receives signals. (High word).	ES2/ EX2	○	○	○	0	0	--	R/W	NO	0
D1234*	Number of output pulses for CH1 (Y2, Y3) ramp-down stop when mark sensor receives signals. (Low word).	ES2/ EX2	○	○	○	0	0	--	R/W	NO	0
D1235*	Number of output pulses for CH2 (Y2, Y3) ramp-down stop when mark sensor receives signals. (High word).	ES2/ EX2	○	○	○	0	0	--	R/W	NO	0
D1240*	When interupt I400/I401/I100/I101 occurs, D1240 stores the low word of high-speed counter.	○	○	○	○	0	0	-	R	NO	0
D1241*	When interupt I400/I401/I100/I101 occurs, D1241 stores the high Word of high-speed counter.	○	○	○	○	0	0	-	R	NO	0
D1242*	When interupt I500/I501/I300/I301 occurs, D1242 stores the low Wordof high-speed counter.	○	○	○	○	0	0	-	R	NO	0
D1243*	When interupt I500/I501/I300/I301 occurs, D1243 stores the high Word of high-speed counter.	○	○	○	○	0	0	-	R	NO	0
D1244	Idle time (pulse number) setting of CH0 (Y0, Y1) The function is disabled if set value≦0.	○	○	○	○	0	-	-	R/W	NO	0
D1245	Idle time (pulse number) setting of CH1 (Y2, Y3) The function is disabled if set value≦0.	○	○	○	○	0	-	-	R/W	NO	0
D1246*	Low word of X2's input pulse frequency (Unit: 0.01Hz) It is used with M1359, available for ES2/EX2: V3.22, ES2-C: V3.68, SA2: V3.02, SX2: V2.66 or later	ES2/ EX2	×	SA2	○	0	0	-	R	NO	0
D1247*	High word of X2's input pulse frequency (Unit: 0.01Hz) It is used with M1359, available for ES2/EX2: V3.22, ES2-C: V3.68, SA2: V3.02, SX2: V2.66 or later	ES2/ EX2	×	SA2	○	0	0	-	R	NO	0
D1248	Enter the number of input points on DIO module for the PLC to check if the number is matched with the value in D1142 when power-on. Available for ES2/EX2/ES2C: V3.62, ES2E: V1.48 or later	ES2/ EX2	×	×	×	-	-	-	R/W	YES	0
D1249	Set value for COM1 (RS-232) data receiving time-out (Unit: 1ms, min. 50ms, value smaller than 50ms will be regarded as 50ms) (only applicable for MODRW/RS instruction) In RS instruction, no time-out setting if "0" is specified.	○	○	○	○	0	-	-	R/W	NO	0
D1250	COM1 (RS-232) communication error code (only applicable for MODRW/RS instruction)	○	○	○	○	0	-	-	R/W	NO	0

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1252	Set value for COM3 (RS-485) data receiving time-out (Unit: 1ms, min. 50ms, value smaller than 50ms will be regarded as 50ms) (only applicable for MODRW/RS instruction) In RS instruction, no time-out setting if "0" is specified	ES2/ EX2	×	○	×	50	-	-	R/W	NO	50
D1253	COM3 (RS-485) communication error code (only applicable for MODRW/RS instruction)	ES2/ EX2	×	○	×	0	-	-	R/W	NO	0
D1254	Enter the number of output points on DIO module for the PLC to check if the number is matched with the value in D1143 when power-on. Available for ES2/EX2/ES2C: V3.62, ES2E: V1.48 or later	ES2/ EX2	×	×	×	-	-	-	R/W	YES	0
D1255*	COM3 (RS-485) PLC communication address	ES2/ EX2	×	○	○	50	-	-	R/W	YES	1
D1256 ↓ D1295	For COM2 RS-485 MODRW instruction. D1256~D1295 store the sent data of MODRW instruction. When MODRW instruction sends out data, the data will be stored in D1256~D1295. Users can check the sent data in these registers.	○	○	○	○	0	-	-	R	NO	0
D1296 ↓ D1311	For COM2 RS-485 MODRW instruction. D1296~D1311 store the converted hex data from D1070 ~ D1085 (ASCII). PLC automatically converts the received ASCII data in D1070 ~ D1085 into hex data.	○	○	○	○	0	-	-	R	NO	0
D1312*	Specify the number of additional pulses for additional pulses output and Z-phase seeking function of ZRN instruction (Has to be used with M1308)	○	×	○	○	0	0	-	R/W	NO	0
D1313*	Second of RTC: 00 ~ 59	○	○	○	○	-	-	-	R/W	YES	0
D1314*	Minute of RTC: 00 ~ 59	○	○	○	○	-	-	-	R/W	YES	0
D1315*	Hour of RTC: 00 ~ 23	○	○	○	○	-	-	-	R/W	YES	0
D1316*	Day of RTC: 01 ~ 31	○	○	○	○	-	-	-	R/W	YES	1
D1317*	Month of RTC: 01 ~ 12	○	○	○	○	-	-	-	R/W	YES	1
D1318*	Week of RTC: 1 ~ 7	○	○	○	○	-	-	-	R/W	YES	2
D1319*	Year of RTC: 00 ~ 99 (A.D.)	○	○	○	○	-	-	-	R/W	YES	8
D1320*	ID of the 1st right side module	ES2/ EX2	×	×	×	0	-	-	R	NO	0
D1321*	ID of the 2nd right side module	ES2/ EX2	×	×	×	0	-	-	R	NO	0
D1322*	ID of the 3rd right side module	ES2/ EX2	×	×	×	0	-	-	R	NO	0
D1323*	ID of the 4th right side module	ES2/ EX2	×	×	×	0	-	-	R	NO	0
D1324*	ID of the 5th right side module	ES2/ EX2	×	×	×	0	-	-	R	NO	0
D1325*	ID of the 6th right side module	ES2/ EX2	×	×	×	0	-	-	R	NO	0
D1326*	ID of the 7th right side module	ES2/ EX2	×	×	×	0	-	-	R	NO	0
D1327*	ID of the 8th right side module	ES2/ EX2	×	×	×	0	-	-	R	NO	0
D1336	PV of Y2 pulse output (Low word)	○	○	○	○	-	-	-	R/W	YES	0
D1337	PV of Y2 pulse output (High word)	○	○	○	○	-	-	-	R/W	YES	0
D1338	PV of Y3 pulse output (Low word)	ES2/ EX2	○	○	○	-	-	-	R/W	NO	0
D1339	PV of Y3 pulse output (High word)	ES2/ EX2	○	○	○	-	-	-	R/W	NO	0
D1340	Start/end frequency of the 1st group pulse output CH0 (Y0, Y1)	○	○	○	○	100	-	-	R/W	NO	100
D1343	Ramp up/down time of the 1st group pulse output CH0 (Y0, Y1)	○	○	○	○	100	-	-	R/W	NO	100
D1348*	When M1534 = ON, set the ramp-down time of CH0 (Y0, Y1) pulse output in D1348.	○	○	○	○	100	-	-	R/W	NO	100
D1349*	When M1535 = ON, set the ramp-down time of CH1 (Y2, Y3) pulse output in D1349.	○	○	○	○	100	-	-	R/W	NO	100

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
D1350*	When M1536 = ON, set the ramp-down time of CH2 (Y4, Y5) pulse output in D1350.	EC5	×	×	×	100	-	-	R/W	NO	100
D1351*	When M1537 = ON, set the ramp-down time of CH3 (Y6, Y7) pulse output in D1351.	EC5	×	×	×	100	-	-	R/W	NO	100
D1352	Start/end frequency of the 2nd group pulse output CH1 (Y2, Y3)	○	○	○	○	100	-	-	R/W	NO	100
D1353	Ramp up/down time of the 2nd group pulse output CH1 (Y2, Y3)	○	○	○	○	100	-	-	R/W	NO	100
D1354	PLC Link scan cycle (Unit: 1ms) ■ Max: K32000 ■ D1354 = K0 when PLC Link stops or when the first scan is completed	○	○	○	○	0	0	0	R	NO	0
D1355*	Starting address for PLC LINK connection ID#1 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1356*	Starting address for PLC LINK connection ID#2 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1357*	Starting address for PLC LINK connection ID#3 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1358*	Starting address for PLC LINK connection ID#4 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1359*	Starting address for PLC LINK connection ID#5 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1360*	Starting address for PLC LINK connection ID#6 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1361*	Starting address for PLC LINK connection ID#7 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1362*	Starting address for PLC LINK connection ID#8 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1363*	Starting address for PLC LINK connection ID#9 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1364*	Starting address for PLC LINK connection ID#10 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1365*	Starting address for PLC LINK connection ID#11 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1366*	Starting address for PLC LINK connection ID#12 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1367*	Starting address for PLC LINK connection ID#13 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1368*	Starting address for PLC LINK connection ID#14 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1369*	Starting address for PLC LINK connection ID#15 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1370*	Starting address for PLC LINK connection ID#16 to read	○	○	○	○	-	-	-	R/W	YES	H'1064
D1375	PV of Y4 pulse output (Low word)	EC5	×	×	×	-	-	-	R/W	YES	0
D1376	PV of Y4 pulse output (High word)	EC5	×	×	×	-	-	-	R/W	YES	0
D1377	PV of Y6 pulse output (Low word)	EC5	×	×	×	-	-	-	R/W	YES	0
D1378	PV of Y6 pulse output (High word)	EC5	×	×	×	-	-	-	R/W	YES	0
D1379	Start/end frequency of the 3rd group pulse output CH2 (Y4, Y5)	EC5	×	×	×	100	-	-	R/W	NO	100
D1380	Start/end frequency of the 4th group pulse output CH3 (Y6, Y7)	EC5	×	×	×	100	-	-	R/W	NO	100
D1381	Ramp up/down time of the 3rd group pulse output CH2 (Y4, Y5)	EC5	×	×	×	100	-	-	R/W	NO	100
D1382	Ramp up/down time of the 4th group pulse output CH3 (Y6, Y7)	EC5	×	×	×	100	-	-	R/W	NO	100
D1385	Enter the number of connected module for the PLC to check if the number is matched with the value in D1143 when power-on. Available for ES2/EX2/ES2C: V3.62, ES2E: V1.48 or later	ES2/ EX2	×	×	×	-	-	-	R/W	YES	0
D1386	ID of the 1st left side module	×	×	○	○	0	-	-	R	NO	0
D1387	ID of the 2nd left side module	×	×	○	○	0	-	-	R	NO	0

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1388	ID of the 3rd left side module	╳	╳	○	○	0	-	-	R	NO	0
D1389	ID of the 4th left side module	╳	╳	○	○	0	-	-	R	NO	0
D1390	ID of the 5th left side module	╳	╳	○	○	0	-	-	R	NO	0
D1391	ID of the 6th left side module	╳	╳	○	○	0	-	-	R	NO	0
D1392	ID of the 7th left side module	╳	╳	○	○	0	-	-	R	NO	0
D1393	ID of the 8th rleft side module	╳	╳	○	○	0	-	-	R	NO	0
D1399*	Starting ID of Slave designated by PLC LINK	○	○	○	○	-	-	-	R/W	YES	1
D1400	Read MAC address from the left side module (ex: the 1st is K100, the 8th is K107) should work with M1145, refer to M1145 for module availability	╳	╳	○	○	-	-	-	R/W	NO	0
D1401~ D1403	Put MAC address in a consecutive order	╳	╳	○	○	-	-	-	R	NO	0
D1410	Set the maximum frequency of the ramp up/down for the fixed slope of Y0 or CH0 (Y0/Y1) in D1410 (Low word). You need to use M1604 to enable this function. Available for ES2/EX2: V3.28, ES2-E: V1.00, EC5: V1.00, 12SA2/SX2: V2.82, SS2: V3.24, SE: V2.00, 28SA2: V2.90.	○	○	○	○	-	-	-	R/W	NO	-
D1411	Set the maximum frequency of the ramp up/down for the fixed slope of Y0 or CH0 (Y0/Y1) in D1411 (High word). You need to use M1604 to enable this function. Available for ES2/EX2: V3.28, ES2-E: V1.00, EC5: V1.00, 12SA2/SX2: V2.82, SS2: V3.24, SE: V2.00, 28SA2: V2.90.	○	○	○	○	-	-	-	R/W	NO	-
D1412	Set the maximum frequency of the ramp up/down for the fixed slope of Y2 or CH1 (Y2/Y3) in D1412 (Low word). You need to use M1605 to enable this function. Available for ES2/EX2: V3.28, ES2-E: V1.00, EC5: V1.00, 12SA2/SX2: V2.82, SS2: V3.24, SE: V2.00, 28SA2: V2.90.	○	○	○	○	-	-	-	R/W	NO	-
D1413	Set the maximum frequency of the ramp up/down for the fixed slope of Y2 or CH1 (Y2/Y3) in D1413 (High word). You need to use M1605 to enable this function. Available for ES2/EX2: V3.28, ES2-E: V1.00, EC5: V1.00, 12SA2/SX2: V2.82, SS2: V3.24, SE: V2.00, 28SA2: V2.90.	○	○	○	○	-	-	-	R/W	NO	-
D1415*	Starting address for PLC LINK connection ID#1 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1416*	Starting address for PLC LINK connection ID#2 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1417*	Starting address for PLC LINK connection ID#3 to write	○	○	○	○	-	-	-	R/W	YES	10C8
D1418*	Starting address for PLC LINK connection ID#4 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1419*	Starting address for PLC LINK connection ID#5 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1420*	Starting address for PLC LINK connection ID#6 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1421*	Starting address for PLC LINK connection ID#7 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1422*	Starting address for PLC LINK connection ID#8 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1423*	Starting address for PLC LINK connection ID#9 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1424*	Starting address for PLC LINK connection ID#10 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1425*	Starting address for PLC LINK connection ID#11 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1426*	Starting address for PLC LINK connection ID#12 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1427*	Starting address for PLC LINK connection ID#13 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1428*	Starting address for PLC LINK connection ID#14 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1429*	Starting address for PLC LINK connection ID#15 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1430*	Starting address for PLC LINK connection ID#16 to write	○	○	○	○	-	-	-	R/W	YES	H'10C8
D1431*	Setting times for PLC LINK polling cycle	○	○	○	○	0	-	-	R/W	NO	0
D1432*	Current times of PLC LINK polling cycle	○	○	○	○	0	-	-	R/W	NO	0
D1433*	Number of connected modules on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1434*	Data length to be read for PLC LINK connection ID#1	○	○	○	○	-	-	-	R/W	YES	16
D1435*	Data length to be read for PLC LINK connection ID#2	○	○	○	○	-	-	-	R/W	YES	16
D1436*	Data length to be read for PLC LINK connection ID#3	○	○	○	○	-	-	-	R/W	YES	16
D1437*	Data length to be read for PLC LINK connection ID#4	○	○	○	○	-	-	-	R/W	YES	16
D1438*	Data length to be read for PLC LINK connection ID#5	○	○	○	○	-	-	-	R/W	YES	16
D1439*	Data length to be read for PLC LINK connection ID#6	○	○	○	○	-	-	-	R/W	YES	16
D1440*	Data length to be read for PLC LINK connection ID#7	○	○	○	○	-	-	-	R/W	YES	16
D1441*	Data length to be read for PLC LINK connection ID#8	○	○	○	○	-	-	-	R/W	YES	16
D1442*	Data length to be read for PLC LINK connection ID#9	○	○	○	○	-	-	-	R/W	YES	16
D1443*	Data length to be read for PLC LINK connection ID#10	○	○	○	○	-	-	-	R/W	YES	16
D1444*	Data length to be read for PLC LINK connection ID#11	○	○	○	○	-	-	-	R/W	YES	16
D1445*	Data length to be read for PLC LINK connection ID#12	○	○	○	○	-	-	-	R/W	YES	16
D1446*	Data length to be read for PLC LINK connection ID#13	○	○	○	○	-	-	-	R/W	YES	16
D1447*	Data length to be read for PLC LINK connection ID#14	○	○	○	○	-	-	-	R/W	YES	16
D1448*	Data length to be read for PLC LINK connection ID#15	○	○	○	○	-	-	-	R/W	YES	16
D1449*	Data length to be read for PLC LINK connection ID#16	○	○	○	○	-	-	-	R/W	YES	16
D1450*	Data length to be written for PLC LINK connection ID#1	○	○	○	○	-	-	-	R/W	YES	16
D1451*	Data length to be written for PLC LINK connection ID#2	○	○	○	○	-	-	-	R/W	YES	16
D1452*	Data length to be written for PLC LINK connection ID#3	○	○	○	○	-	-	-	R/W	YES	16
D1453*	Data length to be written for PLC LINK connection ID#4	○	○	○	○	-	-	-	R/W	YES	16
D1454*	Data length to be written for PLC LINK connection ID#5	○	○	○	○	-	-	-	R/W	YES	16
D1455*	Data length to be written for PLC LINK connection ID#6	○	○	○	○	-	-	-	R/W	YES	16
D1456*	Data length to be written for PLC LINK connection ID#7	○	○	○	○	-	-	-	R/W	YES	16
D1457*	Data length to be written for PLC LINK connection ID#8	○	○	○	○	-	-	-	R/W	YES	16
D1458*	Data length to be written for PLC LINK connection ID#9	○	○	○	○	-	-	-	R/W	YES	16
D1459*	Data length to be written for PLC LINK connection ID#10	○	○	○	○	-	-	-	R/W	YES	16
D1460*	Data length to be written for PLC LINK connection ID#11	○	○	○	○	-	-	-	R/W	YES	16
D1461*	Data length to be written for PLC LINK connection ID#12	○	○	○	○	-	-	-	R/W	YES	16
D1462*	Data length to be written for PLC LINK connection ID#13	○	○	○	○	-	-	-	R/W	YES	16
D1463*	Data length to be written for PLC LINK	○	○	○	○	-	-	-	R/W	YES	16

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
	connection ID#14										
D1464*	Data length to be written for PLC LINK connection ID#15	○	○	○	○	-	-	-	R/W	YES	16
D1465*	Data length to be written for PLC LINK connection ID#16	○	○	○	○	-	-	-	R/W	YES	16
D1480* ↓ D1495*	The data which is read from connection ID#1 on PLC LINK at the time when M1353 is OFF	○	○	○	○	0	-	-	R	NO	0
	The initial data register where the data read from connection ID#1~ID#16 on PLC LINK is stored at the time when M1353 is ON	○	○	○	○	-	-	-	R	YES	0
D1496* ↓ D1511*	The data which is written into connection ID#1 on PLC LINK at the time when M1353 is OFF	○	○	○	○	0	-	-	R/W	NO	0
	The initial data register where the data written into connection ID#1~ID#16 on PLC LINK is stored at the time when M1353 is ON	○	○	○	○	-	-	-	R/W	YES	0
D1512* ↓ D1527*	The data which is read from connection ID#2 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1528* ↓ D1543*	The data which is written into connection ID#2 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1544* ↓ D1559*	The data which is read from connection ID#3 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1560* ↓ D1575*	The data which is written into connection ID#3 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1576* ↓ D1591*	The data which is read from connection ID#4 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1592* ↓ D1607*	The data which is written into connection ID#4 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1608* ↓ D1623*	The data which is read from connection ID#5 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1624* ↓ D1639*	The data which is written into connection ID#5 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1640* ↓ D1655*	The data which is read from connection ID#6 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1656* ↓ D1671*	The data which is written into connection ID#6 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1672* ↓ D1687*	The data which is read from connection ID#7 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1688* ↓ D1703*	The data which is written into connection ID#7 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1704* ↓ D1719*	The data which is read from connection ID#8 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1720* ↓ D1735*	The data which is written into connection ID#8 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1736* ↓ D1751*	The data which is read from connection ID#9 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1752* ↓ D1767*	The data which is written into connection ID#9 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1768* ↓ D1783*	The data which is read from connection ID#10 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1784* ↓ D1799*	The data which is written into connection ID#10 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1800* ↓ D1815*	The data which is read from connection ID#11 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1816* ↓ D1831*	The data which is written into connection ID#11 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1832* ↓ D1847*	The data which is read from connection ID#12 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1848* ↓ D1863*	The data which is written into connection ID#12 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1864* ↓ D1879*	The data which is read from connection ID#13 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1880* ↓ D1895*	The data which is written into connection ID#13 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1896* ↓ D1911*	The data which is read from connection ID#14 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1900* ↓ D1931*	Specify the station number for PLC-Link when M1356 is ON. Consecutive station numbers set by D1399 will be invalid in this case. Note that the registers are latched only when M1356 is ON.	ES2/ EX2	╳	○	○	0	-	-	R/W	NO	
D1912* ↓ D1927*	The data which is written into connection ID#14 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1928* ↓ D1943*	The data which is read from connection ID#15 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1944* ↓ D1959*	The data which is written into connection ID#15 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1960* ↓ D1975*	The data which is read from connection ID#16 on PLC LINK	○	○	○	○	0	-	-	R	NO	0
D1976* ↓ D1991*	The data which is written into connection ID#16 on PLC LINK	○	○	○	○	0	-	-	R/W	NO	0
D1970	Set the maximum frequency of the ramp up/down for the fixed slope of CH2 (Y4/Y5) in D1970 (Low word). You need to use M1610 to enable this function. Or you can with PLC Link via this register.	EC5	╳	╳	╳	-	-	-	RW	NO	-
D1971	Set the maximum frequency of the ramp up/down for the fixed slope of CH2 (Y4/Y5) in D1971 (High word). You need to use M1610 to enable this function. Or you can with PLC Link via this register.	EC5	╳	╳	╳	-	-	-	RW	NO	-
D1972	Set the maximum frequency of the ramp up/down for the fixed slope of CH3 (Y6/Y7) in D1972 (Low word). You need to use M1611 to enable this function. Or you can with PLC Link via this register.	EC5	╳	╳	╳	-	-	-	RW	NO	-
D1973	Set the maximum frequency of the ramp up/down for the fixed slope of CH3 (Y6/Y7) in D1973 (High word). You need to use	EC5	╳	╳	╳	-	-	-	RW	NO	-

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
	M1611 to enable this function. Or you can with PLC Link via this register.										
D1988	Set the maximum frequency of the ramp up/down for the fixed slope of Y1 in D1988 (Low word). You need to use M1606 to enable this function. Or you can with PLC Link via this register. Available for ES2/EX2: V3.28, ES2-E: V1.00, 12SA2/SX2: V2.82, SE: V2.00, 28SA2: V2.90 or later.	ES2/ EX2	×	○	○	-	-	-	RW	NO	-
	Set the maximum frequency of the ramp up/down for the fixed slope of CH2 (Y4/Y5) in D1988 (Low word). You need to use M1610 to enable this function. Or you can with PLC Link via this register.	EC5	×	○	○	-	-	-	RW	NO	-
D1989	Set the maximum frequency of the ramp up/down for the fixed slope of Y1 in D1989 (High word). You need to use M1606 to enable this function. Or you can with PLC Link via this register. Available for ES2/EX2: V3.28, ES2-E: V1.00, 12SA2/SX2: V2.82, SE: V2.00, 28SA2: V2.90 or later.	ES2/ EX2	×	○	○	-	-	-	RW	NO	-
	Set the maximum frequency of the ramp up/down for the fixed slope of CH2 (Y4/Y5) in D1989 (High word). You need to use M1610 to enable this function. Or you can with PLC Link via this register.	EC5	×	○	○	-	-	-	RW	NO	-
D1990	Set the maximum frequency of the ramp up/down for the fixed slope of Y3 in D1990 (Low word). You need to use M1607 to enable this function. Or you can with PLC Link via this register. Available for ES2/EX2: V3.28, ES2-E: V1.00, 12SA2/SX2: V2.82, SE: V2.00, 28SA2: V2.90 or later.	ES2/ EX2	×	○	○	-	-	-	RW	NO	-
	Set the maximum frequency of the ramp up/down for the fixed slope of CH3 (Y6/Y7) in D1990 (Low word). You need to use M1611 to enable this function. Or you can with PLC Link via this register.	EC5	×	○	○	-	-	-	RW	NO	-
D1991	Set the maximum frequency of the ramp up/down for the fixed slope of Y3 in D1991 (High word). You need to use M1607 to enable this function. Or you can with PLC Link via this register. Available for ES2/EX2: V3.28, ES2-E: V1.00, 12SA2/SX2: V2.82, SE: V2.00, 28SA2: V2.90 or later.	ES2/ EX2	×	○	○	-	-	-	RW	NO	-
	Set the maximum frequency of the ramp up/down for the fixed slope of CH3 (Y6/Y7) in D1991 (High word). You need to use M1611 to enable this function. Or you can with PLC Link via this register.	EC5	×	○	○	-	-	-	RW	NO	-
D1994	Remaining times for PLC password setting on DVP-PCC01	○	○	○	○	0					
D1995	Data length for PLC ID Setting on DVP-PCC01	○	○	○	○	0	-	-	R/W	NO	0
D1996	1st Word of PLC ID Setting for DVP-PCC01 (Indicated by Hex format corresponding to ASCII codes)	○	○	○	○	0	-	-	R/W	NO	0

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1997	2nd Word of PLC ID Setting for DVP-PCC01 (Indicated by Hex format corresponding to ASCII codes)	○	○	○	○	0	-	-	R/W	NO	0
D1998	3rd Word of PLC ID Setting for DVP-PCC01 (Indicated by Hex format corresponding to ASCII codes)	○	○	○	○	0	-	-	R/W	NO	0
D1999	4th word of PLC ID Setting for DVP-PCC01 (Indicated by Hex format corresponding to ASCII codes)	○	○	○	○	0	-	-	R/W	NO	0
D6000	Axis number that has error during CANopen communication on Delta Servo (applicable to ES2-C)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6001	Error code of CANopen communication on Delta Servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6002	Record the STEP that error occurs duringCANopen communication on Delta Servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6008	The PR command of the Delta CANopen communication axis 1 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6009	PR command of the Delta CANopen communication axis 2 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6010	PR command of the Delta CANopen communication axis 3 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6011	PR command of the Delta CANopen communication axis 4 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6012	PR command of the Delta CANopen communication axis 5 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6013	PR command of the Delta CANopen communication axis 6 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6014	PR command of the Delta CANopen communication axis 7 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6015	PR command of the Delta CANopen communication axis 8 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6016	Alarm code of the Delta CANopen communication axis 1 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6017	Alarm code of the Delta CANopen communication axis 2 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6018	Alarm code of the Delta CANopen communication axis 3 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6019	Alarm code of the Delta CANopen communication axis 4 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6020	Alarm code of the Delta CANopen communication axis 5 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6021	Alarm code of the Delta CANopen communication axis 6 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6022	Alarm code of the Delta CANopen communication axis 7 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6023	Alarm code of the Delta CANopen communication axis 8 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6024	The DO state of the Delta CANopen communication axis 1 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6025	The DO state of the Delta CANopen communication axis 2 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6026	The DO state of the Delta CANopen communication axis 3 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6027	The DO state of the Delta CANopen communication axis 4 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6028	The DO state of the Delta CANopen communication axis 5 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6029	The DO state of the Delta CANopen communication axis 6 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0

Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D6030	The DO state of the Delta CANopen communication axis 7 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6031	The DO state of the Delta CANopen communication axis 8 from the Delta servo	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6032 D6033	Current position of the Delta CANopen communication axis 1 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6034 D6035	Current position of the Delta CANopen communication axis 2 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6036 D6037	Current position of the Delta CANopen communication axis 3 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6038 D6039	Current position of the Delta CANopen communication axis 4 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6040 D6041	Current position of the Delta CANopen communication axis 5 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6042 D6043	Current position of the Delta CANopen communication axis 6 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6044 D6045	Current position of the Delta CANopen communication axis 7 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6046 D6047	Current position of the Delta CANopen communication axis 8 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6048 D6049	Target position of the Delta CANopen communication axis 1 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6050 D6051	Target position of the Delta CANopen communication axis 2 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6052 D6053	Target position of the Delta CANopen communication axis 3 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6054 D6055	Target position of the Delta CANopen communication axis 4 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6056 D6057	Target position of the Delta CANopen communication axis 5 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6058 D6059	Target position of the Delta CANopen communication axis 6 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6060 D6061	Target position of the Delta CANopen communication axis 7 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	0
D6062 D6063	Target position of the Delta CANopen communication axis 8 from the Delta servo (32-bit)	ES2-C: V2.8	×	×	×	0	-	-	R	NO	
D9800~ D9879	They are for left-side special modules which are connected to an SA2/SX2/SE series MPU.	×	×	○	○	-	-	-	R/W	NO	0
D9900~ D9979	They are for special modules connected to an ES2/EX2 series MPU. (Please refer to **DVP-PLC Operation Manual – Modules** for more information) They are for right-side special modules connected to an SA2/SX2/SE series MPU.	ES2/ EX2 module	×	○	○	-	-	-	R/W	NO	0
D9980	CANopen status message code (Only for DVP-ES2-C series MPUs)	ES2-C:	×	×	×	0	-	-	R	NO	0
D9981~ D9996	(Only for DVP-ES2-C series MPUs) CANopen status message code in connection station 1~connection station 16	ES2-C:	×	×	×	0	-	-	R	NO	0
D9998	Bit0~15 represent station 1~station 16. If a bit is ON, an error occurs.	ES2-C:	×	×	×	H' FFFF	-	-	R	NO	0

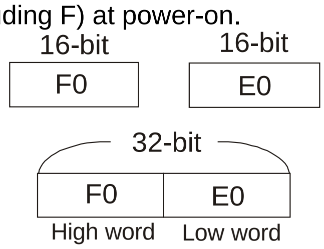
Special D	Content	ES2 EX2 EC5	SS2	SA2 SE	SX2	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
	(It is only applicable to DVP-ES2-C series MPUs. If DVP-ES2-C V3.24 (or above) is turned from OFF to ON, the value in D9998 will be H'0. If DVP-ES2-C V3.26 (or above) is turned from OFF to ON, the value in D9998 will be H'FFFF.)										
D9999	Showing the CAN baud rate K1: 20K; K2: 50K; K3: 125K; K4: 250K; K5: 500K; K6: 1M (It is only applicable to DVP-ES2-C V3.26 and above.)	ES2-C: V3.26	×	×	×	0	-	-	R	NO	0

2.14 E, F Index Registers

Index registers are used as modifiers to indicate a specified device (word, double word) by defining an offset. Devices can be modified includes byte device (KnX, KnY, KnM, KnS, T, C, D) and bit device (X, Y, M, S). E, F registers cannot be used for modifying constant (K, H) Index registers not used as a modifier can be used as general purpose register.

Index register [E], [F]

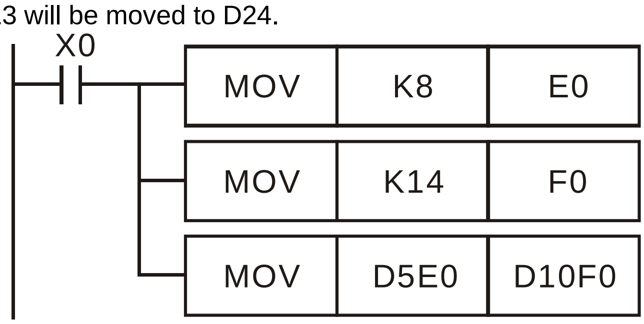
Index registers are 16-bit registers which can be read and written. There are 16 points indicated as E0~E7 and F0~F7. If you need a 32-bit register, you have to designate E. In this case, F will be covered up by E and cannot be used. It is recommended to use instruction DMOVP K0 E to reset E (including F) at power-on.



The combinations of E and F when designating a 32-bit register are:
(E0, F0) , (E1, F1) (E2, F2) (E3, F3) (E4, F4) , (E5, F5) (E6, F6) (E7, F7)

Example:

When X0 = ON and E0 = 8, F0 = 14, D5E0 = D(5+8) = D13, D10F0 = D(10+14) = D24, the content in D13 will be moved to D24.



2.15 Nest Level Pointer [N], Pointer [P], Interrupt Pointer [I]

Pointer	N	Master control nested		N0~N7, 8 points	The control point of master control nested
Pointer	P	For CJ, CALL instructions		P0~P255, 256 points	The location point of CJ, CALL
Pointer	I	For interrupt	External interrupt	I000/I001(X0), I100/I101(X1), I200/I201(X2), I300/I301(X3), I400/I401(X4), I500/I501(X5), I600/I601(X6), I700/I701(X7), 8 points (01, rising-edge trigger , 00, falling-edge trigger)	The location point of interrupt subroutine.
			Timer interrupt	I602/I699, I702/I799, 2 points (Timer resolution=1 ms), I805/I899, 1 point (Timer resolution=0.1 ms) (available for SE/ES2-E, for other series, firmware version should be V2.00 or later)	
			High-speed counter interrupt	I010, I020, I030, I040, I050, I060, I070, I080, 8 points	
			Communication interrupt	I140(COM1: RS232), I150(COM2: RS-485), I160(COM3: RS-485), 3 points	

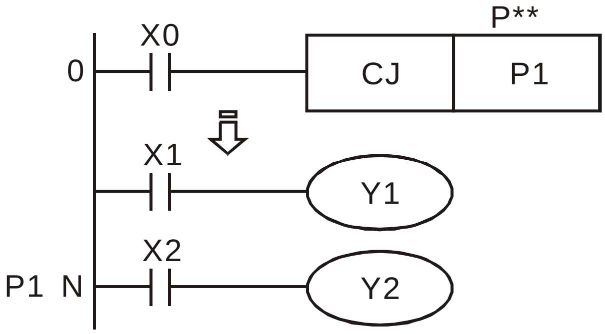
Nest Level Pointer N: used with instruction MC and MCR. MC is master start instruction. When the MC instruction is executed, the instructions between MC and MCR will be executed normally. MC-MCR master control instruction is nested level structure and max. 8 levels can be applicable, which is numbered from N0 to N7.

Pointer P: used with application instructions CJ, CALL, and SRET.

CJ condition jump:

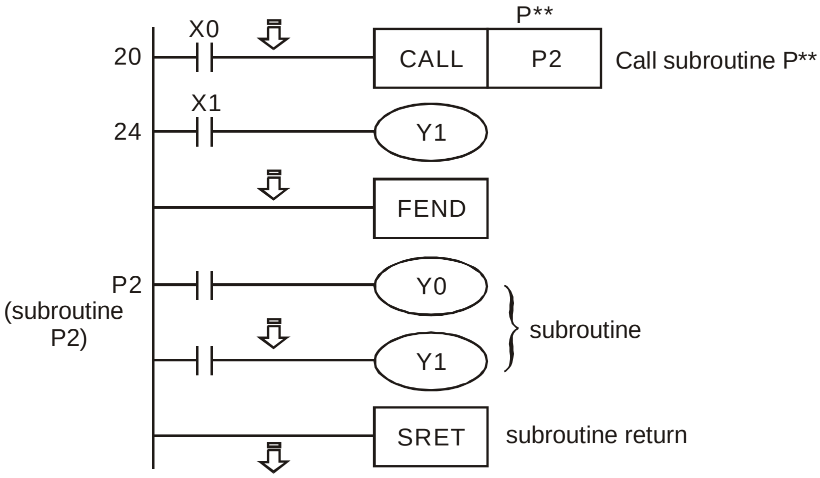
When X0 = ON, program will jump from address 0 to N (designated label P1) and keep on the execution. Instructions between 0 and N will be ignored.

When X0 = OFF, program will execute from 0 and keep on executing the followings. CJ instruction won't be executed at this time.



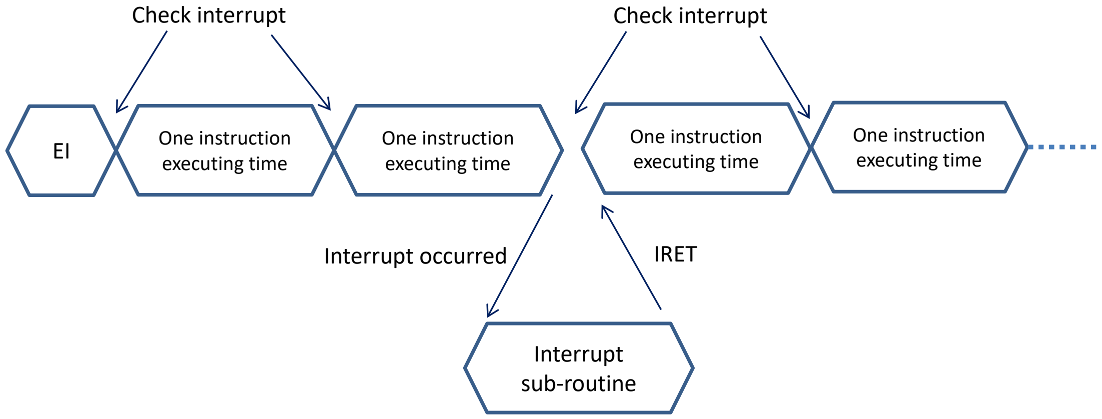
CALL subroutine, SRET subroutine END:

When X0 is ON, program will jump to P2 to execute the designated subroutine. When SRET instruction is executed, it returns to address 24 to go on executing.



Interrupt pointer I: used with application instruction API 04 EI, API 05 DI, API 03 IRET. There are four types of interruption pointers. To insert an interruption, users need to combine EI (enable interruption), DI (disable interruption) and IRET (interruption return) instructions.

When the instruction EI is enabled, the PLC will check if there is any interrupt that need to be done everytime an instruction is executed. If an interrupt is found, the PLC will stop executing the instruction and execute the interrupt first. If no interrupt is found, the PLC will keep executing the instruction. From the explanation, we can conclude that the maximum waiting time for an interrupt to be executed is the execution time of an instruction.



1. External interrupt
   - When input signal of input terminal X0~X7 is triggered on rising-edge or falling-edge, it will interrupt current program execution and jump to the designated interrupt subroutine pointer I000/I001(X0), I100/I101(X1), I200/I201(X2), I300/I301(X3), I400/I401(X4), I500/I501(X5), I600/I601(X6), I700/I701(X7). When IRET instruction is executed, program execution returns to the address before interrupt occurs.
   - When X0 (C243) works with I100/I101 (X1), X0/X1 (C246, C248, C252) works with I400/I401, the value of C243, C246, C248, C252 will be stored in (D1240, D1241)
   - When X2 (C244) works with I300/I301 (X3), X2/X3 (C250, C254) works with I500/I501, the value of C244, C250, C254 will be stored in (D1242, D1243).
2. Timer interrupt
   PLC automatically interrupts the currently executed program every a fixed period of time (2ms~99ms or 0.5ms~9.9ms) and jumps to the execution of a designated interruption subroutine
3. Counter interrupt
   The high-speed counter comparison instruction API 53 DHSCS can designate that when the comparison reaches the target, the currently executed program will be interrupted and jump to the designated interruption subrountine executing the interruption pointers I010, I020, I030, I040, I050 ,I060, I070, I080.

4. Communication interrupt
   I140:
   Communication instruction RS (COM1 RS-232) can be designated to send interrupt request when specific charcters are received. Interrupt I140 and specific characters is set to low byte of D1167.
   This function can be adopted when the PLC receives data of different length during the communication. Set up the specific end word in D1167 and write the interruption subroutine I140. When PLC receives the end word, the program will execute I140.
   I150:
   Communication instruction RS (COM2 RS-485) can be designated to send interrupt request when specific charcters are received. Interrupt I150 and specific characters is set to low byte of D1168.
   This function can be adopted when the PLC receives data of different length during the communication. Set up the specific end word in D1168 and write the interruption subroutine I150. When PLC receives the end word, the program will execute I150..
   I160:
   Communication instruction RS (COM3 RS-485) can be designated to send interrupt request when specific charcters are received. Interrupt I160 and specific characters is set to low byte of D1169
   This function can be adopted when the PLC receives data of different length during the communication. Set up the specific end word in D1169 and write the interruption subroutine I160. When PLC receives the end word, the program will execute I160

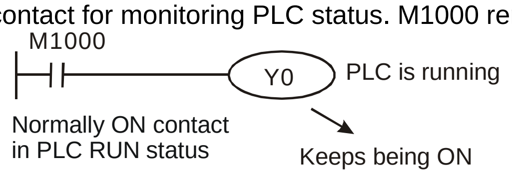
2.16 Applications of Special M Relays and D Registers

Function Group PLC Operation Flag
Number M1000~M1003
Contents:

These relays provide information of PLC operation in RUN status.

M1000:

NO contact for monitoring PLC status. M1000 remains "ON" when PLC is running.



M1001:

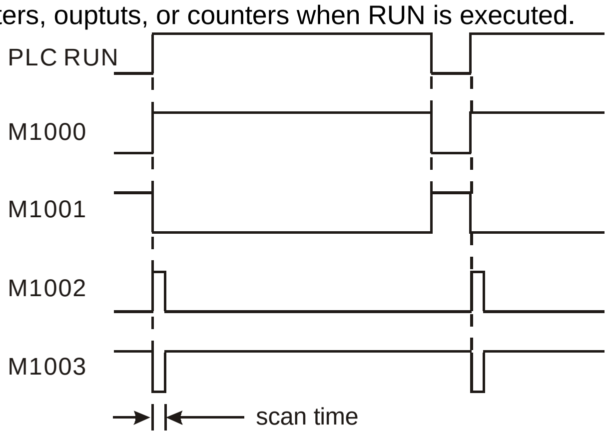
NC contact for monitoring PLC status. M1001 remains "OFF" when PLC is running.

M1002:

Enables single positive pulse for the first scan when PLC RUN is activated. Used to initialize registers, ouptuts, or counters when RUN is executed..

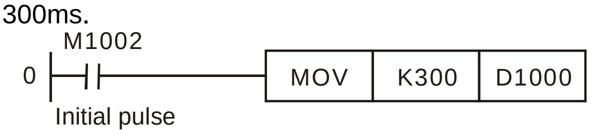
M1003:

Enables single negative pulse for the first scan when PLC RUN is activated. Used to initialize registers, ouptuts, or counters when RUN is executed.



Function Group Watchdog Timer (WDT)
Number D1000
Contents:

1. Monitor timer is used for moitoring PLC scan time. When the scan time exceeds the set value (SV) in the monitor timer, the red ERROR LED will be ON and all outputs will be "OFF".
2. The default in the monitor timer is 200ms. If the program is long or the operation is too complicated, MOV instruction can be used to modify SV. See the example below for SV = 300ms.



3. The maximum SV in the monitor timer is 32,767ms. However, care should be taken when adjusting SV. If SV in D1000 is too big, it cost much longer for operation errors to be detected. Therefore, SV is suggested to be shorter than 200ms.
4. Scan time could be prolonged due to complicated instruction operations or too many I/O modules being connected. Check D1010 ~ D1012 to see if the scan time exceeds the SV in D1000. Besides modifying the SV in D1000, users can also apply WDT instruction (API 07). When program execution progresses to WDT instruction, the internal monitor timer will be reset and therefore the scan time will not exceed the set value in the monitor timer.

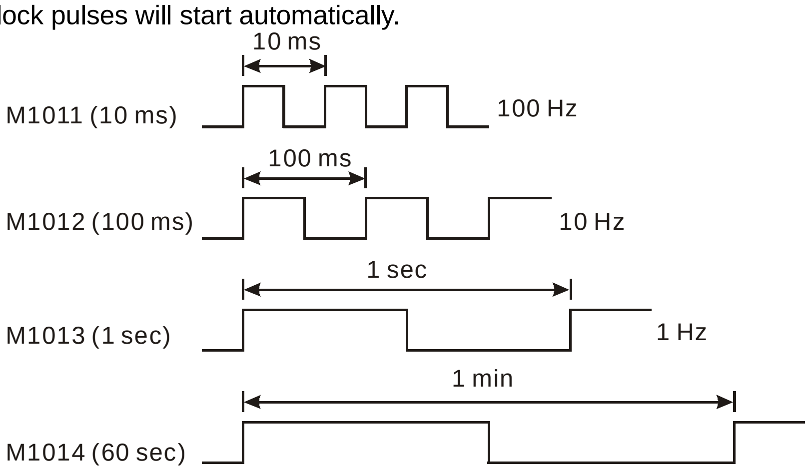
Function Group Program Capacity
Number D1002
Contents:
This register holds the program capacity of the PLC.
SS2: 7,920 steps (Word)
ES2 / EX2 / EC5 / SA2 / SX2 / SE series: 15,872 steps (Word)

Function Group Syntax Check
Number M1004, D1004, D1137
Contents:
1. When errors occur in syntax check, ERROR LED indicator will flash and special relay M1004 = ON.
2. Timings for PLC syntax check:
   a) When the power goes from “OFF” to “ON”.
   b) When WPLSoft writes the program into PLC.
   c) When on-line editing is being conducted on WPLSoft.
3. Errors might result from parameter error or grammar error. The error code of the error will be placed in D1004. The address where the fault is located is saved in D1137. If the error belongs to loop error it may not have an address associated with it. In this case the value in D1137 is invalid.
4. For syntax error codes pease refer to section 6.2 Error Code table.

Function Group Watchdog Timer
Number M1008, D1008
Contents:
1. When the scan is time-out during execution, ERROR LED will be ON and M1008 = ON.
2. D1008 saves the STEP address where the timeout occurred

Function Group Scan Time Monitor
Number D1010~D1012
Contents:
The present value, minimum value and maximum value of scan time are stored in D1010 ~ D1012.
D1010: current scan time
D1011: minimum scan time
D1012: maximum scan time

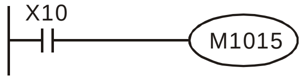
Function Group Internal Clock Pulse
Number M1011~M1014
Contents:
1. PLC provides four different clock pulses to aid the application. When PLC is power-on, the four clock pulses will start automatically.



2. Clock pulse works even when PLC stops, i.e. activation of clock pulse is not synchronized with PLC RUN execution.

Function Group High-speed Timer
Number M1015, D1015
Contents:
1. When M1015 = ON, high-speed timer D1015 will be activated when the current scan proceeds to END instruction. The minimum resolution of D1015 is 100us.
2. The range of D1015 is 0~32,767. When it counts to 32,767, it will start from 0 again.
3. When M1015 = OFF, D1015 will stop timing immediately.

Example:
1. When X10 = ON, M1015 = ON to start high-speed timer and record the present value in D1015.
2. When X10 = OFF, M1015 = OFF. High-speed timer is disabled.



Function Group M1016~M1017, M1077, D1313~D1319
Number Real Time Clock
Contents:
1. Special M and special D relevant to RTC

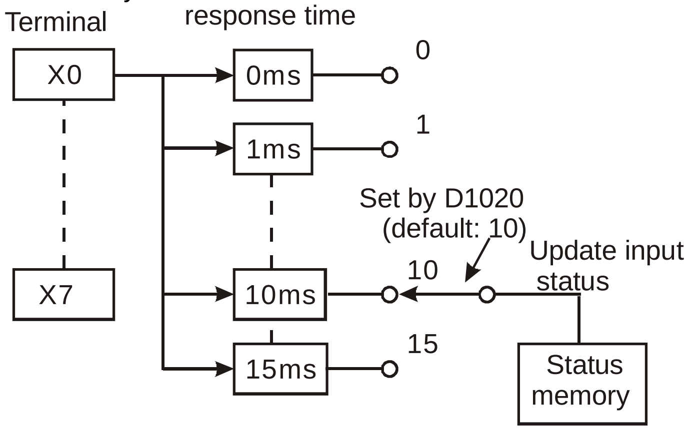
Device	Name	Function
M1016	Year Display	OFF: display the last 2 digits of year in A.D ON: display the last 2 digits of year in A.D. plus 2,000
M1017	±30 seconds correction	When triggered from “Off” to “On”, the correction is enabled. 0 ~ 29 second: minute intact; second reset to 0 30~ 59 second: mimute + 1; second reset to 0
M1077	Real-time clock	Real-time clock is out of order.
D1313	Second	0~59
D1314	Minute	0~59
D1315	Hour	0~23
D1316	Day	1~31
D1317	Month	1~12
D1318	Week	1~7
D1319	Year	0 ~ 99 (last 2 digits of Year in A.D.)

2. If set value for RTC is invalid. RTC will display the time as Second→0, Minute→0, Hour→0, Day→1, Month→1, Week→1, Year→0.
3. Only when power is on can RTCs of SS2 series perform the fuction of timing. Memory of RTC is latched. RTC will resume the time when power is down. For higher accuracy of RTC, please conduction calibratoin on RTC when power resumes.
4. RTCs of SA2/SE V1.0 and ES2/EX2/SX2 V2.0 series can still operate for one or two weeks after the power is off (they vary with the ambient temperature). Therefore, if the machine has not operated since one or two weeks ago, please reset RTC.
5. Methods of modifying RTC:
   a) Apply TWR instruction to modify the built-in real time clock. Please refer to TWR instruction for detail.
   b) Use peripheral devices or WPLSoft to set the RTC value.

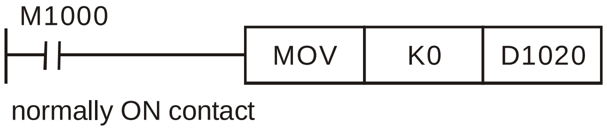
Function Group π (PI)
Number D1018~D1019
Contents:
1. D1018 and D1019 are combined as 32-bit data register for storing the floating point value ofπ
2. Floating point value = H 40490FDB

Function Group Adjustment on Input Terminal Response Time
Number D1020, D1021
Contents:
1. D1020 can be used for setting up the response time of receiving pulses at X0 ~X7 for ES2 series MPU. Default: 10ms, 0~20ms adjustable.
2. D1021 can be used for setting up the response time of receiving pulses at X10~X17 X7 for 28SS2 V3.42 /28SA2 V3.0 /26SE with firmware V1.0 or later versions. Default: 10ms, 0~20ms adjustable.

3. When the power of PLC goes from “OFF” to “ON”, the content of D1020 is set to 10 automatically.



4. If the following programs are executed, the response time of X0 ~ X7 will be set to 0ms. However, the fastest response time of input terminals will be 50μs due to that all terminals are connected with RC filters..



5. It is not necessary to adjust response time when using high-speed counters or interrupts
6. Using API 51 REFF instruction has the same effect as modifying D1020 and D1021.

Function Group X6 pulse width detecting function
Number M1083,M1084, D1023
Contents:
When M1084 = ON, X6 pulse width detecting function is enabled and the detected pulse width is stored in D1023 (unit: 0.1ms)
M1083 On：detecting width of negative half cycle (OFF→ON)
M1083 Off：detecting width of positive half cycle (ON→OFF)

Function Group Communication Error Code
Number M1025, D1025
Contents:
In the connection between PLC and PC/HMI, M1025 will be ON when PLC receives illegal communication request during the data transmission process. The error code will be stored in D1025.
01: illegal instruction code
02: illegal device address.
03: requested data exceeds the range.
07: checksum error

Function Group Pulse output Mark and Mask function
Number M1108, M1110, M1156, M1157, M1158, M1538, M1159, M1540, D1026, D1027, D1135, D1136, D1232, D1233, D1234, D1235, D1348, D1349
Contents:
Please refer to explanations of API 59 PLSR / API 158 DDRVI / API 197 DCLLM instructions.

Function Group Execution Completed Flag
Number M1029, M1030, M1102, M1103
Contents:
Execution Completed Flag:
MTR, HKY, DSW, SEGL, PR:
M1029 = ON for a scan cycle whenever the above instructions complete the execution.
PLSY, PLSR:

1. M1029 = ON when Y0 pulse output completes.
2. M1030 = ON when Y1 pulse output completes
3. M1102 = ON when Y2 pulse output completes.
4. M1103 = ON when Y3 pulse output completes.
5. When PLSY, PLSR instruction are OFF, M1029, M1030, M1102, M1103 will be OFF as well. When pulse output instructions executes again, M1029, M1030, M1102, M1103 will be OFF and turn ON when execution completes.
6. Users have to clear M1029 and M1030 manually.

INCD:

M1029 will be "ON" for a scan period when the assigned groups of data comparison is completed

RAMP, SORT:

1. M1029= ON when instruction is completed. M1029 must be cleared by user manually.
2. If this instruction is OFF, M1029 will be OFF.

DABSR:

1. M1029= ON when instruction is completed.
2. When the instruction is re-executed for the next time, M1029 will turn off first then ON again when the instruction is completed

ZRN, DRVI, DRVA:

1. M1029 will be "ON" after Y0 and Y1 pulse output is completed. M1102 will be "ON" after Y2 and Y3 pulse output is compeleted.
2. When the instruction is re-executed for the next time, M1029 / M1102 will turn off first then ON again when the instruction is completed.

2

Function Group Clear Instruction

Number M1031, M1032

Contents:

M1031 (clear non-latched memory), M1032 (clear latched memory)

Device	Devices will be cleared
M1031 Clear non-latched area	Contact status of Y, general-purpose M and general-purpose S ■ General-purpose contact and timing coil of T ■ General-purpose contact, counting coil reset coil of C ■ General-purpose present value register of D ■ General-purpose present value register of T ■ General-purpose present value register of C
M1032 Clear latched area	Contact status of M and S for latched ■ Contact and timing coil of accumulative timer T ■ Contact and timing coil of high-speed counter C for latched ■ Present value register of D for latched ■ Present value register of accumulative timer T ■ Present value register of high-speed counter C for latched

Function Group Output State Latched in STOP mode

Number M1033

Contents:

When M1033 = ON, PLC outputs will be latched when PLC is switched from RUN to STOP. If the data is in non-latched area, it cannot be cleared to zero.

Function Group Disabling all Y outputs

Number M1034

Contents:

When M1034 = ON, all outputs will turn off.

Function Group RUN/STOP Switch

Number M1035

Contents:

When M1035 = ON, PLC uses input point X7 as the switch of RUN/STOP.

Function Group COM Port Function

Number

Item \ Port	COM1	COM2	COM3
Communication format	D1036	D1120	D1109
Communication setting holding	M1138	M1120	M1136
ASCII/RTU mode	M1139	M1143	M1320
Slave communication address	D1121		D1255

Contents:
COM ports (COM1: RS-232, COM2: RS-485, COM3: RS-485) support communication format of MODBUS ASCII/RTU modes. When RTU format is selected, the data length should be set as 8. COM2 and COM3 support transmission speed up to 921kbps. COM1, COM2 and COM3 can be used at the same time.

COM1:
Can be used in master or slave mode. Supports ASCII/RTU communication format, baudrate (115200bps max), and modification on data length (data bits, parity bits, stop bits). **D1036:** COM1 (RS-232) communication protocol of master/slave PLC. (b8 - b15 are not used) Please refer to table below for setting.

COM2:
Can be used in master or slave mode. Supports ASCII/RTU communication format, baudrate (921kbps max), and modification on data length (data bits, parity bits, stop bits). **D1120:** COM2 (RS-485) communication protocol of master/slave PLC. Please refer to table below for setting.
Note: 600 bps is not available for ES2-C: V4.0 or later; 921kbps is not available for SE: V1.xx; 921kbps is available for SE: V2.00 or later.

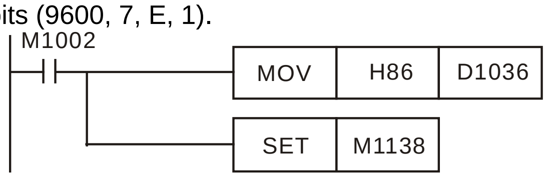
COM3:
Can be used in master or slave mode. Supports ASCII/RTU communication format, baudrate (921kbps max), and modification on data length (data bits, parity bits, stop bits). **D1109:** COM3 (RS-485) communication protocol of master/slave PLC. (b8 - b15 are not used) Please refer to table below for setting.
Note: ES2-C does NOT come with COM3; 921kbps is not available for SE: V1.xx; 921kbps is available for SE: V2.00 or later.

	Content		
b0	Data Length	0: 7 data bits, 1: 8 data bits (RTU supports 8 data bits only)	
b1 b2	Parity bit	00: None 01: Odd 11: Even	
b3	Stop bits	0: 1 bit, 1: 2bits	
b4 b5 b6 b7	Baud rate	0100(H4): 600 0101(H5): 1200 0110(H6): 2400 0111(H7): 4800 1000(H8): 9600 1001(H9): 19200 1010(HA): 38400 1011(HB): 57600 1100(HC): 115200	
		1101(HD): 500000 (COM2 / COM3)	
		1110(HE): 31250 (COM2 / COM3)	
		1111(HF): 921000 (COM2 / COM3)	
b8	Select start bit	0: None	1: D1124

	Content		
b9	Select the 1st end bit	0: None	1: D1125
b10	Select the 2nd end bit	0: None	1: D1126
b11~b15	Undefined		

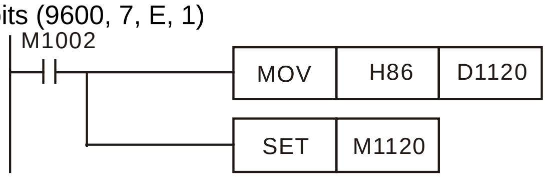
Example 1: Modifying COM1 communication format

1. Add the below instructions on top of the program to modify the communication format of COM1. When PLC switches from STOP to RUN, the program will detect whether M1138 is ON in the first scan. If M1138 is ON, the program will modify the communication settings of COM1 according to the value set in D1036
2. Modify COM1 communication format to ASCII mode, 9600bps, 7 data bits, even parity, 1 stop bits (9600, 7, E, 1).



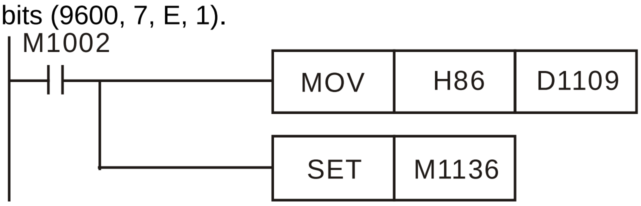
Example 2: Modiying COM2 communication format

1. Add the below instructions on top of the program to modify the communication format of COM2. When PLC switches from STOP to RUN, the program will detect whether M1120 is ON in the first scan. If M1120 is ON, the program will modify the communication settings of COM2 according to the value set in D1120
2. Modify COM2 communication format to ASCII mode, 9600bps, 7 data bits, even parity, 1 stop bits (9600, 7, E, 1)



Example 3: Modifying COM3 communication format

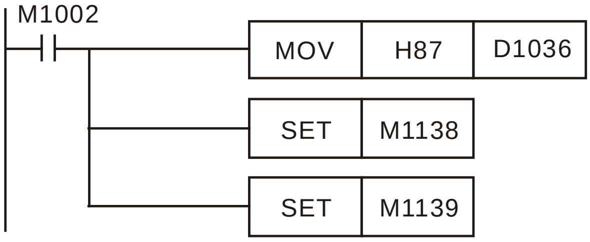
1. Add the below instructions on top of the program to modify the communication format of COM3. When PLC switches from STOP to RUN, the program will detect whether M1136 is ON in the first scan. If M1136 is ON, the program will modify the communication settings of COM3 according to the value set in D1109
2. Modify COM3 communication format to ASCII mode, 9600bps, 7 data bits, even parity, 1 stop bits (9600, 7, E, 1).



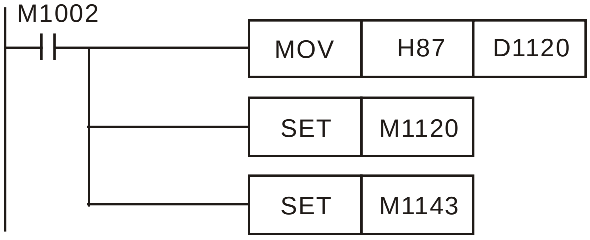
Example 4: RTU mode setting of COM1、COM2、COM3

1. COM1, COM2 and COM3 support ASCII/RTU mode. COM1 is set by M1139, COM2 is set by M1143 and COM3 is set by M1320. Set the flags ON to enable RTU mode or OFF to enable ASCII mode.
2. Modify COM1/COM2/COM3 communication format to RTU mode, 9600bps, 8 data bits, even parity, 1 stop bits (9600, 8, E, 1).

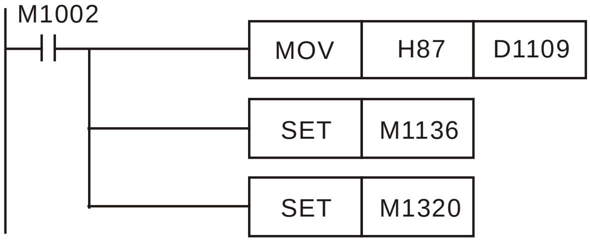
COM1:



COM2:



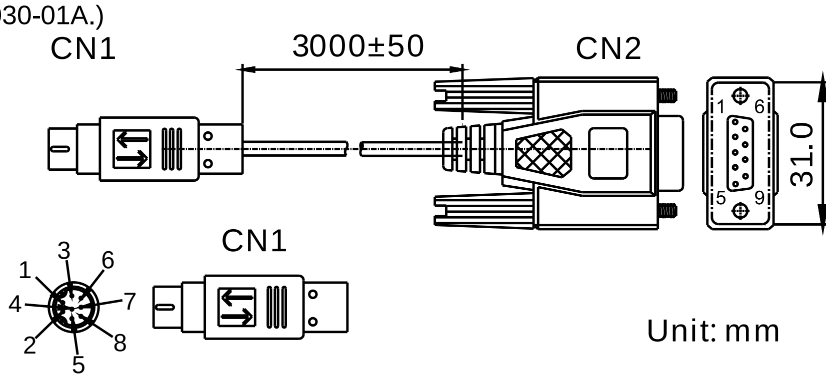
COM3:

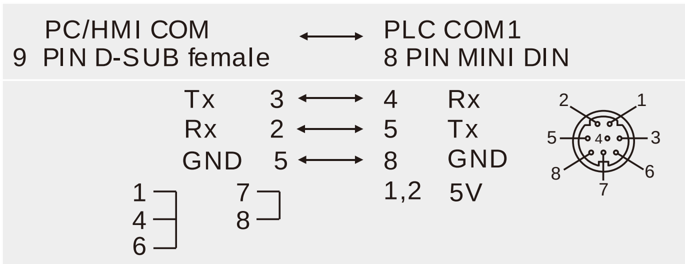


Note:

1. The modified communication format will not be changed when PLC state turns from RUN to STOP.
2. If the PLC is powered OFF then ON again in STOP status, the modified communication format on COM1~COM3 will be reset to default communication format (9600, 7, E, 1).

Definitions of the pins in COM1: (It is suggested that users should use the Delta communication cable UC-MS030-01A.)





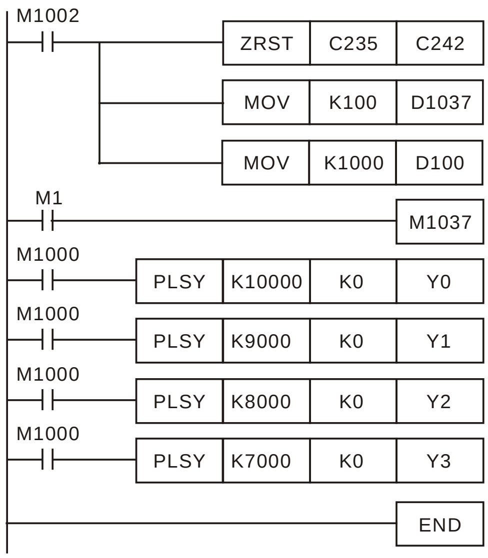
Function Group Enable SPD function
Number M1037, D1037
Contents:

1. M1037 and D1037 can be used to enable 8 sets of SPD instructions. When M1037 is ON, 8 sets of SPD instructions will be enabled. When M1037 is OFF, the function will be disabled.

2. Availability:

Model	ES2/EX2	ES2-C	ES2-E	12SA2	SX2	SS2	12SE	26SE	28SA2
Firmware version	V1.42	V3.00	V1.08	V2.60	V2.40	V3.28	V2.02	V1.00	V2.90

3. The detected speed will be stored in the registers designated by D1037, e.g. if D1037 = K100, the user has to set up the value in D100, indicating the interval for capturing the speed value (unit: ms). In addition, the captured speed value will be stored in D101 ~ D108 in order.
   ※ When the function is enabled, C235~C242 will be occupied and unavailable in PLC execution process program.



Function Group Communication Response Delay
Number D1038
Contents:

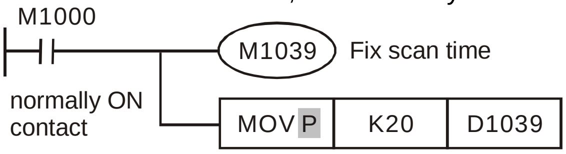
1. Data response delay time can be set when PLC is a Slave in COM2, COM3 RS-485 communication. Unit: 0.1ms. 0~10,000 adjustable.
2. For the following models and firmware versions, the setting range is 0-20.

Model	ES2/EX2 /ES2-C	EC5	ES2-E	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware version	V3.64	V1.00	V1.40	V3.00	V3.60	V1.86	V1.40	V3.00

3. By using PLC-Link, D1038 can be set to send next communication data with delay. Unit: 1 scan cycle. 0~10,000 adjustable

Function Group Fixed scan time
Number M1039, D1039
Contents:

1. When M1039 is ON, program scan time is determined by D1039. When program execution is completed, next scan will be activated only when the fixed scan time is reached. If D1039 is less than actual scan time, it will scan by the actual program scan time.



Scan time is fixed to 20ms

2. Instructions related to scan time, RAMP, HKY, SEGL, ARWS and PR should be used with "fixed scan time" or "timed interrupt".
3. Particularly for instruction HKY, which is applied for 16-keys input operated by 4x4 matrix, scan time should be set to 20ms or above.
4. Scan time displayed in D1010~D1012 also includes fixed scan time.

Function Group Analog Function built in the PLC
Number D1062, D1110~D1118
Contents:

1. The function is for EX2/SX2 Only
2. Resolution of AD (analog input) channels: 12 bits for 20EX2 and 20SX2; 16 bits for the voltage/current mode of 30EX2; 0.1 ℃ for the temperature mode of 30EX2
3. The analog input signals and their corresponding digital values:

Mode \ Model		20EX2/SX2	30EX2
Voltage	-10 V~+10 V	-2000~+2000	-32000~+32000
	-5 V~+5 V	Not support	-32000~+32000
	+1 V~+5 V	Not support	+0~+32000
Current	-20 mA~+20 mA	-2000~+2000	-32000~+32000
	+4 mA~+20 mA	+0~+2000	+0~+32000
Temperature	PT100/PT1000 -180 ℃~+800 ℃	Not support	-1800~+8000
	NI100/NI1000 -80 ℃ ~ +170 ℃	Not support	-800~+1700

4. Resolution of DA (analog output) channels: 12 bits

5. The analog output signals and their corresponding digital values:

Mode \ Model		20EX2/SX2	30EX2
Voltage	-10 V~+10 V	-2000~+2000	-32000~+32000
Current	+0 mA~+20 mA	+0~+4000	+0~+32000
	+4 mA~+20 mA	+0~+4000	+0~+32000

6. The descriptions of the special data registers for the analog functions:

Device	Function
D1062	Average number of times analog input signals are input through CH0~CH3 of 20EX2/SX2: 1~20, Default = K2
	Average number of times analog input signals are input through CH0~CH2 of 30EX2: 1~15, Default = K2
D1110	Average value of EX2/SX2 analog input channel 0 (AD 0)
D1111	Average value of EX2/SX2 analog input channel 1 (AD 1)
D1112	Average value of EX2/SX2 analog input channel 2 (AD 2)
D1113	Average value of 20EX2/SX2 analog input channel 3 (AD 3)

Device	Function
	If D1062 is ON, the average value is the current value.
	Displaying the status of the analog input channel of 30EX2 Please see the explanation below for more information.
D1114	Enable/disable 20EX2/SX2 AD channels (0: enable (default) / 1: disable) bit0~bit3 sets AD0~AD3.
	30EX2 does not support this function.
D1116	Output value of analog output channel 0 (DA 0) of EX2/SX2
D1117	Output value of analog output channel 1 (DA 1) of 20EX2/SX2
	30EX2 does not support this function.
D1118	For EX2/SX2 series, sampling time of analog/digital conversion. Sampling time will be regarded as 2ms If D1118≦2.

The description of D1113 for 30EX2:

Bit15~12	Bit11~8	Bit7~4	Bit3~0
Reserved	Status of the analog input channel (AD2)	Status of the analog input channel (AD1)	Status of the analog input channel (AD0)

The status of the analog input channel of 30EX2:

Status	0x0	0x1	0x2
Description	Normal	The analog input exceeds the upper/lower limit.	The temperature sensor is disconnected.

The upper/lower limit values for the analog input mode of 30EX2:

Analog input mode		Upper limit value	Lower limit value
Voltage	-10~+10 V	+32384	-32384
	-5V~+5 V		
	+1 V~+5 V	+32384	-384
Current	-20 mA~+20 mA	+32384	-32384
	+4 mA~+20 mA	+32384	-384
Temperature	PT100/PT1000	+8100	-1900
	NI100/NI1000	+1800	-900

Device number	Function
D1115	20EX2/SX2 analog input/output mode setting (Default=H'0) bit0~bit5: Selection between the voltage/current mode (0: Voltage; 1: Current; Default: Voltage) bit0~bit3: Analog inputs (AD0~AD3) bit4~bit5: Analog outputs (DA0~DA1) bit8~bit 13: Current mode bit8~bit11: AD0~AD3 (0: -20 mA~20 mA; 1: 4~20 mA) bit12~bit13: DA0~DA1 (0: 0~20 mA; 1: 4~20 mA)
	30EX2 analog input/output mode setting (Default=H'FFFF)

The description of D1115 for 30EX2:

Bit15~12	Bit11~8	Bit7~4	Bit3~0
Analog output mode of DA0	Analog input mode of AD2	Analog input mode of AD1	Analog input mode of AD0

The analog input modes for 30EX2:

Code	0x0	0x1	0x2	0x3
Description	Two-wire PT100	Three-wire NI100	Two-wire PT1000	Two-wire NI1000
Code	0x4	0x5	0x6	0x7
Description	Three-wire	Three-wire	Three-wire	Three-wire

	PT100	NI100	PT1000	NI1000
Code	0x8	0x9	0xA	0xB
Description	Voltage: -10 V~+10 V	Voltage: -5 V~+5 V	Voltage: +1 V~+5 V	Current: -20 mA~+20 mA
Code	0xC	0xD	0xE	0xF
Description	Current: +4 mA~+20 mA	Reserved		Unused

The analog output modes for 30EX2:

Code	0x0	0x1	0x2	0xF
Description	Voltage: -10 V~+10 V	Current: +0 mA~+20 mA	Current: +4 mA~+20 mA	Unused

The example of setting D1115 for 30EX2:
If the analog input mode of AD0 is the two-wire NI100, the analog input mode of AD1 is the three-wire 1000, the analog input mode of AD2 is the voltage mode (+1 V~ +5 V), and the analog output mode of DA0 is the current mode (+4 mA ~ +20 mA), the setting value in D1115 is H'2A61.

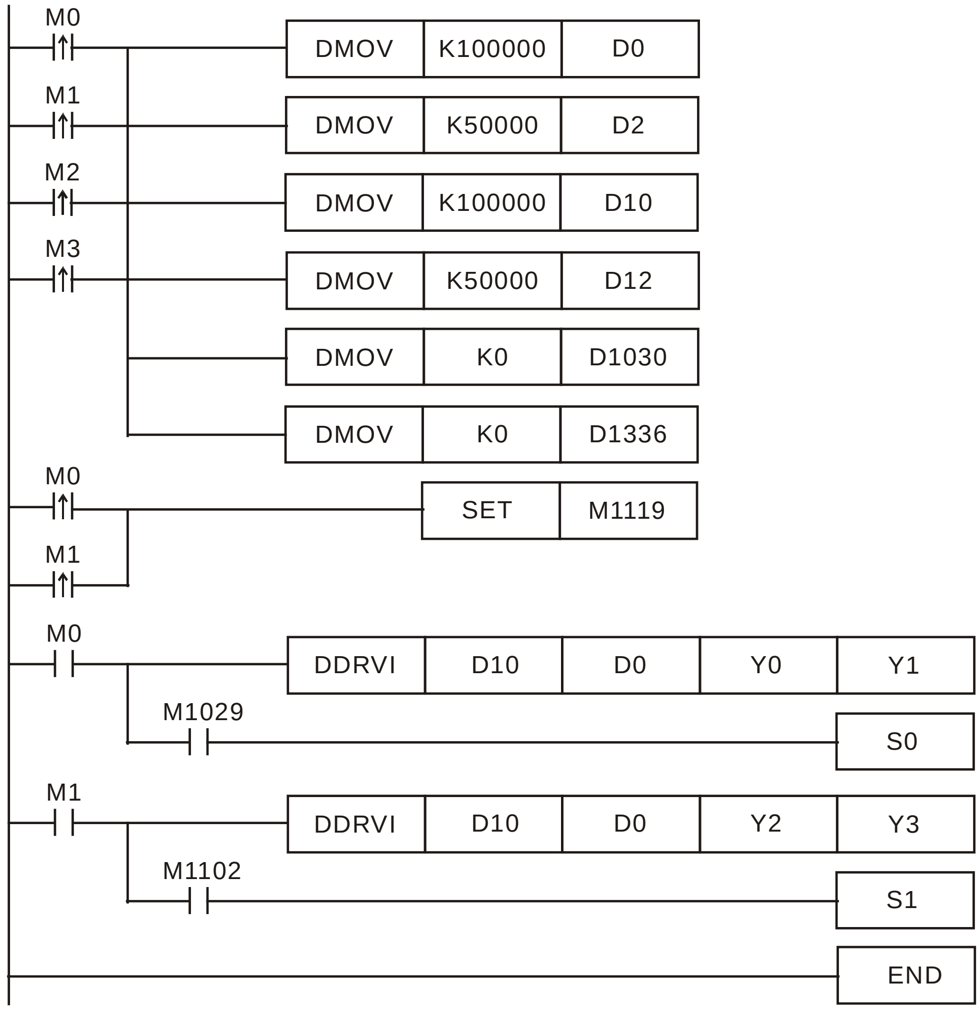
Function Group Enable 2-speed output function of DDRVI/DDRVA instruction
Number M1119
Contents:
Availability:

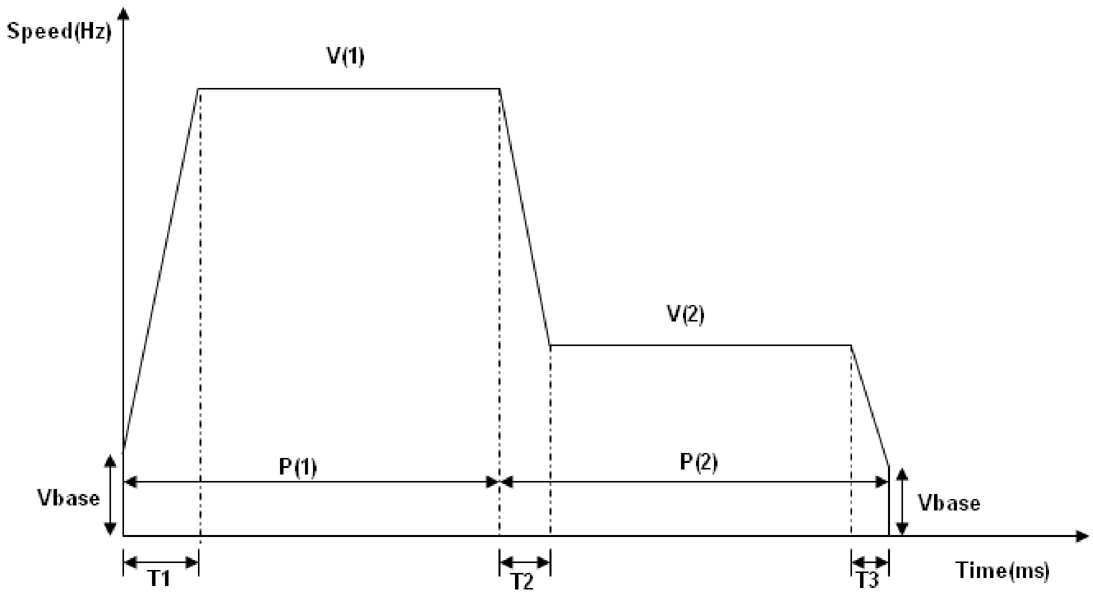
Models / Function	ES2/EX2	ES2-C	ES2-E	12SA2/ SX2	SS2	12SE	26SE	28SA2
DDRVI	V1.42	V1.42	V1.08	V1.40	V2.2	V2.02	V1.0	V2.88
DDRVA	V2.8	V2.8	V1.08	SA2_V2.42 SX2_V2.20	V3.60	V2.02	V1.0	V2.88

You need to set up M1119 before executing DDRVI/DDRVA instruction. After the execution is complete, M1119 will be cleared.

When P (1) is set to 100K and P (2) is set to 200K, the actions after the execution of DDRVI/DDRVA will be as below.
DDRVI: sending out 100K of pulses in the first section and sending out 200K pulses at the second section; a total of 300K pulses will be sent.
DDRVA: sending out 100K of pulses in the first section and sending out 100K pulses at the second section; a total of 200K pulses will be sent.
(The positions of the first and second sections should be in the same direction.)

Example:
When M1119 is ON, 2-speed output function of DDRVI/DDRVA will be enabled.
Assume that D0 (D1) is the first speed and D2(D3) is the second speed. D10(D11) is the output pulse number of the first speed and D12(D13) is the output pulse number of the second speed.





Vbase	T1	T2+T3	P(1)	V(1)	P(2)	V(2)
Initial frequency	Ramp-up time	Ramp-down time	Position of the first speed	The first speed	Position of the second speed	The second speed

Function Group Program Execution Error
Number M1067~M1068, D1067~D1068
Contents:

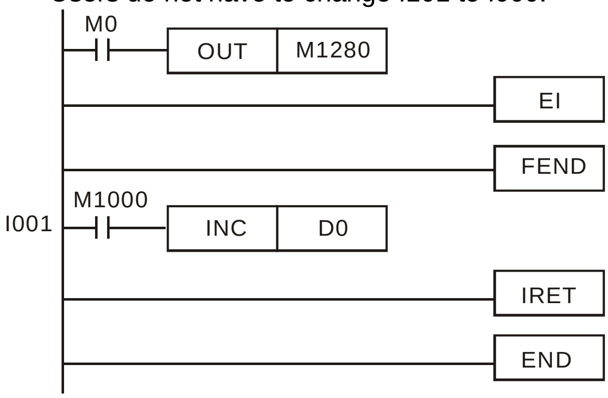
Device	Explanation	Latched	STOP→RUN	RUN→STOP
M1067	Program execution error	None	Clear	Unchanged
M1068	Execution error locked	None	Unchanged	Unchanged
D1067	Error code for program execution	None	Clear	Unchanged
D1068	Address of program execution error	None	Unchanged	Unchanged

Error code explanation:

D1067 error code	Function
0E18	BCD conversion error
0E19	Divisor is 0
0E1A	Use of device exceeds the range (including E, F index register modification)
0E1B	Square root value is negative
0E1C	FROM/TO instruction communication error

Function Group I/O Modules Detection
Number D1140, D1142, D1143, D1145
Contents:
D1140: Number of right-side modules (AIO, PT, TC, etc.), max. 8 modules can be connected.
D1142: Number of input points (X) on DIO modules.
D1143: Number of output points (Y) on DIO modules.
D1145: Number of left-side modules (AIO, PT, TC, etc.), max. 8 modules can be connected. (Only applicable for SA2/SX2/SE).

Function Group Reverse Interrupt Trigger Pulse Direction
Number M1280, M1284, M1286
Contents:
1. The falgs should be used with EI instruction and should be inserted before EI instruction
2. The default setting of interrupt I101 (X0) is rising-edge triggered. If M1280 is ON and EI instruction is executed, PLC will reverse the trigger direction as falling-edge triggered. The trigger pulse direction of X1 will be set as rising-edge again by resetting M1280.
3. When M0 = OFF, M1280 = OFF. X0 external interrupt will be triggered by rising-edge pulse.
4. When M0 = ON, M1280 = ON. X0 external interrupt will be triggered by falling-edge pulse. Users do not have to change I101 to I000.



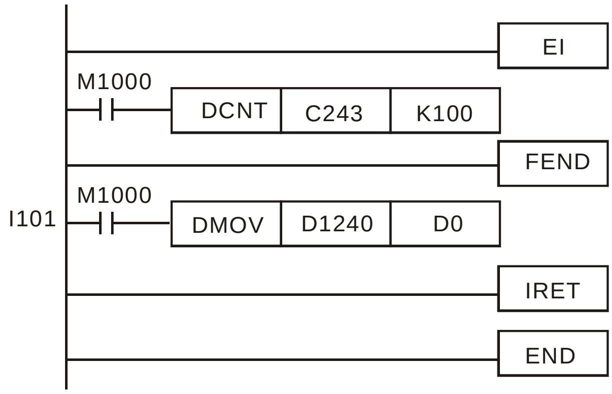
Function Group Stores Value of High-speed Counter when Interrupt Occurs
Number D1240~D1243
Contents:

1. If extertal interrupts are applied on input points for Reset, the interrupt instructions have the priority in using the input points. In addition, PLC will move the current data in the counters to the associated data registers below then reset the counters.

Special D	D1241, D1240				D1243, D1242		
Counter	C243	C246	C248	C252	C244	C250	C254
Interrupt signal	X1(I100/I101)	X4(I400/I401)			X3(I300/I301)	X5(I500/I501)	

2. Function:
   a) When X0 (counter input) and X1 (external Interrupt) correspondingly work together with C243, and I100/I101, PLC will move the count value to D1241 and D1240.
   b) When X0 (counter input) and X4 (external Interrupt) correspondingly work together with C246, C248, C252 and I400/I401, PLC will move the count value to D1241 and D1240
   c) When X2 (counter input) and X3 (external Interrupt) correspondingly work together with C244, and I300/I301, PLC will move the count value to D1243 and D1242.
   d) When X2 (counter input) and X5 (external Interrupt) correspondingly work together with C250, C254 and I500/I501, PLC will move the count value to D1243 and D1242.

Example:



When external interrupt (X1, I101) occurs during counting process of C243, the count value in C243 will be stored in (D1241, D1240) and C243 is reset. After this, the interrupt subroutine I101 will be executed

Function Group Enabling force-ON/OFF of input point X
Number M1304
Contents:
When M1304 = ON, WPLSoft or ISPSoft can set ON/OFF of input pont X, but the associated hardware LED will not respond to it.

Function Group Output specified pulses or seek Z phase signal when zero point is achieved.
Number M1308, D1312
Contents:
When zero point is achieved, PLC can output specified pulses or seek Z phase signal by this function. Input terminals X2, X3 are the Z-phase signal input point of CH1, CH2. When M1308= ON, D1312 is the setting register to specify the additional pulses within the range -30,000~30,000. Specified value exceeds the range will be changed as the max/min value automatically. When D1312 is set to 0, the additional pulses output function will be disabled.
Functions of other input terminals:

X4 → CH1 DOG signal input	X6 → CH2 DOG signal input
X5 → CH1 LSN signal input	X7 → CH2 LSN signal input

Function Group ID of right side modules on ES2/EX2/SS2/SA2/SX2/SE
Number D1320~ D1327
Contents:
When right side modules are connected on ES2/EX2, the ID of each I/O module will be stored in D1320~D1327 in connection order.
ID of each special module:

Name	ID (HEX)	Name	ID (HEX)
DVP04AD-E2	H'0080	DVP06PT-E2	H'00C2
DVP02DA-E2	H'0041	DVP04TC-E2	H'0083
DVP04DA-E2	H'0081	DVP02PU-E2	H'0045
DVP06XA-E2	H'00C4	DVP10RC-E2	H'0026
DVP04PT-E2	H'0082		

When right side modules are connected on SS2/SX2/SA2/SE, the ID of each I/O module will be stored in D1320~D1327 in connection order.
ID of each special module:

Name	ID (HEX)	Name	ID (HEX)	Name	ID (HEX)
DVP04AD-S	H'0088	DVP06PT-S	H'00CA	DVP02TUR-S	H'034F
DVP02DA-S	H'0049	DVP04TC-S	H'008B	DVP04AD-S2	H'0090
DVP04DA-S	H'0089	DVP01PU-S	H'0110	DVP04DA-S2	H'0091
DVP06AD-S	H'00C8	DVP08NTC-S	H'010D	DVP06X2-S2	H'00D4
DVP06XA-S	H'00CC	DVP02TUL-S	H'014F		
DVP04PT-S	H'008A	DVP02TUN-S	H'024F		

2

Function Group ID of left side modules on SA2/SX2/SE
Number D1386~D1393
Contents:
When left side modules are connected on SA2/SX2/SE, the ID of each I/O module will be stored in D1386~D1393 in connection order. (There is no communication port on left side of DVP28SA2 and DVP26SE.)

ID of each special module:

Name	ID (HEX)	Name	ID (HEX)	Name	ID (HEX)
DVP04AD-SL	H'4400	DVP01HC-SL	H'4120	DVP201LC-SL	H'5106
DVP04DA-SL	H'4401	DVP02HC-SL	H'4220	DVP202LC-SL	H'5206
DVP04PT-SL	H'4402	DVPDNET-SL	H'4131	DVP211LC-SL	H'5906
DVP04TC-SL	H'4403	DVPEN01-SL	H'4050	DVP01LC-SL	H'4106
DVP06XA-SL	H'6404	DVPMDM-SL	H'4040	DVP02LC-SL	H'4206
DVP01PU-SL	H'4110	DVPCOPM-SL	H'4133	DVP02PU-SL	H'4207

Function Group Mapping function of SA2/SX2/SE for left-side high-speed special modules
Number M1182, D9800~D9879
Contents:
The default value of M1182 in SA2 version 2.42/SX2 version 2.20 and below is Off. When M1182 is Off, the mapping function is enabled.
The default value of M1182 in SA2 version 2.60/SX2 version 2.40 and above/SE is On. When M1182 is On, the mapping function is disabled.
Example:
If the modules connected to SA2 from left to right are 04DA-SL and 04AD-SL, and M1182 is Off, D9810~D9813 will be assigned to 04DA-SL, and D9800~D9803 will be assigned to 04AD-SL.

Model name	04DA-SL	04AD-SL	SA2
Channel 1 (Ch1)	D9810	D9800	
Channel 2 (Ch2)	D9811	D9801	
Channel 3 (Ch3)	D9812	D9802	
Channel 4 (Ch4)	D9813	D9803	

Function Group Mapping function for right-side high-speed special modules
Number M1183, D9900 ~ D9979
Contents:
The default value of M1183 in ES2/EX2 is Off. When M1183 is Off, the mapping function is enabled.
The default value of M1183 in SA2/SX2/SS2/SE is On. When M1183 is On, the mapping function is disabled.
Example:
If the modules **connected** to ES2 from left to right are 04DA-E2, 04AD-E2 and 02PU-E2, and M1183 is Off, the assignment is shown as below.

Order from left to right	1	2	3
Model name	04DA-E2	04AD-E2	02PU-E2
Device number	D9900~ D9903	D9910~ D9913	D9920~ D9921
Action	Output value of DA channel 1-4	Input value of AD channel 1-4	D9920 = value in X point D9921 = axis state code

State code of DVP02PU-E2

State Code Byte #	Description	Axis 1-2
0	Error flag	Axis 1
1	The output is active.	
2	The output has stopped working.	
3	The instruction execution is complete.	
4	The postivie limit is reached.	
5	The negative limit is reached.	
6	Current position value overflow	
7	Pulse direction (positive or negative)	
8	Error flag	Axis 2
9	The output is active.	
10	The output has stopped working.	
11	The instruction execution is complete.	
12	The postivie limit is reached.	
13	The negative limit is reached.	
14	Current position value overflow	
15	Pulse direction (positive or negative)	

The corresponding error flag will be ON when the above mentioned incidents happened: 4/12, 5/13, 6/14. Once the error flag is ON, you need to use instruction to clear the shown error codes.

Function Group Output clear signals when ZRN is completed
Number M1346
Contents:
When M1346 = ON, PLC will output clear signals when ZRN is completed. The clear signals to Y0, Y1 will be sent by Y4 for 20ms, and the clear signals to Y2, Y3 will be sent by Y5 for 20ms.

Function Group PLC LINK
Number M1350-M1356, M1360-M1439, D1355-D1370, D1399, D1415-D1465, D1480-D1991
Contents:
1. PLC LINK supports COM2 (RS-485) with communication of up to 16 slaves and access of up to 50 words. (DVP-12SE V1.6 and DVP-26SE V2.0 can connect to up to 32 slaves, and read/write up to 100 words.)
2. Special D and special M corresponding to Connection ID1~ Connection ID8: (M1353 = OFF, access available for only 16 words)

MASTER PLC															
CONNECTION ID 1		CONNECTION ID 2		CONNECTION ID 3		CONNECTION ID 4		CONNECTION ID 5		CONNECTION ID 6		CONNECTION ID 7		CONNECTION ID 8	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
Special D registers for storing the read/written 16 data (Auto-assigned)															
D1480-D1495	D1496-D1511	D1512-D1527	D1528-D1543	D1544-D1559	D1560-D1575	D1576-D1591	D1592-D1607	D1608-D1623	D1624-D1639	D1640-D1655	D1656-D1671	D1672-D1687	D1688-D1703	D1704-D1719	D1720-D1735
Data length to be read/written for connections (Max 16 pieces of data; 0: no reading/writing)															
D1434	D1450	D1435	D1451	D1436	D1452	D1437	D1453	D1438	D1454	D1439	D1455	D1440	D1456	D1441	D1457
Starting address of the Connection to be read, written or connected.*															
D1355	D1415	D1356	D1416	D1357	D1417	D1358	D1418	D1359	D1419	D1360	D1420	D1361	D1421	D1362	D1422
M1355 = ON, Connection status is user-defined. Activate the Connection manually with M1360~M1367. M1355 = OFF, Connection status is auto-detected. Monitor if the connection is established and activated with M1360~M1367															
M1360		M1361		M1362		M1363		M1364		M1365		M1366		M1367	
Communication status of Connection ID#1-8 on PLC LINK (ON: communicating)															
M1376		M1377		M1378		M1379		M1380		M1381		M1382		M1383	
Error flag for errors occurred when reading and writing (ON = normal; OFF = error)															
M1392		M1393		M1394		M1395		M1396		M1397		M1398		M1399	
"Reading completed" flag (System resets to OFF after reading is complete on one module.)															
M1408		M1409		M1410		M1411		M1412		M1413		M1414		M1415	
"Writing completed" flag (System resets to OFF after writing is complete on one module.)															
M1424		M1425		M1426		M1427		M1428		M1429		M1430		M1431	

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

Slave PLC*															
CONNECTION ID 1		CONNECTION ID 2		CONNECTION ID 3		CONNECTION ID 4		CONNECTION ID 5		CONNECTION ID 6		CONNECTION ID 7		CONNECTION ID 8	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
D100-D115	D200-D215	D100-D115	D200-D215	D100-D115	D200-D215	D100-D115	D200-D215	D100-D115	D200-D215	D100-D115	D200-D215	D100-D115	D200-D215	D100-D115	D200-D215

3. Special D and special M corresponding to Connection ID9~ Connection ID16: (M1353 = OFF, access available for only 16 words)

MASTER PLC															
CONNECTION ID 9		CONNECTION ID 10		CONNECTION ID 11		CONNECTION ID 12		CONNECTION ID 13		CONNECTION ID 14		CONNECTION ID 15		CONNECTION ID 16	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
Special D registers for storing the read/written 16 pieces of data (Auto-assigned)															
D1736-D1751	D1752-D1767	D1768-D1783	D1784-D1799	D1800-D1815	D1816-D1831	D1832-D1847	D1848-D1863	D1864-D1879	D1880-D1895	D1896-D1911	D1912-D1927	D1928-D1943	D1944-D1959	D1960-D1975	D1976-D1991
Data length to be read/written for connections (Max 16 pieces of data; 0: no reading/writing)															
D1442	D1458	D1443	D1459	D1444	D1460	D1445	D1461	D1446	D1462	D1447	D1463	D1448	D1464	D1449	D1465

Starting address of the Connection to be read, written or connected.*															
D1363	D1423	D1364	D1424	D1365	D1425	D1366	D1426	D1367	D1427	D1368	D1428	D1369	D1429	D1370	D1430
M1355 = ON, Connection status is user-defined. Activate the Connection manually with M1368~M1375. M1355 = OFF, Connection status is auto-detected. Monitor if the connection is established and activated with M1368~M1375.															
M1368		M1369		M1370		M1371		M1372		M1373		M1374		M1375	
Communication status of Connection ID#9-16 on PLC LINK (ON: communicating)															
M1384		M1385		M1386		M1387		M1388		M1389		M1390		M1391	
Error flag for errors occurred when reading or writing on Connection ID#9-16(ON = normal; OFF = error)															
M1400		M1401		M1402		M1403		M1404		M1405		M1406		M1407	
Reading complete on PLC LINK Connection ID#9-16 (System resets to OFF after reading is complete on one module.)															
M1416		M1417		M1418		M1419		M1420		M1421		M1422		M1423	
Writing complete on PLC LINK connection ID#9-16 (System resets to OFF after writing is complete on one module.)															
M1432		M1433		M1434		M1435		M1436		M1437		M1438		M1439	

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

Slave PLC*															
CONNECTION ID 9		CONNECTION ID 10		CONNECTION ID 11		CONNECTION ID 12		CONNECTION ID 13		CONNECTION ID 14		CONNECTION ID 15		CONNECTION ID 16	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215

4. Special D and special M corresponding to Connection ID1~ID8: (M1353 = ON, access available for up to 50 words) (DVP-12SE V1.6 and DVP-26SE V2.0 supports 100 words at most.)

MASTER PLC															
CONNECTION ID 1		CONNECTION ID 2		CONNECTION ID 3		CONNECTION ID 4		CONNECTION ID 5		CONNECTION ID 6		CONNECTION ID 7		CONNECTION ID 8	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
M1353 = ON, enable access up to 50 words. The user can specify the starting register for storing the read/written data in registers below.															
D1480	D1496	D1481	D1497	D1482	D1498	D1483	D1499	D1484	D1500	D1485	D1501	D1486	D1502	D1487	D1503
M1356 = ON, the user can specify the station number of Connection ID1~ID8 in D1900~D1907															
D1900		D1901		D1902		D1903		D1904		D1905		D1906		D1907	
Data length to be read/written for connections (Max 50 pieces of data; 0: no reading/writing)															
D1434	D1450	D1435	D1451	D1436	D1452	D1437	D1453	D1438	D1454	D1439	D1455	D1440	D1456	D1441	D1457
Starting address of the Connection to be read, written or connected.*															
D1355	D1415	D1356	D1416	D1357	D1417	D1358	D1418	D1359	D1419	D1360	D1420	D1361	D1421	D1362	D1422
M1355 = ON, Connection status is user-defined. Activate the Connection manually with M1360~M1367. M1355 = OFF, Connection status is auto-detected. Monitor if the connection is established and activated with M1360~M1367.															
M1360		M1361		M1362		M1363		M1364		M1365		M1366		M1367	
Communication status of Connection ID#1-8 on PLC LINK (ON: communicating)															
M1376		M1377		M1378		M1379		M1380		M1381		M1382		M1383	
Error flag for errors occurred when reading or writing on Connection ID#1-8 (ON = normal; OFF = error)															
M1392		M1393		M1394		M1395		M1396		M1397		M1398		M1399	
Reading complete on PLC LINK Connection ID#1-8 (System resets to OFF after reading is complete on one module.)															
M1408		M1409		M1410		M1411		M1412		M1413		M1414		M1415	
Writing complete on PLC LINK connection ID#1-8 (System resets to OFF after writing is complete on one module.)															
M1424		M1425		M1426		M1427		M1428		M1429		M1430		M1431	

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

Slave PLC*															
CONNECTION ID 1		CONNECTION ID 2		CONNECTION ID 3		CONNECTION ID 4		CONNECTION ID 5		CONNECTION ID 6		CONNECTION ID 7		CONNECTION ID 8	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215

5. Special D and special M corresponding to Connection ID9~ID16: (M1353 = ON, access available for up to 50 words) (DVP-12SE V1.6 and DVP-26SE V2.0 supports 100 words at most.)

MASTER PLC															
CONNECTION ID 9		CONNECTION ID 10		CONNECTION ID 11		CONNECTION ID 12		CONNECTION ID 13		CONNECTION ID 14		CONNECTION ID 15		CONNECTION ID 16	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
M1353 = ON, enable access up to 50 words. The user can specify the starting register for storing the read/written data in registers below.															
D1488	D1504	D1489	D1505	D1490	D1506	D1491	D1507	D1492	D1508	D1493	D1509	D1494	D1510	D1495	D1511
M1356 = ON, the user can specify the station number of Connection ID9~ID16 in D1908~D1915															
D1908		D1909		D1910		D1911		D1912		D1913		D1914		D1915	
Data length to be read/written for connections (Max 50 pieces of data; 0: no reading/writing)															
D1442	D1458	D1443	D1459	D1444	D1460	D1445	D1461	D1446	D1462	D1447	D1463	D1448	D1464	D1449	D1465
Starting address of the Connection to be read, written or connected.*															
D1363	D1423	D1364	D1424	D1365	D1425	D1366	D1426	D1367	D1427	D1368	D1428	D1369	D1429	D1370	D1430
M1355 = ON, Connection status is user-defined. Activate the Connection manually with M1368~M1375. M1355 = OFF, Connection status is auto-detected. Monitor if the connection is established and activated with M1368~M1375.															
M1368		M1369		M1370		M1371		M1372		M1373		M1374		M1375	
Communication status of Connection ID#9-16 on PLC LINK (ON: communicating)															
M1384		M1385		M1386		M1387		M1388		M1389		M1390		M1391	
Error flag for errors occurred when reading or writing on Connection ID#9-16 (ON = normal; OFF = error)															
M1400		M1401		M1402		M1403		M1404		M1405		M1406		M1407	
Reading complete on PLC LINK Connection ID#9-16 (System resets to OFF after reading is complete on one module.)															
M1416		M1417		M1418		M1419		M1420		M1421		M1422		M1423	
Writing complete on PLC LINK connection ID#9-16 (System resets to OFF after writing is complete on one module.)															
M1432		M1433		M1434		M1435		M1436		M1437		M1438		M1439	

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

Slave PLC*															
CONNECTION ID 9		CONNECTION ID 10		CONNECTION ID 11		CONNECTION ID 12		CONNECTION ID 13		CONNECTION ID 14		CONNECTION ID 15		CONNECTION ID 16	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215

*Note:
- Default setting for starting reference of the Slave (DVP-PLC) to be read: H1064 (D100)
- Default setting for starting reference of the Slave (DVP-PLC) to be written: H10C8 (D200)

6. Special D and special M corresponding to Connection ID17~ID24: (M1353 = ON, access available for up to 100 words) (Model supported: DVP-12SE V1.6 and DVP-26SE V2.0)

MASTER PLC															
CONNECTION ID 17		CONNECTION ID 18		CONNECTION ID 19		CONNECTION ID 20		CONNECTION ID 21		CONNECTION ID 22		CONNECTION ID 23		CONNECTIO N ID 24	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
DVP-SE supports M1353. When M1353 is On, 32 stations in the Link and the function of reading/writing more than 16 data (SET M1353) are enabled. The user can specify the starting register for storing the read/written data in registers below.															
D1576	D1592	D1577	D1593	D1578	D1594	D1579	D1595	D1580	D1596	D1581	D1597	D1582	D1598	D1583	D1599
If M1356 is ON, users can set the station numbers of Connection ID17~ID24 in D1916~D1923. The master station sends commands according to the station numbers set.															
D1916		D1917		D1918		D1919		D1920		D1921		D1922		D1923	
Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data
D1544	D1560	D1545	D1561	D1546	D1562	D1547	D1563	D1548	D1564	D1549	D1565	D1550	D1566	D1551	D1567
Starting address of the Connection to be read, written or connected.*															
D1512	D1528	D1513	D1529	D1514	D1530	D1515	D1531	D1516	D1532	D1517	D1533	D1518	D1534	D1519	D1535
Activation status of connection ID#17-24 on PLC LINK (ON: activated)															
M1440		M1441		M1442		M1443		M1444		M1445		M1446		M1447	
Communication status of Connection ID#17-24 on PLC LINK (ON: communicating)															
M1456		M1457		M1458		M1459		M1460		M1461		M1462		M1463	
Error flag for errors occurred when reading or writing on Connection ID#17-24 (ON = normal; OFF = error)															
M1472		M1473		M1474		M1475		M1476		M1477		M1478		M1479	
Reading complete on PLC LINK Connection ID#17-24 (System resets to OFF after reading is complete on one module.)															
M1488		M1489		M1490		M1491		M1492		M1493		M1494		M1495	

Writing complete on PLC LINK connection ID#17-24 (System resets to OFF after writing is complete on one module.)							
M1504	M1505	M1506	M1507	M1508	M1509	M1510	M1511

CONNECTION ID 17		CONNECTION ID 18		CONNECTION ID 29		CONNECTION ID 20		CONNECTION ID 21		CONNECTION ID 22		CONNECTION ID 23		CONNECTIO N ID 24	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215

- Default start communication address D1512 ~ D1519 to be read = H1064 (D100)
- Default start communication address D1528 ~ D1535 to be written = H10C8 (D200)

7. Special D and special M corresponding to Connection ID25~ID32: (M1353 = ON, access available for up to 100 words) (Mode supported: DVP-12SE V1.6 and DVP-26SE V2.0)

MASTER PLC															
CONNECTION ID 25		CONNECTION ID 26		CONNECTION ID 27		CONNECTION ID 28		CONNECTION ID 29		CONNECTION ID 30		CONNECTION ID 31		CONNECTIO N ID 32	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
DVP-SE supports M1353. When M1353 is On, 32 stations in the Link and the function of reading/writing more than 16 data (SET M1353) are enabled. The user can specify the starting register for storing the read/written data in registers below.															
D1584	D1600	D1585	D1601	D1586	D1602	D1587	D1603	D1588	D1604	D1589	D1605	D1590	D1606	D1591	D1607
If M1356 is ON, users can set the station numbers of Connection ID25~ID32 in D1924~D1931. The master station sends commands according to the station numbers set.															
D1924		D1925		D1926		D1927		D1928		D1929		D1930		D1931	
Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data	Number of data
D1552	D1568	D1553	D1569	D1554	D1570	D1555	D1571	D1556	D1572	D1557	D1573	D1558	D1574	D1559	D1575
Starting address of the Connection to be read, written or connected.*															
D1520	D1536	D1521	D1537	D1522	D1538	D1523	D1539	D1524	D1540	D1525	D1541	D1526	D1542	D1527	D1543
Activation status of connection ID#25-32 on PLC LINK (ON: activated)															
M1448		M1449		M1450		M1451		M1452		M1453		M1454		M1455	
Communication status of Connection ID#25-32 on PLC LINK (ON: communicating)															
M1464		M1465		M1466		M1467		M1468		M1469		M1470		M1471	
Error flag for errors occurred when reading or writing on Connection ID#25-32 (ON = normal; OFF = error)															
M1480		M1481		M1482		M1483		M1484		M1485		M1486		M1487	
Reading complete on PLC LINK Connection ID#25-32 (System resets to OFF after reading is complete on one module.)															
M1496		M1497		M1498		M1499		M1500		M1501		M1502		M1503	
Writing complete on PLC LINK connection ID#25-32 (System resets to OFF after writing is complete on one module.)															
M1512		M1513		M1514		M1515		M1516		M1517		M1518		M1519	

CONNECTION ID 25		CONNECTION ID 26		CONNECTION ID 27		CONNECTION ID 28		CONNECTION ID 29		CONNECTION ID 30		CONNECTION ID 31		CONNECTIO N ID 32	
Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in	Read out	Write in
D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215	D100 │ D115	D200 │ D215

- Default start communication address D1520 ~ D1527 to be read = H1064 (D100)
- Default start communication address D1536 ~ D1543 to be written = H10C8 (D200)

8. Explanation: (up to 16 connections can be supported.)
   a) PLC LINK is based on MODBUS communication protocol.
   b) Baud rate and communication format of all phariferal devices connected to the Slave PLC should be the same as the communication format of Master PLC, no matter which COM port of Slave PLC is used.
   c) When M1356 = OFF(Default), the station number of the connection ID1 can be designated by D1399 of Master PLC through PLC LINK, and PLC will automatically assign ID2~ID16 with consecutive station numbers according to the station number of ID1. For example, if D1399 = K3, Master PLC will send out communication commands to ID1~ID16 which carry station number K3~K18. In addition, care should be taken when setting the station number

of Slaves. All the station number of the connection IDs should not be the same as the station number of the Master PLC, which is set up in D1121/D1255.

d) When both M1353 and M1356 are ON, the station number of connection ID1~ID16 can be specified by the user in D1900~D1915 of Master PLC. For example, when D1900~D1903 = K3, K3, K5, K5, Master PLC will access the Slave with station number K3 for 2 times, then the slave with station number K5 for 2 times as well. Note that all station numbers of slaves should not be the same as the station number of the Master PLC, and M1353 must be set ON for this function.

e) Station number selection function (M1356 = ON) is supported by versions of ES2/EX2 v1.4.2 or later, SS2/SX2 v1.2 or later, and SA2 v1.0 or later.

9. Explanation: (up to 32 connections can be supported. The models which are supported now are DVP-12SE V1.6 and DVP-26SE V2.0)

   a) PLC LINK is based on MODBUS communication protocol.

   b) Baud rate and communication format of all phariferal devices connected to the Slave PLC should be the same as the communication format of Master PLC, no matter which COM port of Slave PLC is used.

   c) When M1356 = OFF (Default), the station number of the connection ID1 can be designated by D1399 of Master PLC through PLC LINK, and PLC will automatically assign ID2~ID16 with consecutive station numbers according to the station number of ID1. (When M1356 = ON, the station number of the connection ID1 can be designated by D1399 of Master PLC through PLC LINK, and PLC will automatically assign ID2~ID32 with consecutive station numbers according to the station number of ID1). For example, if D1399 = K3, and M1353 = Off, Master PLC will send out communication commands to ID1~ID16 which carry station number K3~K18. If D1399 = K3, and M1353 = On, In addition, Master PLC will send out communication commands to ID1~ID32 which carry station number K3~K34. In addition, care should be taken when setting the station number of connections. All station numbers of slaves should not be the same as the station number of the Master PLC, which is set up in D1121/D1255.

   d) When both M1353 and M1356 are ON, the station number of ID1~ID32 can be specified by the user in D1900~D1931 of Master PLC. For example, when D1900~D1903 = K3, K3, K5, K5, Master PLC will access the Slave with station number K3 for 2 times, then the station number of connections K5 for 2 times as well. Note that all station numbers of connections should not be the same as the station number of the Master PLC (D1121/D1255), and M1353 must be set ON for this function.

   e) When M1356 is ON, the station number selection function is enabled.

10. Operation:

   a) Set up the baud rates and communication formats. Master PLC and all connected Slave PLCs should have the same communication settings. COM1_RS-232: D1036, COM2_RS-485: D1120, COM3_RS-485: D1109.

   b) Set up Master PLC ID by D1121 and the connection ID by D1399. Then, set connection ID of each slave PLC. The ID of master PLC and slave PLC cannot be the same.

   c) Set data length for accessing. (If data length is not specified, PLC will take default setting or the previous value as the set value. For details of data length device, please refer to the tables above)

   d) Set starting address of the connection to be accessed. (Default setting for starting device to be read: H1064 (D100); default setting for starting device to be written: H10C8 (D200). For details of starting device, please refer to the tables above)

   e) Steps to start PLC LINK:

      - Set ON M1354 to enable simultabeous data read/write in a polling of PLC LINK.
      - M1355 = ON, Slave status is user-defined. Set the linking statuses of Connection ID 1~Connection ID 16 (Connection ID 1~Connection ID 32) manually by M1360~M1375 (M1360~M1375 and M1440~M1455). M1355 = OFF, the linking statuses of Connection ID 1~Connection ID 16 (Connection ID 1~Connection ID 32) are auto-detected. The linking statuses of Connection ID 1~Connection ID 32 can be monitored by M1360~M1375, and M1440~M1455.

- Select auto mode on PLC LINK by M1351 or manual mode by M1352 (Note that the 2 flags should not be set ON at the same time.) After this, set up the times of polling cycle by D1431.
- Finally, enable PLC LINK (M1350)

11. The Operation of Master PLC:
    a) M1355 = ON indicates that connection status is user-defined. Set the linking status of Connection ID 1~Connection ID 16 (Connection ID 1~Connection ID 32) manually by M1360~M1375 (M1360~M1375 and M1440~M1455).
    b) M1355 = OFF indicates that the linking statuses of Connection ID 1~Connection ID 16 (Connection ID 1~Connection ID 32) are auto-detected. The linking statuses of Connection ID 1~Connection ID 32 can be monitored by M1360~M1375, and M1440~M1455.
       - Enable PLC LINK (M1350). Master PLC will detect the connected Slaves and store the number of connected PLCs in D1433. The time for detection differs by number of connected Slaves and time-out setting in D1129.
       - M1360~M1375 indicate the linking statuses of Connection ID 1~Connection ID 16. If M1353 is ON, M1360~M1375 and M1440~M1455 will indicate the linking statuses of Connection ID 1~Connection ID 32.
       - If no slave is detected, M1350 will be OFF and PLC LINK will be stopped.
       - PLC will only detect the number of slaves at the first time when M1350 turns ON.
       - After auto-detection is completed, master PLC starts to access each connected slave. Once slave PLC is added after auto-detection, master PLC cannot access it unless auto-detection is conducted again.

2

    c) Simultaneous read/write function (M1354) has to be set up before enabling PLC LINK. Setting up this flag during PLC LINK execution will not take effect.
    d) When M1354 = ON, PLC takes Modbus Function H17 (simultaneous read/write function) for PLC LINK communication function. If the data length to be written is set to 0, PLC will select Modbus Function H03 (read multiple WORDs) automatically. In the same way, if data length to be read is set to 0, PLC will select Modbus Function H06 (write single WORD) or Modbus Function H10 (write multiple WORDs) for PLC LINK communication function.
    e) When M1353 = OFF, PLC LINK accesses the Slave with max 16 words, and the data is automatically stored in the corresponding registers. When M1353 = ON, up to 100 words are accessible and the user can specify the starting register for storing the read/written data.
       For example, if the register for storing the read/written data on Connection ID1 is specified as D1480 = K500, D1496 = K800, access data length D1434 = K50, D1450 = K50, registers of Master PLC D500~D549 will store the data read from Connection ID1, and the data stored in D800~D849 will be written into Connection ID1.
    f) Master PLC conducts reading before writing. Both reading and writing is executed according to the range specified by user.
    g) Master PLC accesses slave PLCs in order, i.e. data access moves to next slave only when access on previous slave is completed.
    h) Modbus Function H03 will be replaced by Modbus Function H04 for read/write function code. M1700~M1715 are corresponding to Connection ID 1~16 orderly; when the status is ON, the read/write function code can be changed from H04 to H04 for the following series.

Series	ES2/ EX2	ES2-C	ES2-E EC5	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware version	V3.48	V3.48	V1.0	V3.0	V3.60	--	V1.0	V3.0

M1700~M1731 are corresponding to Connection ID 1~32 for 26SE series.

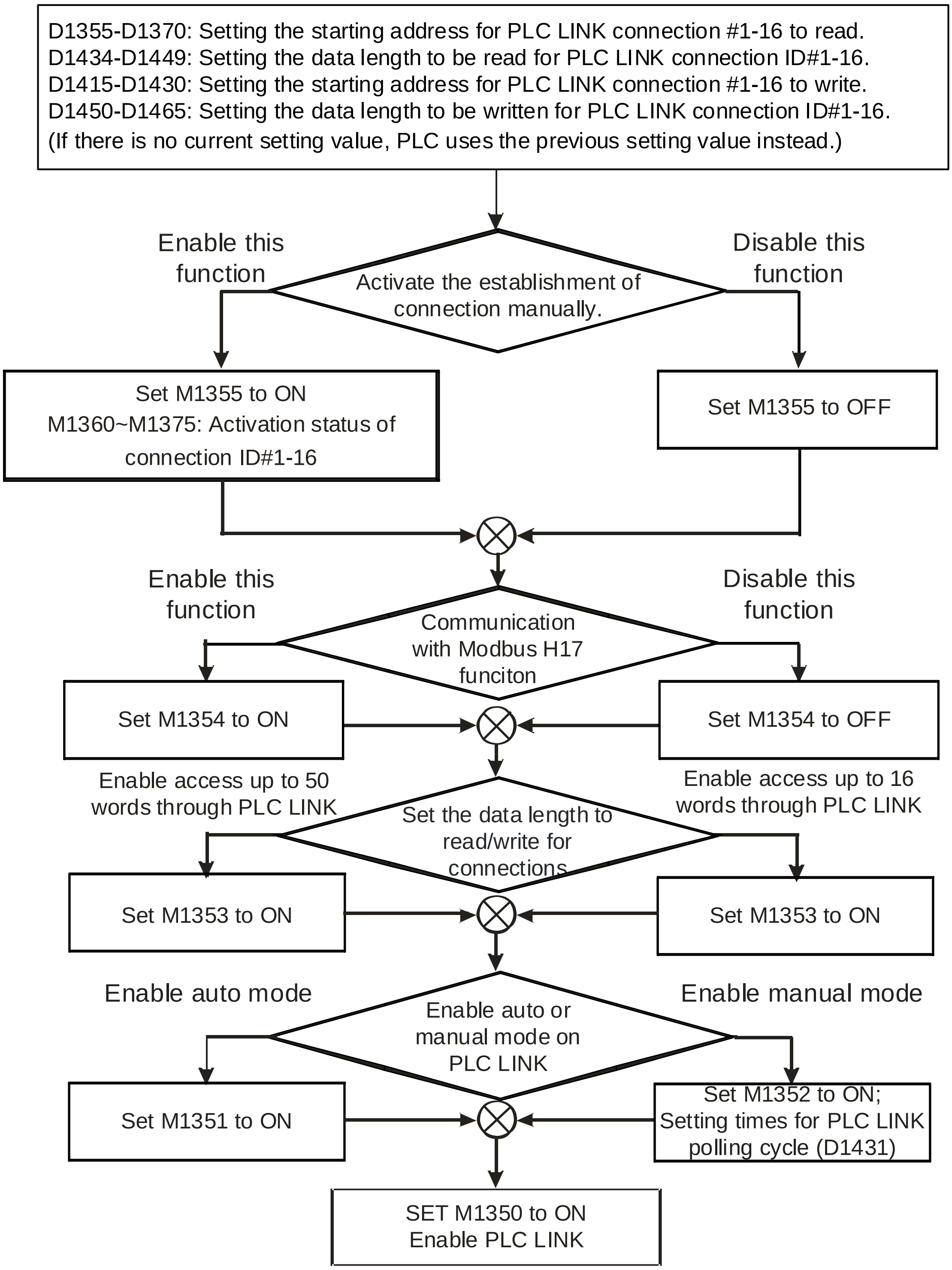
12. Auto mode and Manual mode:
    a) Auto mode (M1351): when M1351 = ON, Master PLC will access connections as the operation described above, and stop the polling till M1350 or M1351 is OFF.
    b) Manual mode (M1352): When manual mode is selected, times of polling cycle in D1431 has to be set up. A full polling cycle refers to the completion of accessing all connecitons.

When PLC LINK is enabled, D1432 starts to store the times of polling. When D1431 = D1432, PLC LINK stops and M1352 is reset. When M1352 is set ON again, PLC will start the polling according to times set in D1431 automatically.

c) Note:

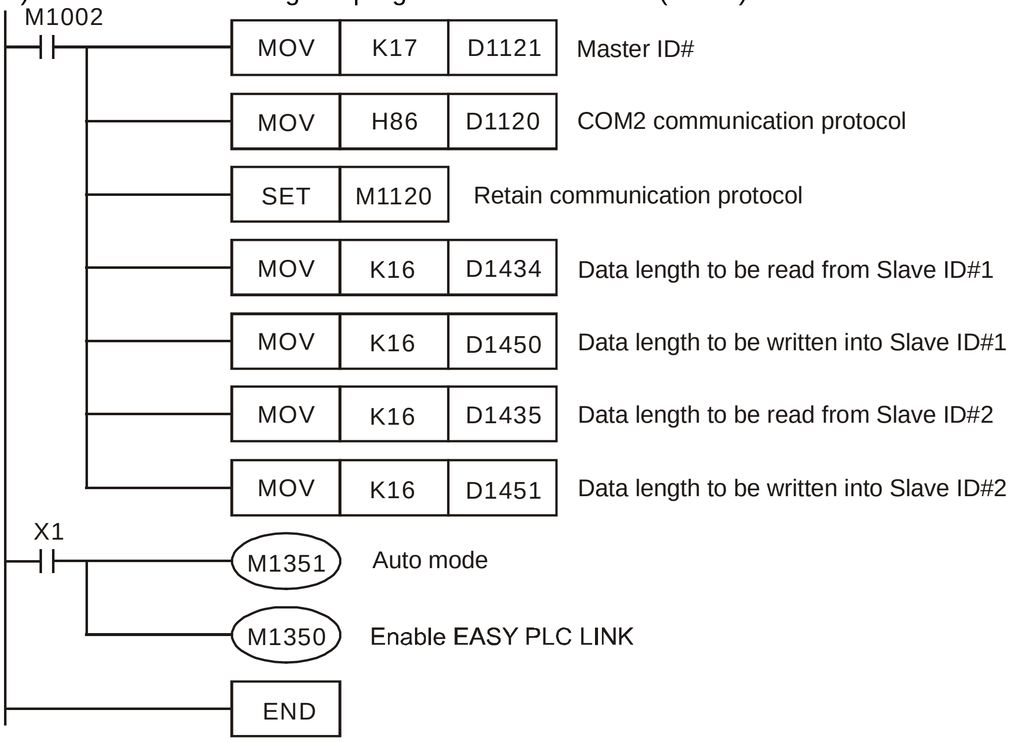
- Auto mode M1351 and manual mode M1352 cannot be enabled at the same time. If M1351 is enabled after M1352 is ON, PLC LINK will stop and M1350 will be reset.
- Communication timeout setting can be modified by D1129 with available range 200 ≦D1129 ≦ 3000. PLC will take the upper / lower bound value as the set value if the specified value is out of the available range. D1129 has to be set up before M1350 = ON.
- PLC LINK function is only valid when baud rate is higher than 1200 bps. When baud rate is less than 9600 bps, please set communication time-out to more than 1 second.
- The communication is invalid when data length to be accessed is set to 0.
- Access on 32-bit high speed counters (C200~C255) is not supported.
- Available range for D1399: 1 ~ 230. PLC will take the upper / lower bound value as the set value if the specified value exceeds the availanle range.
- D1399 has to be set up before enabling PLC LINK. Setting up this register during PLC LINK execution will not take effect.
- Advantage of using D1399 (Designating the ID of starting Slave):
  In old version PLC LINK, PLC detects connection from ID1 to ID16. Therefore, when PLC LINK is applied in multi-layer networks, e.g. 3 layers of networks, the Connection ID of 2nd and 3rd layer will be repeated. When Connection ID is repeated, i.e. the same as Master ID, the connections will be passed. In this case, only 15 connections can be connected in 3rd layer. To solve this problem, D1399 can be applied for increasing the connectable connecitons in multi-layer network structure.

13. Operation flow chart: In the flow chart below, there are 16 connections, and 50 words are accessed.

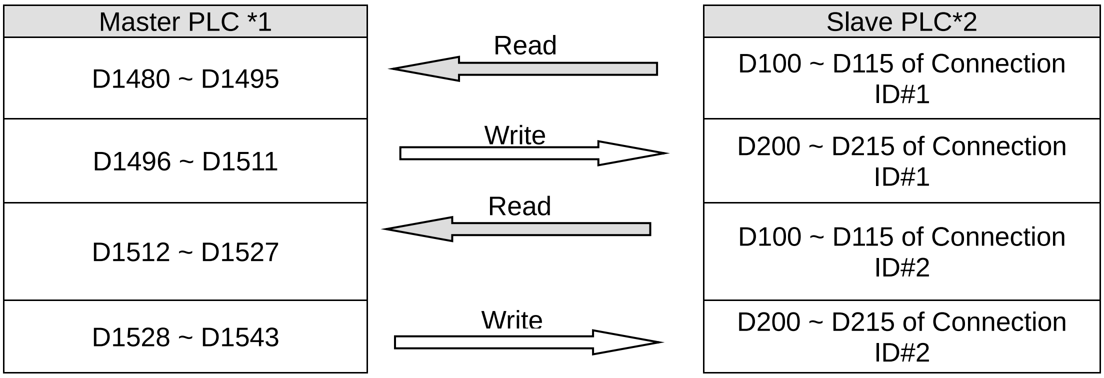


14. Example 1: Connect 1 Master and 2 Slaves by RS-485 and exchange 16 data between Master and Slaves through PLC LINK

a) Write the ladder diagram program into Master PLC (ID#17)



b) When X1 = On, the data exchange between Master and the two Slaves will be automatically executed by PLC LINK. The data in D100 ~ D115 in the two Slaves will be read into D1480 ~ D1495 and D1512 ~ D1527 of the Master, and the data in D1496 ~ D1511 and D1528 ~ D1543 will be written into D200 ~ D215 of the two Slaves.



c) Assume the data in registers for data exchange before enabling PLC LINK (M1350 = OFF) is as below:

Master PLC	Preset value	Slave PLC	Preset value
D1480 ~ D1495	K0	D100 ~ D115 of Connection ID#1	K5,000
D1496 ~ D1511	K1,000	D200 ~ D215 of Connection ID#1	K0
D1512 ~ D1527	K0	D100 ~ D115 of Connection ID#2	K6,000
D1528 ~ D1543	K2,000	D200 ~ D215 of Connection ID#2	K0

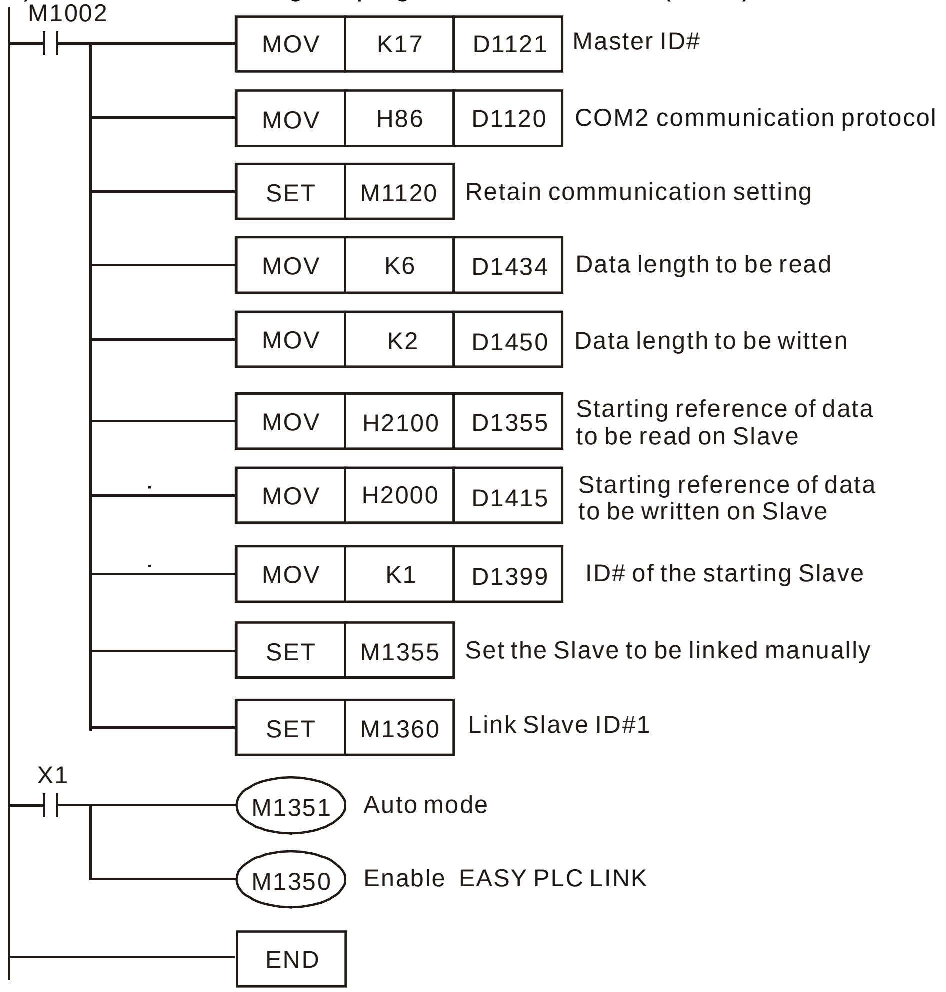
After PLC LINK is enabled (M1350 = ON), the data in registers for data exchange becomes:

Master PLC	Preset value	Slave PLC	Preset value
D1480 ~ D1495	K5,000	D100 ~ D115 of Connection ID#1	K5,000
D1496 ~ D1511	K1,000	D200 ~ D215 of Connection ID#1	K1,000

Master PLC	Preset value	Slave PLC	Preset value
D1512 ~ D1527	K6,000	D100 ~ D115 of Connection ID#2	K6,000
D1528 ~ D1543	K2,000	D200 ~ D215 of Connection ID#2	K2,000

d) Up to16 Slaves can be accessed through PLC LINK. For allocation of D100 ~ D115 and D200 ~ D215 in each Slave PLC, please refer to the tables of Special M and Special D of this function in previous pages.

15. Example 2: Conncet DVP-PLC with VFD-M inverter and control the RUN, STOP, Forward operation, Reverse operation through PLC LINK.
    a) Write the ladder diagram program into Master PLC (ID#17)



    b) M1355 = ON. Set the Slave to be linked manually by M1360~M1375. Set ON M1360 to link Connection ID#1.
    c) Address H2100-H2105 maps to registers D1480-D1485 of PLC. When X1 = ON, PLC LINK executes, and the data in H2100-H2105 will be displayed in D1480-D1485.
    d) Address H2000-H2001 maps to registers D1496-D1497 of PLC. When X1 = ON, PLC LINK executes, and the parameter in H2000-H2001 will be specified by D1496-D1497.
    e) Commands of VFD can be specified by changing the value in D1496. (e.g. D1496 = H12=>VFD forward operation; D1496 = H1=> VFD stops)
    f) Frequency of VFD can be specified by changing the value in D1497. (e.g. D1497 = K5000, set VFD frequency as 50kHz.)
    g) In addition to VFD AC motor drives, devices support MODBUS protocol such as DTA/DTB temperature controllers and ASDA servo drives can also be connected as Slaves. Up to 16 Slaves can be connected.
16. TD1354 is a PLC link scan cycle (unit: 1ms), and max. display value is K32000. D1354 = K0 when PLC Link stops or when the first scan is completed.

Function Group Frequency Detection Function

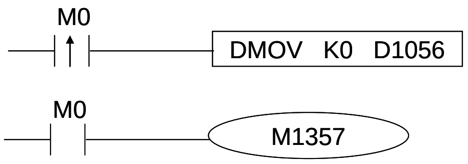
Number M1357-M1359, D1056-D1059, D1246-D1247

Contents:

1. The special M devices and the Special D devices which are related to the frequency detection function are listed below.

Pulse input	Enabling the frequency detection	Showing the input frequency (Unit: 0.001Hz)
X0	M1357	D1056/D1057 (32 bits)
X1	M1358	D1058/D1059 (32 bits)
X2	M1359	D1246/D1247 (32 bits)

2. The minimum input frequency which can be detected by the function is 0.5Hz (K500), the maximum input frequency which can be detected by the function is 1KHz (K1000000). If the input frequency is less than 0.5Hz, or there is no pulse input for more than 2 seconds, the value in the corresponding special D device will automatically become 0. If the input frequency exceeds 1KHz, the PLC will continue catch the input frequency. If the input frequency exceeds the hardware specifications for the input, the PLC will not be able to catch the input frequency.
3. If the frequency detection function is disabled (the special M device is Off), the last value which is stored in the special D device will be retained.
4. If the input frequency is less than 100Hz, the error will be less than one ten thousandth. If the input frequency exceeds 100Hz, the error will become bigger, but the maximum error will not exceed one one thousandth.
5. Difference between the frequency detection function and SPD: The frequency detection function is mainly used to detect the frequencies less than 1KHz, and is used in the application environments which need high precision (unit: 0.001Hz). For example, the frequency detection function can be used to monitor the output frequency of a generator.
6. After the frequency detection function is enabled, the other functions of the input will not be enabled. (For example, the external interrupt or SPD will not be enabled after the frequency detection function is enabled.)
7. DVP-ES2/EX2 series PLCs (exclusive of DVP-ES2-E series PLCs) whose firmware version is 3.22 (or above), ES2-C (V3.68), SA2 (V3.02) and DVP-SX2 (V2.66) or later versions support this function.
8. Example: Detecting X0's input frequency
   Program in the PLC:



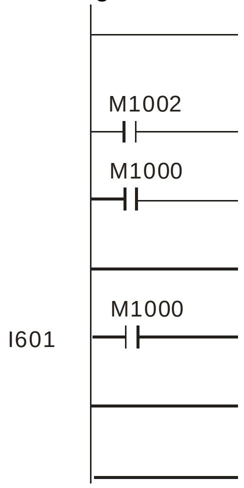
If X0' s input frequency is 50Hz, the 32-bit value in (D1057, D1056) will be K50000.

Function Group Fetching the Value in a Hardware Counter
Number M1598-M1599, D1150-D1153
Contents:

1. The special M devices and the Special D devices which are related to the function of fetching the value in a hardware counter are listed below.

Hardware counter	Fetchinng signal	Enabling the fetching of the value in the hardware counter	Value which is fetched
C243/C245/C246/C247/C248/C251/C252	X6	M1598	D1150/D1151 (32 bits)
C244/C249/C250/C253/C254	X7	M1599	D1152/D1153 (32 bits)

2. The function needs to be used with an external interrupt (X6 (I600/I601) or X7 (I700/I701)). The value in a hardware counter is moved to a special D device when there is a transition in a fetching signal from low to high or form high to low. The setting of an external interrupt determines when the value in a hardware counter is moved to a special D device.
3. DVP-ES2/EX2/SS2 series PLCs whose firmware version is 3.28 (or above), and DVP-SA2/SX2 series PLCs whose firmware version is 2.82 (or above) support this function.
4. Example: The value in C243 is fetched when there is a transition in X6's signal from low to high. Program in the PLC:



Function Group When the conditional contacts are closed, execute the ramp-down on the outputs
Number M1334, M1335
Contents:

1. When M1334 or M1335 is enabled, execute API59 PLSR/DPLSR instructions on Y0 or Y2 to ramp-down when the conditional contacts are closed.
2. When M1334 or M1335 is enabled, execute API158 DDRVI or API159 DDRVA instructions on CH0 (CH1) to ramp-down when the conditional contacts are closed.
3. Once the conditional contacts are closed, the deceleration stops and the flags M1334 and M135 are cleared automatically. You need to enable the flags M1334 and M1335 again for next execution.
4. This function is available for the followings:

Series	ES2/EX2	ES2-C	ES2-E EC5	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware version (including later versions)	V3.42	V3.48	V1.00	V2.86	V3.28	V2.02	V1.0	V3.0

Function Group If the PLC detects that the external 24V voltage is unstable, the error LED flashes
Number M1019
Contents:

1. When M1019 is ON, if the PLC detects the external 24V voltage is unstable, the error LED keeps flashing.
2. When M1019 is OFF, if the PLC detects the external 24V voltage is below 17.8V, the error LED flashes. After the PLC detects the external voltage is normal again for more than 2 seconds, the error LED stops flashing.
3. This function is available for the followings:

Series	ES2/ EX2	ES2-C	ES2-E EC5	12SA2/ SX2	SS2	26SE	26SE	28SA2
Firmware version or later versions	V3.60	V3.60	V1.00	V3.00	V3.50	V1.92	V1.00	V3.0

Function Group Read MAC address from the left side network module EN01
Number M1145 (should work with D1400~1403)
Contents:

1. Enter K101 in D1400 to read MAC Address from the 2nd left side EN01 module
2. Once M1145 is set, PLC stores the MAC address of EN01 in D1401~1403.
3. For example if the MAC address of EN01 is 11:22:33:44:55:66, the contents of D1401~D1403 are D1401 = 0x1122, D1402 = 0x3344, D1403 = 0x5566.
4. This function is available for the followings:

Series	12SA2 / SX2	12SE
Firmware version or later versions	V3.00	V2.00

Function Group Check the input/output points of the digital extension module or number of the special modules when power-on.
Number M1582 (should work with D1248, D1254, D1385)
Contents:

1. Set M1582 to ON to check the digital input points (D1248), the digital output points (D1254) or the number of the special modules (D1385) when power-on. (ON: enable; OFF: disable)

   D1248: Enter the number of input points on DIO module for the PLC to check if the number is matched with the value in D1142 when power-on.
   D1254: Enter the number of output points on DIO module for the PLC to check if the number is matched with the value in D1143 when power-on.
   D1385: Enter the number of connected module for the PLC to check if the number is matched with the value in D1143 when power-on.

2. When the numbers don't match, the PLC stops running and record the error message. The error code 16#C465 will be stored in D1004.

3. This function is available for the followings:

Series	ES2/EX2/ES2-C	ES2-E
Firmware version or later versions	V3.62	V1.48

Instruction Set 3

This chapter explains all of the instructions which are used with DVP-ES2/EX2/EC5/SS2/SA2/SX2/SE as well as detailed information concerning the usage of the instructions.

Table of Contents

3.1 Basic Instructions (without API numbers)

Instruction	Function	Operand	Execution speed (us)		Steps
			ES2/EX2/EC5 /SS2SA2/SX2	SE	
LD	Load NO contact	X, Y, M, S, T, C	0.76	0.64	1~3
LDI	Load NC contact	X, Y, M, S, T, C	0.78	0.68	1~3
AND	Connect NO contact in series	X, Y, M, S, T, C	0.54	0.58	1~3
ANI	Connect NC contact in series	X, Y, M, S, T, C	0.56	0.62	1~3
OR	Connect NO contact in parallel	X, Y, M, S, T, C	0.54	0.62	1~3
ORI	Connect NC contact in parallel	X, Y, M, S, T, C	0.56	0.64	1~3
ANB	Connect a block in series	N/A	0.68	0.68	1
ORB	Connect a block in parallel	N/A	0.76	0.76	1
MPS	Start of branches. Stores current result of program evaluation	N/A	0.74	0.68	1
MRD	Reads the stored current result from previous MPS	N/A	0.64	0.54	1
MPP	End of branches. Pops (reads and resets) the stored result in previous MPS	N/A	0.64	0.54	1
OUT	Output coil	Y, S, M	0.88	0.68	1~3
SET	Latches the ON status	Y, S, M	0.76	0.68	1~3
RST	Resets contacts, registers or coils	Y, M, S, T, C, D, E, F	2.2	1.04	3
MC	Master control Start	N0~N7	1	0.8	3
MCR	Master control Reset	N0~N7	1	0.8	3
END	Program End	N/A	1	0.8	1
NOP	No operation	N/A	0.4	0.5	1
P	Pointer	P0~P255	0.4	0.5	1
I	Interrupt program pointer	I□□□	0.4	0.5	1
STL*1	Step ladder start instruction	S	2.2	2	1
RET*1	Step ladder return instruction	N/A	1.6	1.4	1
NP	Negative contact to Positive contact	N/A	1.66	0.72	1
PN	Positive contact to Negative contact	N/A	1.62	0.72	1

Note:

*1: Refer to section 5.1 for more information on instructions STL and RET.

*2: The execution speed is obtained by basic test programs, therefore the actual instruction execution time could be longer due to a more complicated program, e.g. program contains multiple interruptions or high speed input/output.

3.2 Explanations to Basic Instructions

Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
LD	X, Y, M, S, T, C	Load NO contact	1~3				

Explanations:

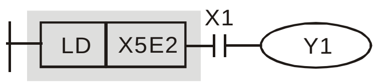
1. The LD instruction is used to load NO contact which connects to left side bus line or starts a new block of program connecting in series or parallel connection.
2. DVP-ES2/EX2 (V3.20), DVP-SS2 (V3.00), DVP-SA2 (V2.60), DVP-SE (V1.20), DVP-SX2 (V2.40), DVP-EC5 (V1.00) or later versions support the operands X, Y, M, and S. These operands can be qualified by E or F. Users have to use WPLSoft version 2.31 (or above) ISPSoft version 2.01 (or above). Note: EC5 series PLC should work with ISPSoft V3.17 or later.

Program example:

Ladder diagram:

Instruction:		Operation:
LD	**X0**	Load NO contact X0
AND	X1	Connect NO contact X1 in series
OUT	Y1	Drive coil Y1

Ladder diagram:



Instruction:		Operation:
LD	**X5E2**	**Load NO contact X3 (SupposeE2=K-2)**
AND	X1	Connect NO contact X1 in series
OUT	Y1	Drive coil Y1

Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
LDI	X, Y, M, S, T, C	Load NC contact	1~3				

Explanations:

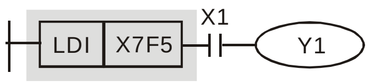
1. The LDI instruction is used to load NC contact which connects to left side bus line or starts a new block of program connecting in series or parallel connection.
2. Refer to instruction LD for more information on the supported firmware versions for each modules.

Program example:

Ladder diagram:

Instruction:		Operation:
LDI	**X0**	Load NC contact X0
AND	X1	Connect NO contact X1 in series
OUT	Y1	Drive coil Y1

Ladder diagram:



Instruction:		Operation:
LDI	**X7F5**	**Load NC contact X12** **(Suppose F5=K3)**
AND	X1	Connect NO contact X1 in series
OUT	Y1	Drive coil Y1

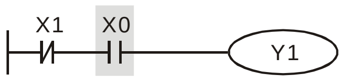
Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
AND	X, Y, M, S, T, C	Connect NO contact in series	1~3				

Explanations:

1. The AND instruction is used to connect NO contact in series.
2. Refer to instruction LD for more information on the supported firmware versions for each modules.

Program example:

Ladder diagram:



Instruction:		Operation:
LDI	X1	Load NC contact X1
AND	**X0**	**Connect NO contact X0 in series**
OUT	Y1	Drive Y1 coil

Ladder diagram:

Instruction:		Operation:
LDI	X1	Load NC contact X1
AND	**X10E2**	**Connect NO contact X20 in series (Suppose E2 = K8)**
OUT	Y1	Drive Y1 coil

Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
ANI	X, Y, M, S, T, C	Connect NC contact in series	1~3				

Explanations:

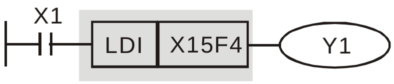
1. The ANI instruction is used to connect NC contact in series.
2. Refer to instruction LD for more information on the supported firmware versions for each modules.

Program example:

Ladder diagram:

Instruction:		Operation:
LD	X1	Load NO contact X1
ANI	**X0**	**Connect NC contact X0 in series**
OUT	Y1	Drive Y1 coil

Ladder diagram:



Instruction:		Operation:
LD	X1	Load NO contact X1
ANI	**X15F4**	**Connect NC contact X11 in series (Suppose F4=K-4)**
OUT	Y1	Drive Y1 coil

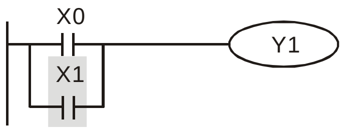
Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/EC5	SS2	SA2 SE	SX2
OR	X, Y, M, S, T, C	Connect NO contact in parallel	1~3				

Explanations:

1. The OR instruction is used to connect NO contact in parallel.
2. Refer to instruction LD for more information on the supported firmware versions for each modules.

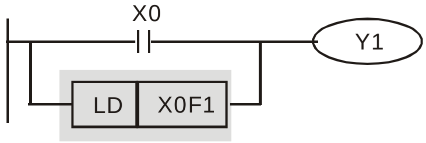
Program example:

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
OR	**X1**	**Connect NO contact X1 in parallel**
OUT	Y1	Drive Y1 coil

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
OR	**X0F1**	**Connect NO contact X5 in parallel (Suppose F1=K5)**
OUT	Y1	Drive Y1 coil

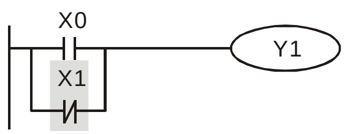
Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
ORI	X, Y, M, S, T, C	Connect NC contact in parallel	1~3				

Explanations:

1. The ORI instruction is used to connect NC contact in parallel.
2. Refer to instruction LD for more information on the supported firmware versions for each modules.

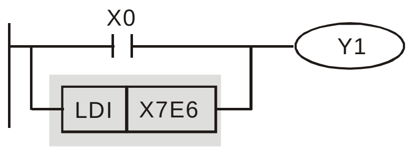
Program example:

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
ORI	**X1**	**Connect NC contact X1 in parallel**
OUT	Y1	Drive Y1 coil

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
ORI	**X7E6**	**Connect NC contact X4 in parallel (Suppose E6=K-3)**
OUT	Y1	Drive Y1 coil

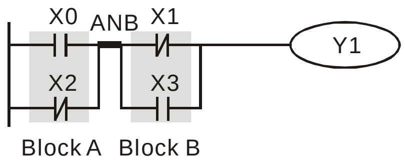
Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/EC5	SS2	SA2 SE	SX2
ANB	Connect a block in series	1				

Explanations:

The ANB instruction is used to connect a circuit block to the preceding block in series. Generally, the circuit block to be connected in series consists of several contacts which form a parallel connection structure.

Program example:

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
ORI	X2	Connect NC contact X2 in parallel
LDI	X1	Load NC contact X1
OR	X3	Connect NO contact X3 in parallel
ANB		Connect circuit block in series
OUT	Y1	Drive Y1 coil

3

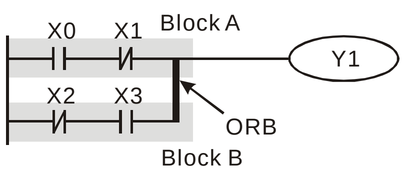
Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
ORB	Connect a block in parallel	1				

Explanations:

The ORB instruction is used to connect a circuit block to the preceding block in parallel. Generally, the circuit block to be connected in parallel consists of several contacts which form a serial connection structure.

Program example:

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
ANI	X1	Connect NC contact X1 in series
LDI	X2	Load NC contact X2
AND	X3	Connect NO contact X3 in series
ORB		Connect circuit block in parallel
OUT	Y1	Drive Y1 coil

Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
MPS	Start of branches. Stores current result of program evaluation	1				

Explanations:

As the start of branches, MPS stores current result of program evaluation at the point of divergence.

Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
MRD	Reads the stored current result from previous MPS	1				

Explanations:

MRD reads the stored current result from previous MPS and operates with the contact connected after MRD.

Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
MPP	End of branches. Pops (reads and resets) the stored result in previous MPS.	1				

Explanations:

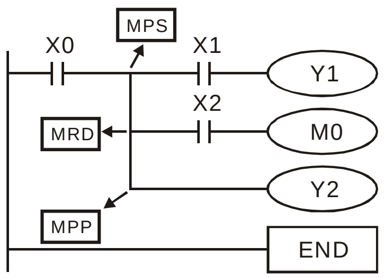
As the end of branches, MPP pops the stored result in previous MPP, which means it operates with the contact connected first then resets the storage memory.

Points to note:

1. Every MPS can not be applied without a corresponding MPP
2. Max. 8 MPS-MPP pairs can be applied.

Program example:

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
MPS		Store current status
AND	X1	Connect NO contact X1 in series
OUT	Y1	Drive Y1 coil
MRD		Read the stored status
AND	X2	Connect NO contact X2 in series
OUT	M0	Drive M0 coil
MPP		Read the stored status and reset
OUT	Y2	Drive Y2 coil
END		End of program

Note: When compiling ladder diagram with WPLSoft, MPS, MRD and MPP will be automatically added to the compiled results in instruction format. However, users programming in instruction mode have to enter branch instructions as required.

Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SA2
OUT	Y, M, S	Output coil	1~3				

Explanations:

1. Output the program evaluation results before OUT instruction to the designated device.
2. Refer to instruction LD for more information on the supported firmware versions for each modules. They support the operands Y, M, and S. These operands can be qualified by E or F. Users have to use WPLSoft version 2.31 or later and ISPSoft version 2.01 or later. After the operands Y, M and S are qualified by E or F, the repeated usages of output coils in the steps is no longer allowed.

State of coil contact

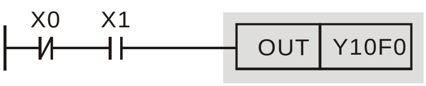
Evaluation result	OUT instruction		
	Coil	Associated Contacts	
		NO contact（normal open）	NC contact（normal close）
FALSE	OFF	Current blocked	Current flows
TRUE	ON	Current flows	Current blocked

Program example:

Ladder diagram:

Instruction:		Operation:
LDI	X0	Load NC contact X0
AND	X1	Connect NO contact X1 in series
OUT	**Y1**	**Drive Y1 coil**

Ladder diagram:



Instruction:		Operation:
LDI	X0	Load NC contact X0
AND	X1	Connect NO contact X1 in series
OUT	**Y10F0**	**Drive Y5 coil (Suppose F0=K-3)**

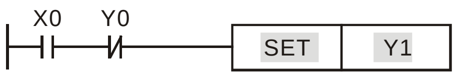
Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
SET	Y, M, S	Latches the ON status	1~3				

Explanations:

1. When the SET instruction is driven, its designated device will be ON and latched whether the SET instruction is still driven. In this case, RST instruction can be applied to turn off the device.
2. Refer to instruction LD for more information on the supported firmware versions for each modules.

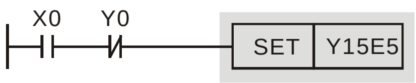
Program example:

Ladder Diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
ANI	Y0	Connect NC contact Y0 in series
SET	**Y1**	**Drive Y1 and latch the status**

Ladder Diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
ANI	Y0	Connect NC contact Y0 in series
SET	**Y15E5**	**Drive Y20 and latch the status (Suppose E5=K3)**

Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
RST	Y, M, S, T, C, D, E, F	Resets contacts, registers or coils	3				

Explanations:

1. Device status when RST instruction is driven:

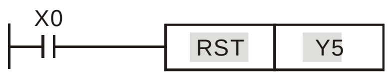
Device	Status
S, Y, M	Coil and contact are set to OFF.
T, C	Current value is cleared. Associated contacts or coils are reset .
D, E, F	The content is set to 0.

State of designated devices remains the same when RST instruction is not executed.

1. Refer to instruction LD for more information on the supported firmware versions for each modules.

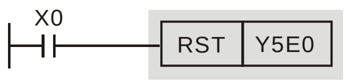
Program example:

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
RST	**Y5**	**Reset contact Y5**

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
RST	**Y5E0**	**Reset contact Y5**
		(Suppose E0=K0)

Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
MC/MCR	N0~N7	Master control Start/Reset	3				

Explanations:

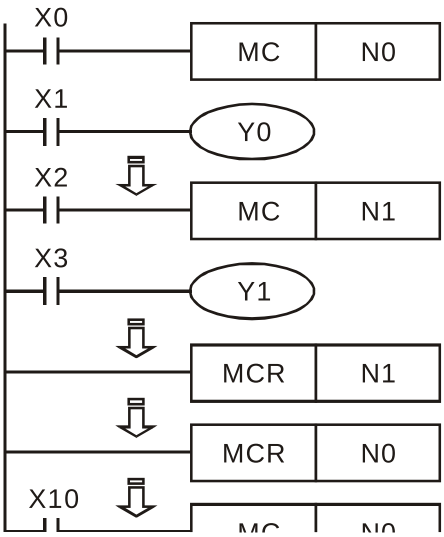
MC is the master-control start instruction. When MC instruction executes, the program execution turns to the designated nest level and executes the instructions between MC and MCR. However, MCR is the master-control reset instruction placed at the end of the designated nest level and no drive contact is required before MCR. When MC/MCR is not active, devices and instructions between MC/MCR will operate as the following table.

Instruction type	Explanation
General purpose timer	Present value = 0, Coil is OFF, No action on associated contact
Subroutine timer	Present value = 0, Coil is OFF, No action on associated contact
Accumulative timer	Coil is OFF, present value and contact status remains
Counter	Coil is OFF, present value and contact status remains
Coils driven by OUT instruction	All OFF
Devices driven by SET/RST instructions	Stay intact
Application instructions	All disabled. The FOR-NEXT nested loop will still execute back and forth for N times. Instructions between FOR-NEXT will act as other instructions between MC and MCR.

Note: MC-MCR master-control instruction supports max 8 layers of nest levels. Please use the instructions in order from N0~ N7.

Program example:

Ladder diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
MC	**N0**	Enable N0 nest level
LD	X1	Load NO contact X1
OUT	Y0	Drive coil Y1
:		
LD	X2	Load NO contact X2
MC	**N1**	Enable N1 nest level
LD	X3	Load NO contact X3
OUT	Y1	Drive coil Y1
:		
MCR	**N1**	Reset N1 nest level

:		
MCR	**N0**	Reset N0 nest level
:		
LD	X10	Load NO contact X10
MC	**N0**	Enable N0 nest level
LD	X11	Load NO contact X11
OUT	Y10	Drive coil Y10
:		
MCR	**N0**	Reset N0 nest level

Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
END	Program End	1				

Explanations:

END instruction needs to be connected at the end of program. PLC will scan from address 0 to END instruction and return to address 0 to scan again.

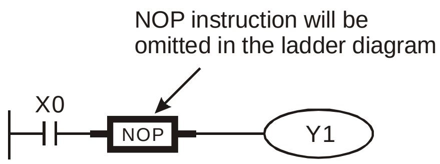
Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
NOP	No operation	1				

Explanation:

NOP instruction does not conduct any operations in the program, i.e. the operation result remains the same after NOP is executed. Generally NOP is used for replacing certain instruction without altering original program length.

Program example:

Ladder Diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
NOP		No operation
OUT	Y1	Drive coil Y1

Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
NP	Negative contact to Positive contact	1				

Explanation:

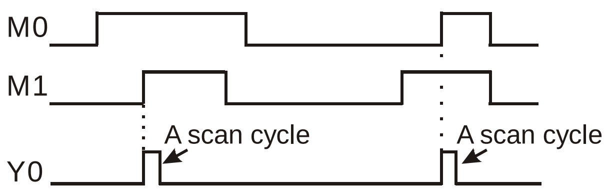
When the conditions preceding NP command change from false to true, NP command (works as contact A) will be ON for a scan cycle. In the next scan cycle it turns OFF.

Program Example:

Ladder Diagram:

Instruction:		Operation:
LD	M0	Load NO contact M0
AND	M1	Connect NO contact M1 in series
NP		Negative contact to Positive contact
OUT	Y0	Drive coil Y0

Timing Diagram:



Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
PN	Positive contact to Negative contact	1				

Explanation:

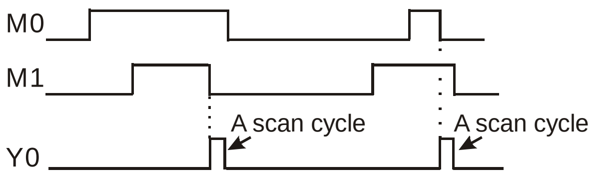
When the conditions preceding PN command change from true to false, PN command (works as contact A) will be ON for a scan cycle. In the next scan cycle it turns OFF.

Program Example:

Ladder Diagram:

Instruction:		Operation:
LD	M0	Load NO contact M0
AND	M1	Connect NO contact M1 in series
PN		Negative contact to Positive contact
OUT	Y0	Drive coil Y0

Timing Diagram:



3.3 Pointers

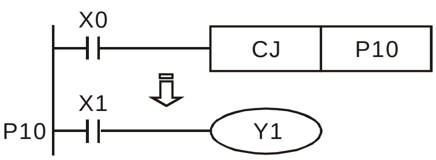
Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/EC5	SS2	SA2 SE	SX2
P	P0~P255	Pointer	1				

Explanation:

Pointer P is used with API 00 CJ and API 01 CALL instructions. The use of P does not need to start from P0, and the No. of P cannot be repeated; otherwise, unexpected errors may occur. For other information on P pointers, please refer to section 2.15 in this manual

Program example 1:

Ladder Diagram:



Instruction:		Operation:
LD	X0	Load NO contact X0
CJ	P10	Jump to P10
:		
P10		Pointer P10
LD	X1	Load NO contact X1
OUT	Y1	Drive coil Y1

3.4 Interrupt Pointers

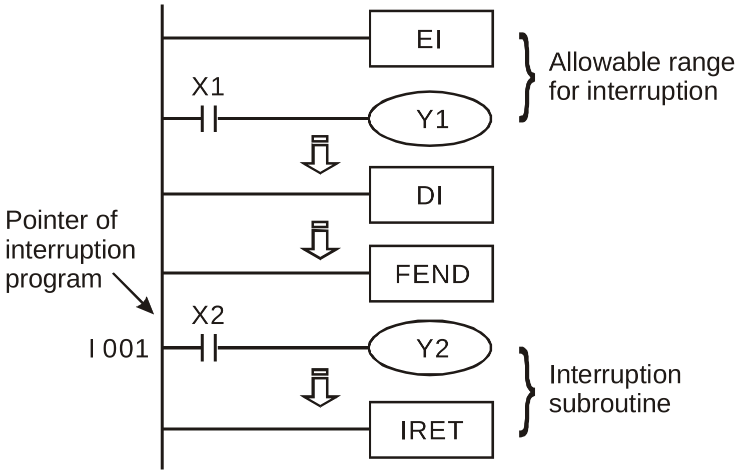
Mnemonic	Function	Program steps	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
I	Interrupt program pointer	1				

Explanations:

An interruption program has to start with a interruption pointer (I□□□) and ends with API 03 IRET. I instruction has to be used with API 03 IRET, API 04 EI, and API 05 DI. For detailed information on interrupt pointes, please refer to section 2.15 in this manual.

Program example:

Ladder diagram:



Instruction code:		Operation:
EI		Enable interruption
LD	X1	Load NO contact X1
OUT	Y1	Drive Y1 coil
:		
DI		Disable interruption
:		
FEND		Main program ends
I001		Interruption pointer
LD	X2	Load NO contact X2
OUT	Y2	Drive Y2 coil
:		
IRET		Interruption return

External interrupt:

ES2 supports 8 external input interrupts: (I000/I001, X0), (I100/I101, X1), (I200/I201, X2), (I300/I301, X3), (I400/I401, X4), (I500/I501, X5), (I600/I601, X6) and (I700/I701, X7). (01, rising-edge trigger, 00, falling-edge trigger)

Timer Interrupts:

ES2 supports 3 timer interrupts: I602~I699, I702~I799, (Timer resolution: 1ms), I805/I899, 1 point (Timer resolution=0.1 ms), available for SE/ES2-E/EC5 version 1.00, for other series, firmware version should be V2.00 or later.

Communication Interrupts:

ES2 supports 3 communication interrupts: I140, I150 and I160.

Counter Interrupts:

ES2 supports 8 high-speed counter interrupts: I010, I020, I030, I040, I050, I060, I070 and I080.

3.5 Application Programming Instructions

1. PLC instructions are provided with a unique mnemonic name to make it easy to remember instructions. In the example below the API number given to the instruction is 12, the mnemonic name is MOV and the function description is Move.

API	Mnemonic			Operands	Function	Controllers			
12	D	MOV	P	(S) (D)	Move	ES2/EX2/ EC5	SS2	SA2 SE	SX2

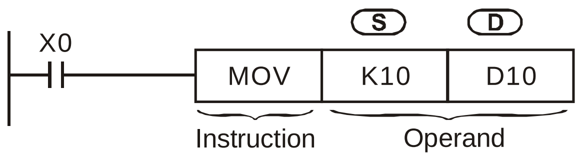
Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MOV, MOVP: 5 steps
S					*	*	*	*	*	*	*	*	*	*	*	DMOV, DMOVP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2	SX2

2. The area of 'Operands' lists the devices (operands) required for the instruction. Identification letters are used to associate each operand with its function, e.g. D-destination, S-source, n, m-number of devices. Additional numeric suffixes will be attached if there are more than one operand with the same function, e.g. S_1, S_2.
3. When using WPLSoft for programming user program, it is not necessary to remember the API number of an instruction since WPLSoft offers drop down list to select an instruction.
4. Applicable controllers are identified by the boxes at the right of the table. For individual instruction properties of Pulse, 16-bit or 32-bit, please refer to the box down the table.
5. Pulse operation requires a 'P' to be added directly after the mnemonic while 32 bit operation requires a 'D' to be added before the mnemonic, i.e. if an instruction was being used with both pulse and 32 bit operation it appears as "D***P" where *** is the basic mnemonic.

Instruction Composition

The application instructions are specified by API numbers 0~--- and each has its mnemonic. When designing the user program with ladder editing program (WPLSoft), users only need to key in the mnemonic, e.g. MOV, and the instruction will be inserted. Instructions consist of either just the instruction or the instruction followed by operands for parameter settings. Take MOV instruction for example:



Mnemonic : Indicates the name and the function of the instruction

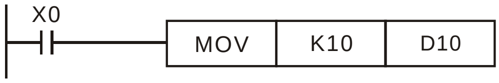
Operand : The parameter setting for the instruction

S	Source: if there are more than one source is required, it will be indicated as S_1, S_2...etc.
D	Destination: if there are more than one destination is required, it will be indicated as D_1, D_2...etc.
If the operand can only be constant K/H or a register, it will be represented as **m**, **m_1**, **m_2**, **n**, **n_1**, **n_2**…etc.	

Length of Operand (16-bit or 32-bit instruction)

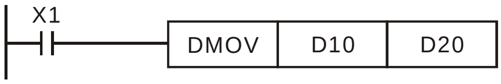
The length of operand can be divided into two groups: 16-bit and 32-bit for processing data of different length. A prefix "D" indicates 32-bit instructions.

16-bit MOV instruction



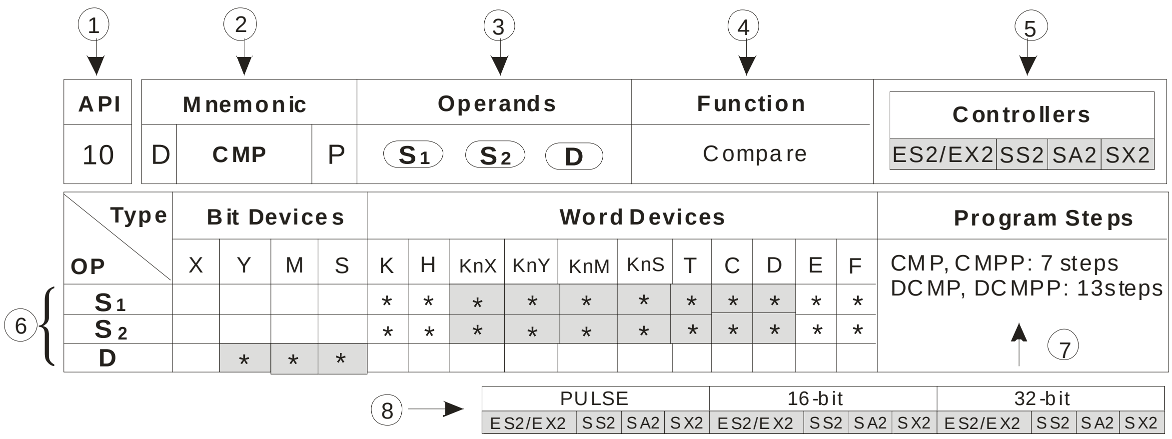
When X0 = ON, K10 will be sent to D10.

32-bit DMOV instruction



When X1 = ON, the content in (D11, D10) will be sent to (D21, D20).

Explanation of the format of application instruction



1. API number for instruction
2. The core mnemonic code of instruction
   A prefix "D" indicates a 32 bit instruction
   A suffix "P" in this box indicates a pulse instruction
3. Operand format of the instruction
4. Function of the instruction
5. Applicable PLC models for this instruction
6. A symbol **"*"** is the device can use the index register. For example, device D of operand S1 supports index E and F.
   A symbol **"*"** is given to device which can be used for this operand
7. Steps occupied by the 16-bit/32-bit/pulse instruction
8. Applicable PLC models for 16-bit/32-bit/pulse execution instruction.

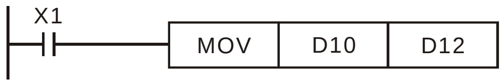
Continuous execution vs. Pulse execution

1. There are two execution types for instructions: continuous execution instruction and pulse instruction. Program scan time is shorter when instructions are not executed. Therefore, using the pulse execution instruction can reduce the scan time of the program.
2. The 'pulse' function allows the associated instruction to be activated on the rising edge of the drive contact. The instruction is driven ON for the duration of one program scan.
3. In addition, while the control input remains ON, the associate instruction will not be executed for the second time. To re-execute the instruction the control input must be turned from OFF to ON again.

Pulse execution instruction

When X0 goes from OFF to ON, MOVP instruction will be executed once and the instruction will not be executed again in the scan period

Continuous execution instruction



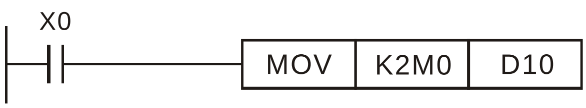
When X1=ON, the MOV instruction can be re-executed again in every scan of program. This is called continuous execution instruction.

Operands

1. Bit devices X, Y, M, and S can be combined into word device, storing values and data for operations in the form of KnX, KnY, KnM and KnS in an application instruction.
2. Data register D, timer T, counter C and index register E, F are designated by general operands.
3. A data register D consists of 16 bits, i.e. a 32-bit data register consists of 2 consecutive D registers.
4. If an operand of a 32-bit instruction designates D0, 2 consecutive registers D1 and D0 will be occupied. D1 is thehigh word and D0 is the low word. This proncipal also applys to timer T and 16-bit counters C0 ~ C199.
5. When the 32-bit counters C200 ~ C255 are used as data registers, they can only be designataed by the operands of 32-bit instructions.

Operand Data format

1. X, Y, M, and S are defined as bit devices which indicate ON/OFF status.
2. 16-bit (or 32-bit) devices T, C, D, and registers E, F are defined as word devices.
3. “Kn” can be placed before bit devices X, Y, M and S to make it a word device for performing word-device operations. (n = 1 refers to 4 bits. For 16-bit instruction, n = K1 ~ K4; for 32-bit instruction, n = K1 ~ K8). For example, K2M0 refers to 8 bits, M0 ~ M7.



When X0 = ON, the contents in M0 ~ M7 will be moved to b0 ~b7 in D10 and b8 ~b15 will be set to “0”.

Kn values

16-bit instruction	32-bit instruction
Designated value: K-32,768 ~ K32,767	Designated value: K-2,147,483,648 ~ K2,147,483,647
16-bit instruction: (K1~K4)	32-bit instruction: (K1~K8)

K1 (4 bits)	0~15
K2 (8 bits)	0~255
K3 (12 bits)	0~4,095
K4 (16 bits)	-32,768~+32,767

K1 (4 bits)	0~15
K2 (8 bits)	0~255
K3 (12 bits)	0~4,095
K4 (16 bits)	0~65,535
K5 (20 bits)	0~1,048,575
K6 (24 bits)	0~167,772,165
K7 (28 bits)	0~268,435,455
K8 (32 bits)	-2,147,483,648~+2,147,483,647

Flags

1. General Flags

   The flags listed below are used for indicating the operation result of the application instruction:

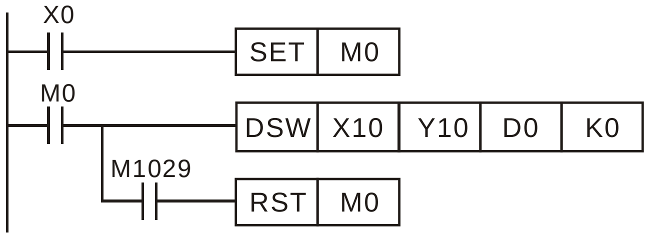
   M1020: Zero flag

   M1021: Borrow flag

   M1022: Carry flag

   M1029: Execution of instruction is completed

   All flags will turn ON or OFF according to the operation result of an instruction. For example, the execution result of instructions ADD/SUB/MUL/DVI will affect the state of M1020 ~ M1022. When the instruction is not executed, the ON/OFF state of the flag will be held. The state of the four flags relates to many instructions. See relevant instructions for more details.



   When X0 = ON, DSW will be enabled.

   When X0 = OFF, M0 is latched. M0 will be reset only when DSW instruction is completed to activate M1029.

2. Error Operation Flags

   Errors occur during the execution of the instruction when the combination of application instructions is incorrect or the devices designated by the operand exceed their range. Other than errors, the flags listed in the table below will be On, and error codes will also appear.

3. Flags to Extend Functions

   Some instructions can extend their function by using some special flags.

   Example: instruction RS can switch transmission mode 8-bit and 16-bit by using M1161.

Device	Explanation
M1067 D1067 D1069	When operational errors occur, M1067 = ON. D1067 displays the error code. D1069 displays the address where the error occurs. Other errors occurring will update the contents in D1067 and D1069. M1067 will be OFF when the error is cleared.
M1068 D1068	When operational errors occur, M1068 = ON. D1068 displays the address where the error occurs. Other errors occurring wil not update the content in D1068. RST instruction is required to reset M1068 otherwise M1068 is latched.

Limitations for times of using instructions

Some instructions can only be used a certain number of times in a program. These instructions can be modified by index registers to extend their functionality.

1. Instructions can be used once in a program:

   API 60 (IST)

   API 155 (DABSR)

2. Instruction can be used twice in a program:

   API 77 (PR)

3. Instruction can be used 8 times in a program:

   API 64 (TTMR)

4. For counters C232~C242, the total max times for using DHSCS, DHSCR and DHSZ instructions: 6. DHSZ can only be used less than 6 times.
5. For counters C243, C245~C248, C251, C252, the total max times for using DHSCS, DHSCR and DHSZ instructions: 4. DHSZ takes up 2 times of the total available times.
6. For counters C244, C249, C250, C253, C254, the total max times for using DHSCS, DHSCR and DHSZ instructions: 4. DHSZ takes up 2 times of the total available times.

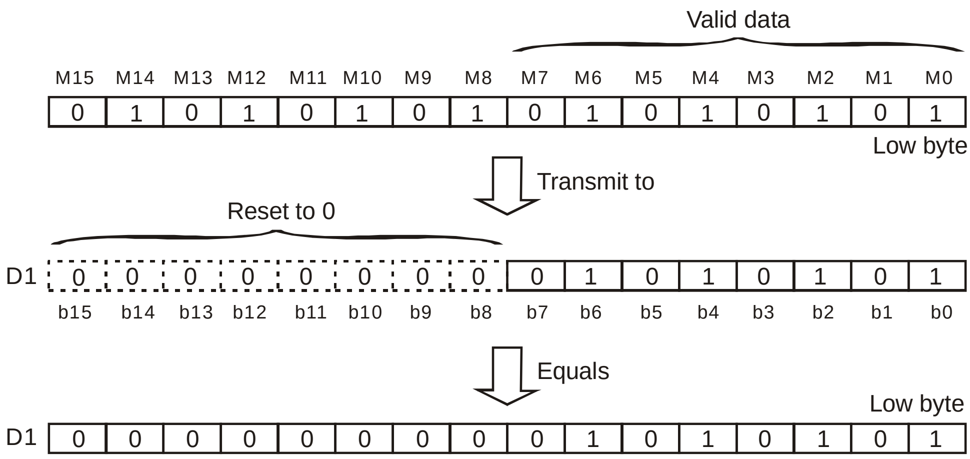
Limitation of synchronized execution

Most instructions have no limitation on the times to be used in a program, but there are limitations on the number of instruction to be executed in the same scan cycle.

1. Only 1 instruction can be executed at the same scan cycle: API 52 MTR, API 69 SORT, API 70 TKY, API 71 HKY, API 72 DSW, API 74 SEGL, API 75 ARWS.
2. Only 4 instruction can be executed at the same scan cycle: API 56 SPD, API 169 HOUR.
3. There is no limitation on the times of using the high-speed output instructions API 57 PLSY, API 58 PWM, API 59 PLSR, API 156DZRN, API 158 DDRVI, API 159 DDRVA and API 195 DPTPO, but only one high-speed output instruction will be executed in the same scan time.
4. There is no limitation on the times of using the communication instructions API 80 RS, API 100 MODRD, API 101 MODWR, API 102 FWD, API 103 REV, API 104 STOP, API 105 RDST, API 106 RSTEF , API 150 MODRW, but only one communication instruction will be executed on single COM port during the same scan cycle.

Numeric Values

1. Devices indicates ON/OFF status are called bit devices, e.g. X, Y, M and S. Devices used for storing values are called word devices, e.g. T, C, D, E and F. Although bit device can only be ON/OFF for a single point, they can also be used as numeric values in the operands of instructions if the data type declaration device Kn is added in front of the bit device.
2. For 16-bit data, K1~K4 are applicable. For 32-bit data, K1~K8 are applicable. For example, K2M0 refers to a 8-bit value composed of M0 ~ M7.



3. Transmit K1M0, K2M0, K3M0 to 16-bit registers. Only the valid bit data will be transmitted and the upper bits in the 16-bit register will all be filled with 0. The same rule applies when sending K1M0, K2M0, K3M0, K4M0, K5M0, K6M0, K7M0 to 32-bit registers.
4. When the Kn value is specified as K1~K3 (K4~K7) for a 16-bit (32-bit) operation, the empty upper bits of the target register will be filled with "0." Therefore, the operation result in this case is positive since the MSB(Most significant bit) is 0.

The BCD value combined by X0 to X7 will be converted to D0 as BIN value.

Assign Continuous Bit Numbers

As already explained, bit devices can be grouped into 4 bit units. The "n" in Kn defines the number of groups of 4 bits to be combined for data operation. For data register D, consecutive D refers to D0, D1, D2, D3, D4…; For bit devices with Kn, consecutive No. refers to:

K1X0	K1X4	K1X10	K1X14…
K2Y0	K2Y10	K2Y20	Y2X30…
K3M0	K3M12	K3M24	K3M36…
K4S0	K4S16	K4S32	K4S48…

Note: To avoid errors, please do not skip over the continuous numbers. In additoin, when K4Y0 is used in 32-bit operation, the upper 16-bit is defined as 0. Therefore, it is recommended to use K8Y0 in 32bit operation.

Floating Point Operation

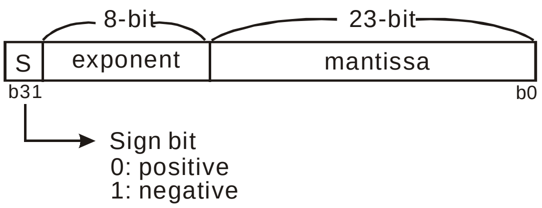
The operations in DVP-PLC are conducted in BIN integers. When the integer performs division, e.g. 40 ÷ 3 = 13, the remainder will be 1. When the integer performs square root operations, the decimal point will be left out. To obtain the operation result with decimal point, please use floating point instructions.

Application instructions revelant to floating point:

FLT	DECMP	DEZCP	DMOVR	DRAD
DDEG	DEBCD	DEBIN	DEADD	DESUB
DEMUL	DEDIV	DEXP	DLN	DLOG
DESQR	DPOW	INT	DSIN	DCOS
DTAN	DASIN	DACOS	DATAN	DADDR
DSUBR	DMULR	DDIVR	FLD※	FAND※
FOR※				

Binary Floating Point

DVP-PLC represents floating point value in 32 bits, following the IEEE754 standard:



Equation $(-1)^S \times 2^{E-B} \times 1.M ; B = 127$

Therefore, the range of 32-bit floating point value is from $\pm 2^{-126}$ to $\pm 2^{+128}$, i.e. from $\pm 1.1755 \times 10^{-38}$ to $\pm 3.4028 \times 10^{+38}$.

Example 1: Represent "23" in 32-bit floating point value

Step 1: Convert "23" into a binary value: 23.0 = 10111

Step 2: Normalize the binary value: 10111 = 1.0111 × 2^4, in which 0111 is mantissa and 4 is exponent.

Step 3: Obtain the exponent: ∵ E – B = 4 → E – 127 = 4 ∴ E = 131 = 10000011_2

Step 4: Combine the sign bit, exponent and mantissa into a floating point

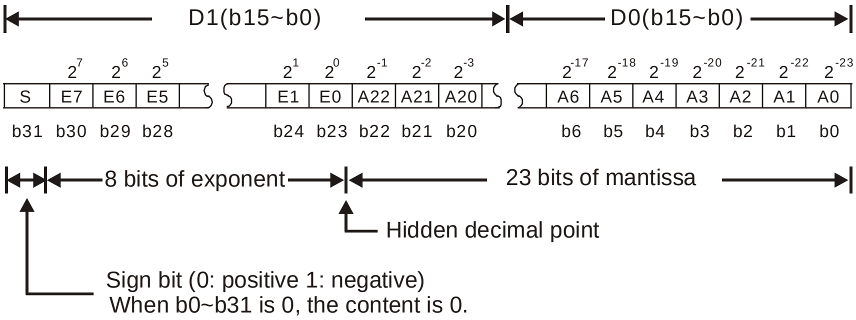
0 100**0001**1 011**1000**0000**0000**0000**0000**$_2$ = $41B80000_{16}$

Example 2: Represent "-23.0" in 32-bit floating point value

The steps required are the same as those in Example 1 and only differs in modifying the sign bit into "1".

1 100**0001**1 011**1000**0000**0000**0000**0000**$_2$=$C1B80000_{16}$

DVP-PLC uses registers of 2 continuous No. to store a 32-bit floating point value. For example, we use registers (D1, D0) for storing a binary floating point value as below:



Decimal Floating Point

- Since the binary floating point value is not very user-friendly, we can convert it into a decimal floating point value for use. However, please note that the floating point operation in DVP-PLC is still operated in binary floating point format.
- The decimal floating point is represented by 2 continuous registers. The register of smaller number is for the constant while the register of bigger number is for the exponent.

Example: Store a decimal floating point in registers (D1, D0)

Decimal floating point = [constant D0] × 10 $^{[\text{exponent D1}]}$

Constant D0 = ±1,000 ~ ±9,999

Exponent D1 = -41 ~ +35

The constant 100 does not exist in D0 because 100 is represented as 1,000 × 10^{-1}. The range of decimal floating point is ±1175 × 10^{-41} ~ ±3402×10^{+35}.

- The decimal floating point can be used in the following instructions:

  D EBCD: Convert binary floating point to decimal floating point

D EBIN: Convert decimal floating point to binary floating point

- Zero flag (M1020), borrow flag (M1021), carry flag (M1022) and the floating point operation instruction

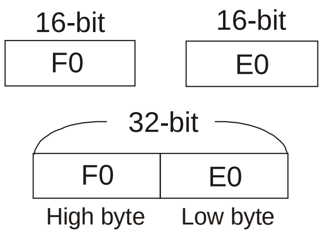
  Zero flag: M1020 = On if the operational result is “0”.

  Borrow flag: M1021 = On if the operational result exceeds the minimum unit.

  Carry flag: M1022 = On if the absolute value of the operational result exceeds the range of use.

Index register E, F

The index registers are 16-bit registers. There are 16 devices including E0 ~ E7 and F0 ~ F7.



- E and F index registers are 16-bit data registers which can be read and written.
- If you need a 32-bit register, you have to designate E. In this case, F will be covered up by E and cannot be used; otherwise, the contents in E may become incorrect. (We recommend you use MOVP instruction to reset the contents in D to 0 when the PLC is switched on.)
- Combination of E and F when you designate a 32-bit index register: (E0, F0), (E1, F1), (E2, F2), … (E7, F7)

Devices modifiable: P, X, Y, M, S, KnX, KnY, KnM, KnS, T, C, D.

E and F can modify the devices listed above but cannot modify themselves and Kn., e.g. K4M0E0 is valid and K0E0M0 is invalid. Grey columns in the table of operand at the beginning page of each application instruction indicate the operands modifiable by E and F.

If you need to modify device P, I, X, Y, M, S, KnX, KnY, KnM, KnS, T, C and D by applying E, F, you have to select a 16-bit register, i.e. you can designate E or F.

3.6 Detailed Instruction Explanation

3.6.1 Loop Control

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
00	CJ	-	✓	Conditional jump	✓	✓	✓	✓	3	-
01	CALL	-	✓	Call subroutine	✓	✓	✓	✓	3	-
02	SRET	-	-	Subroutine return	✓	✓	✓	✓	1	-
03	IRET	-	-	Interrupt return	✓	✓	✓	✓	1	-
04	EI	-	-	Enable interrupt	✓	✓	✓	✓	1	-
05	DI	-	-	Disable interrupt	✓	✓	✓	✓	1	-
06	FEND	-	-	The end of the main program (First end)	✓	✓	✓	✓	1	-
07	WDT	-	✓	Watchdog timer refresh	✓	✓	✓	✓	1	-
08	FOR	-	-	Start of a For-Next Loop	✓	✓	✓	✓	3	-
09	NEXT	-	-	End of a For-Next Loop	✓	✓	✓	✓	1	-

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
00		CJ	P	S	Conditional Jump				

OP	Range	Program Steps
S	P0~P255	CJ, CJP: 3 steps

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: The destination pointer P of the conditional jump.

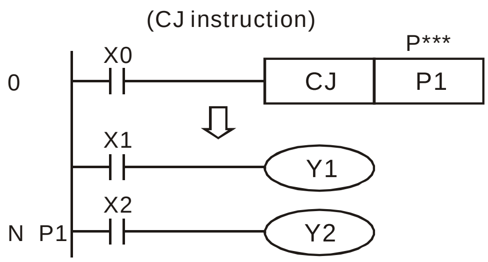
Explanations:

1. If users need to skip a particular part of PLC program in order to shorten the scan time and execute dual outputs, CJ instruction or CJP instruction can be adopted.
2. When the program designated by pointer P is prior to CJ instruction, WDT timeout will occur and PLC will stop running. Please use it carefully.
3. CJ instruction can designate the same pointer P repeatedly. However, CJ and CALL cannot designate the same pointer P; otherwise operation error will occur
4. Actions of all devices while conditional jump is being executed:
   a) Y, M and S remain their previous status before the conditional jump takes place.
   b) 10ms and 100ms timer that is executing stops.
   c) Timer T192 ~ T199 that execute the subroutine program will continue and the output contact executes normally.
   d) The high-speed counter that is executing the counting continues counting and the output contact executes normally.
   e) General counters stop executing.
   f) If timer is reset before CJ instruction executes, the timer will still be in the reset status while CJ instruction is being executed.
   g) The application instructions that are being executed, i.e. DHSCS, DHSCR, DHSZ, SPD, PLSY, PWM, PLSR, PLSV, DRVI, DRVA, continue being executed.

Program example 1:

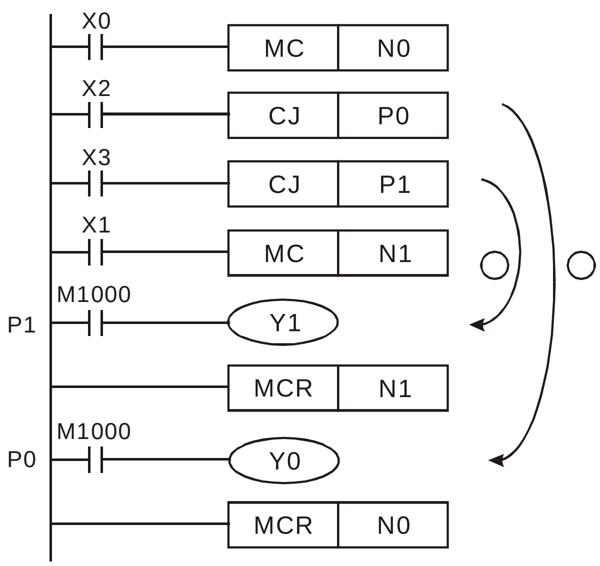
When X0 = ON, the program will skip from address 0 to N (Pointer P1) automatically and keep on executing. Instructions between address 0 and N will be skipped..

When X0 = OFF, program flow will proceed with the row immediately after the CJ instruction.



Program example 2:

1. The instruction CJ between the instruction MC and the instruction MCR can be used in the five conditions below.
   a). The execution of the program jumps from the part of the program outside one MC/MCR loop to the part of the program outside another MC/MCR loop.
   b). The execution of the program jumps from the part of the program outside the MC/MCR loop to the part of the program inside the MC/MCR loop.
   c). The execution of the program jumps from the part of the program inside the MC/MCR loop to the part of the program inside the MC/MCR loop.
   d). The execution of the program jumps from the part of the program inside the MC/MCR loop to the part of the program outside the MC/MCR loop.
   e). The execution of the program jumps from the part of the program inside one the MC/MCR loop to the part of the program inside another the MC/MCR loop.



2. When the instruction MC is executed, the previous state of the switch contact is put onto the top of the stack inside the PLC. The stack is controlled by the PLC, and can not be changed by users. When the instruction MCR is executed, the previous state of the switch contact is popped from the top of the stack. Under the conditions listed in (b), (d), and (e) above, the number of times the items are pushed onto the stack may be different from the number of times the items are popped from the stack. When this situation occurs, at most 32 items can be pushed onto the stack, and the items can be popped form the stack until the stack is empty. Therefore, when CJ or CJP is used with MC and MCR, users have to be careful of the pushing of the item onto the stack and the popping of the item from the stack.

Program example 3:

The table explains the device status in the ladder diagram below.

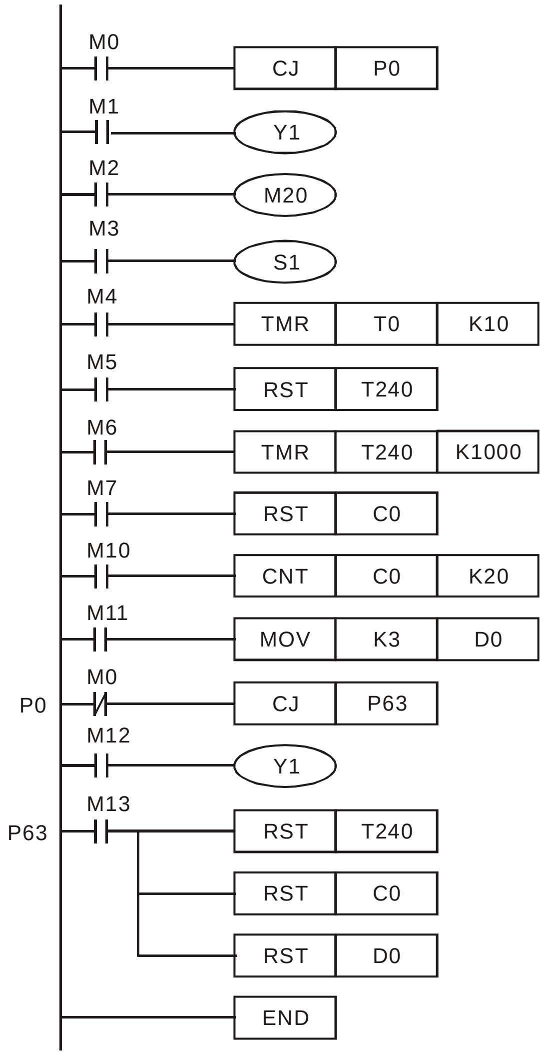
Device	Contact state before CJ execution	Contact state during CJ execution	Output coil state during CJ execution
Y, M, S	M1, M2, M3 OFF	M1, M2, M3 OFF→ON	Y1 *1, M20, S1 OFF
	M1, M2, M3 ON	M1, M2, M3 ON→OFF	Y1 *1, M20, S1 ON
10ms, 100ms Timer*2	M4 OFF	M4 OFF→ON	Timer is not activated
	M4 ON	M4 ON→OFF	Timer T0 immediately stops and is latched. When M0 ON → OFF, T0 will be reset.
1ms,10ms, 100ms accumulative Timer	M6 OFF	M6 OFF→ON	Timer T240 is not activated
	M6 ON	M6 ON→OFF	Timer T240 immediately stops and is latched. When M0 ON → OFF, T240 will still be latched.
C0~C234 *3	M7, M10 OFF	M10 is ON/OFF triggered	Counter C0 stops
	M7 OFF, M10 is ON/OFF triggered	M10 is ON/OFF triggered	Counter C0 stops and latched. When M0 is OFF, C0 resumes counting.
Application instruction	M11 OFF	M11 OFF→ON	Application instructions will not be executed.
	M11 ON	M11 ON→OFF	The skipped application instruction will not be executed but API 53~59, API 157~159 keep executing.

*1: Y1 is dual output. When M0 is OFF, it is controlled by M1. When M0 is ON, M12 will control Y1

*2: When timer that subroutine used (T184~T199) executes first and then CJ instruction is executed, the timer will keep counting. After the timer reaches the set value, output contact of timer will be ON.

*3: When high-speed counters (C235~C254) executes first and then CJ instruction is executed, the counter will keep counting and its associated output status remains.

Y1 is a dual output. When M0 = OFF, Y1 is controlled by M1. M0 = ON, Y1 is controlled by M12.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
01		CALL	P	S	Call Subroutine				

OP	Valid Range	Program Steps
S	P0~P255	CALL, CALLP: 3 steps

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: The destination pointer P of the call subroutine.

Explanations:

1. When the CALL instruction is active it forces the program to run the subroutine associated with the called pointer.
2. A CALL instruction must be used in conjunction with FEND (API 06) and SRET (API 02) instructions.
3. The program jumps to the subroutine pointer (located after an FEND instruction) and processes the contents until an SRET instruction is encountered. This forces the program flow back to the line of ladder immediately following the original CALL instruction.

Points to note:

1. Subroutines must be placed after FEND instruction.
2. Subroutines must end with SRET instruction.
3. CALL pointers and CJ instruction pointers are not allowed to coincide.
4. CALL instructions can call the same CALL subroutine any number of times.
5. Subroutines can be nested 5 levels including the initial CALL instruction. (If entering the six levels, the subroutine won't be executed.)

API	Mnemonic	Function	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
02	SRET	Subroutine Return				

OP	Descriptions	Program Steps
N/A	No contact to drive the instruction is required Automatically returns program execution to the address after CALL instruction in O100.	SRET: 1 step

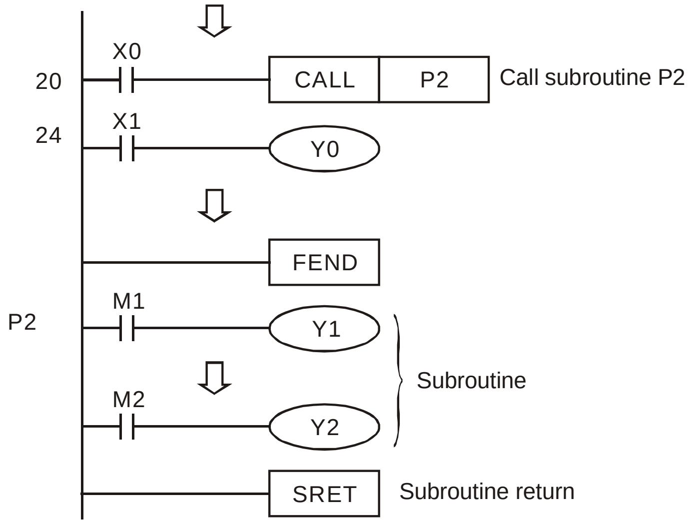
PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Explanations:

SRET indicates the end of subroutine program. The subroutine will return to main program and begin execution with the instruction after the CALL instruction.

Program example 1:

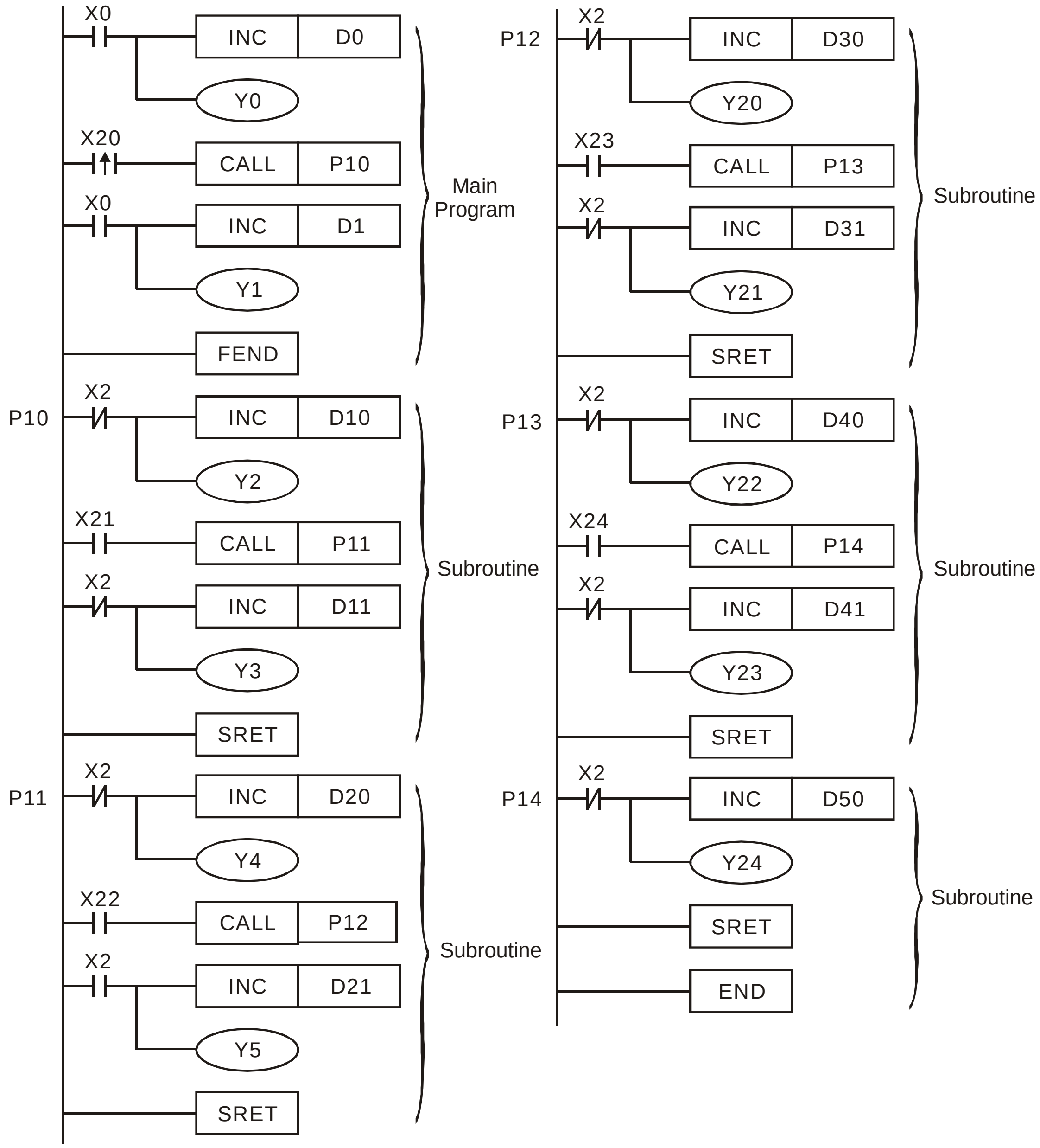
When X0 = ON, the CALL instruction will jump to P2 and run the subroutine. With the execution of the SRET instruction, it will jump back to address 24 and continue the execution.



Program example 2:

1. When the rising-edge of X20 is triggered, CALL P10 instruction will transfer execution to subroutine P10.
2. When X11 is ON, execute CALL P11, jump to and run subroutine P11.
3. When X12 is ON, execute CALL P12, jump to and run subroutine P12.
4. When X13 is ON, execute CALL P13, jump to and run subroutine P13.
5. When X14 is ON, execute CALL P14, jump to and run subroutine P14. When the SRET instruction is reached, jump back to the last P subroutine to finish the remaining instructions.

6. The execution of subroutines will go backwards to the subroutine of upper level until SRET instruction in P10 subroutine is executed. After this program execution will return to the main program.



API	Mnemonic	Function	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
03	IRET	Interrupt Return				

OP	Descriptions	Program Steps
N/A	No contact to drive the instruction is required. IRET ends the processing of an interrupt subroutine and returns execution back to the main program	IRET: 1 step

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

API	Mnemonic	Function	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
04	EI	Enable Interrupt				

OP	Descriptions	Program Steps
N/A	No contact to drive the instruction is required. Enables Interrupts, explanation of this instruction also coincides with the explanation of the DI (disable interrupts instruction), see the DI instruction for more information. M1050~M1059	EI: 1 step

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

API	Mnemonic	Function	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
05	DI	Disable Interrupt				

OP	Descriptions	Program Steps
N/A	No contact to drive the instruction is required. DI instruction disables PLC to accept interrupts. When the special auxiliary relay M1050 ~ M1059 for disabling interruption is driven, the corresponding interruption request will not be executed even in the range allowed for interruptions.	DI: 1 step

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Explanations:

1. EI instruction allows interrupting subroutine in the program, e.g. external interruption, timer interruption, and high-speed counter interruption.
2. In the program, interruption subroutines are enabled between EI and DI instructions. If there is no section requires to be interrupt-disabled, DI instruction can be omitted.
3. Interrupt subroutines must be placed after the FEND instruction.
4. Other interrupts are not allowed during execution of a current interrupt routine.
5. When many interruptions occur, the priority is given to the firstly executed interruption. If several interruptions occur at the same time, the priority is given to the interruption with the smaller pointer No.
6. Any interrupt request occurring between DI and EI instructions will not be executed immediately. The interrupt will be memorized and executed when the next EI occurs.
7. When using the interruption pointer, DO NOT repeatedly use the high-speed counter driven by the same X input contact.
8. When immediate I/O is required during the interruption, write REF instruction in the program to update the state of I/O

Points to note:

Interrupt pointers (I):

a) External interrupts: 8 points including (I000/I001, X0), (I100/I101, X1), (I200/I201, X2), (I300/I301, X3), (I400/I401, X4), (I500/I501, X5), (I600/I601, X6) and (I700/I701, X7) (00 designates interruption in falling-edge, 01 designates interruption in rising-edge)

Timer interrupts: 2 points including I605~I699 and I705~I799 (Timer resolution = 1ms), Timer interrupts: 1 point including I805~I899 (Timer resolution = 0.1ms), available for SE, ES2-E/EC5, for other modules, this function is available for modules with firmware V2.00 or later.

b) High-speed counter interrupts: 8 points including I010, I020, I030, I040, I050, I060, I070, and I080. (used with API 53 DHSCS instruction to generate interrupt signals)

c) Communication interrupts: 3 points including I140, I150 and I160

d) Associated flags:

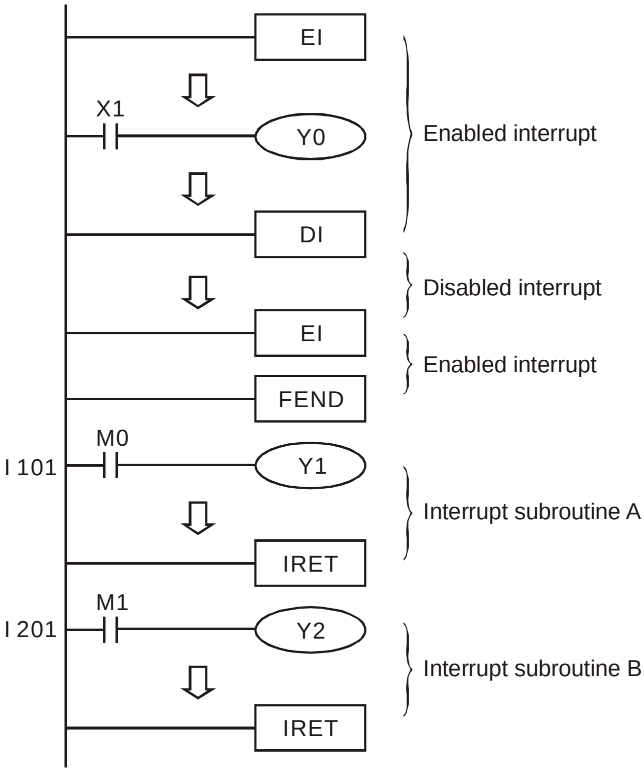
Flag	Function
M1050	Disable external interruption I000 / I001
M1051	Disable external interruption I100 / I101
M1052	Disable external interruption I200 / I201
M1053	Disable external interruption I300 / I301
M1054	Disable external interruption I400 / I401
M1055	Disable external interruption I500 / I501, I600 / I601, I700 / I701
M1056	Disable timer interrupts I605~I699
M1057	Disable timer interrupts I705~I799 and I805~I899
M1059	Disable high-speed counter interruptions I010~I080
M1280	I000/I001 Reverse interrupt trigger pulse direction (Rising/Falling)
M1284	I400/I401 Reverse interrupt trigger pulse direction (Rising/Falling)
M1286	I600/I601 Reverse interrupt trigger pulse direction (Rising/Falling)

Note: Default setting of I000(X0) is falling-edge triggered. When M1280=ON and EI is enabled, PLC will reverse X0 as rising-edge triggered. To reset X0 as falling-edge, reset M1280 first and execute DI instruction. After this, X0 will be reset as falling-edge when EI is executed again.

3

Program example:

During the PLC operation, the program scans the instructions between EI and DI, if X1 or X2 are ON, the subroutine A or B will be interruptted. When IRET is reached, the main program will resume.



EI
X1
Y0
DI
EI
FEND
M0
I 101
Y1
IRET
M1
I 201
Y2
IRET
Enabled interrupt
Disabled interrupt
Enabled interrupt
Interrupt subroutine A
Interrupt subroutine B

API	Mnemonic	Function	Controllers			
			ES2/EX2/ EC5	SS2	SA2 SE	SX2
06	FEND	The End of The Main Program (First End)				

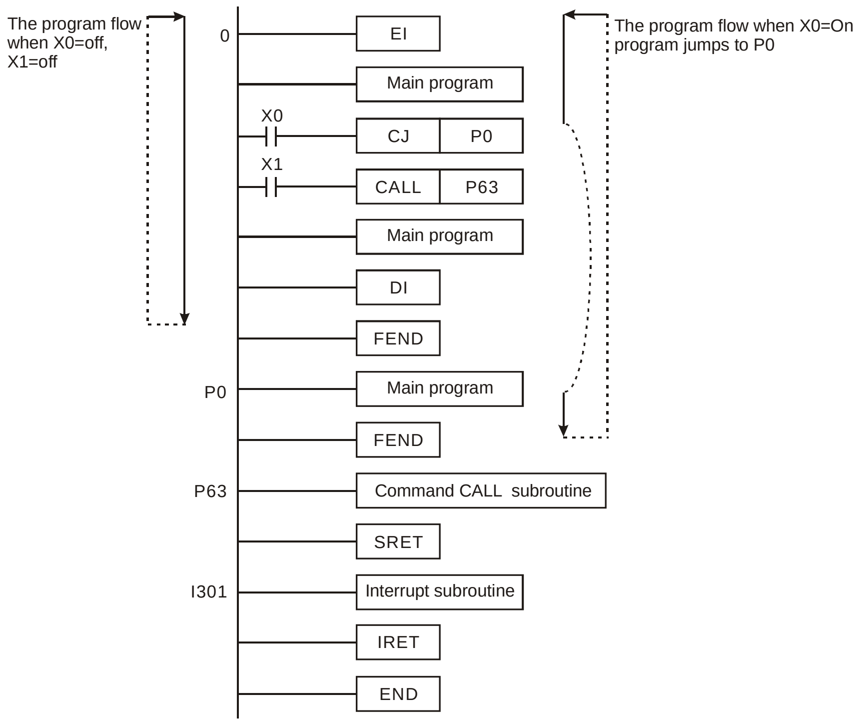
OP	Descriptions	Program Steps
N/A	No contact to drive the instruction is required.	FEND: 1 step

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

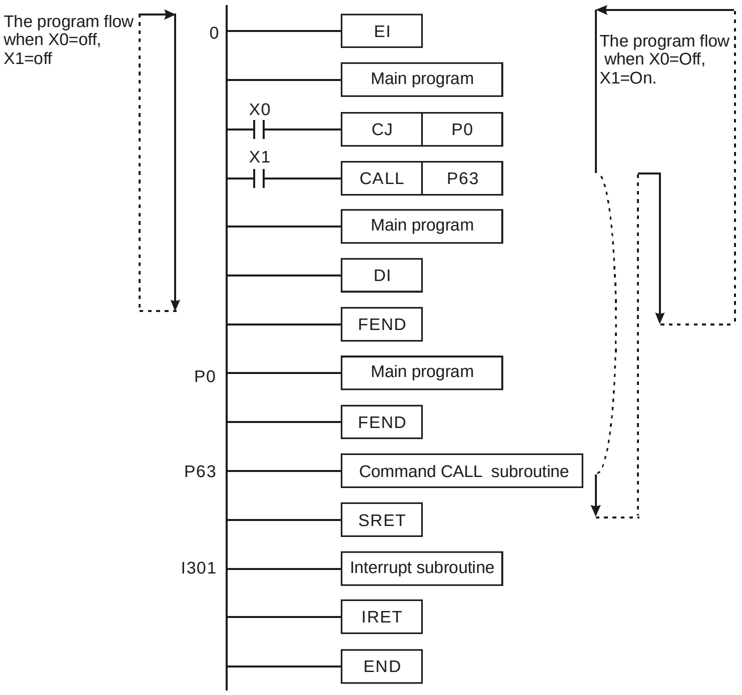
Explanations:

1. Use FEND instruction when the program uses either CALL instructions or interrupts. If no CALL instruction or interrupts are used, use END instruction to end the main program.
2. The instruction functions same as END instruction in PLC operation process.
3. CALL subroutines must be placed after the FEND instruction. Each CALL subroutine must end with the SRET instruction.
4. Interrupt subroutines must be placed after the FEND instruction. Each interrupt subroutine must end with the IRET instruction.
5. When using the FEND instruction, an END instruction is still required, but should be placed as the last instruction after the main program and all subroutines.
6. If several FEND instructions are in use, place the subroutine and interruption service programs between the final FEND and END instruction.
7. When CALL instruction is executed, executing FEND before SRET will result in errors.
8. When FOR instruction is executed, executing FEND before NEXT will result in errors.

CJ Instruction Program Flow



CALL Instruction Program Flow



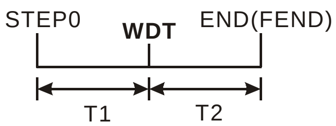
API	Mnemonic			Function	Controllers			
07		WDT	P	Watchdog Timer Refresh	ES2/EX2/ EC5	SS2	SA2 SE	SX2

OP	Descriptions	Program Steps
N/A		WDT, WDTP: 1 step

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Explanations:

1. WDT instruction can be used to reset the Watch Dog Timer. If the PLC scan time (from address 0 to END or FEND instruction) is more than 200ms, the ERROR LED will flash. In this case, users have to turn the power OFF and then ON to clear the fault. PLC will determine the state of RUN/STOP according to RUN/STOP switch. If there is no RUN/STOP switch, PLC will return to STOP status automatically.
2. Time to use WDT:
   a) When an error occurs in the PLC system.
   b) When the scan time of the program exceeds the WDT value in D1000. It can be modified by using the following two methods.
      i. Use WDT instruction



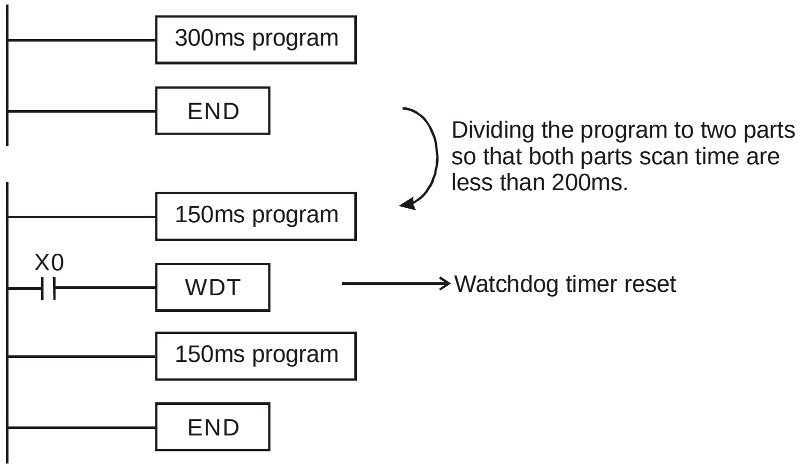
      ii. Use the set value in D1000 (Default: 200ms) to change the time for watchdog.

Points to note:

1. When the WDT instruction is used it will operate on every program scan as long as its input condition has been made. To force the WDT instruction to operate for only ONE scan, users have to use the pulse (P) format of the WDT instruction, i.e. WDTP.
2. The watchdog timer has a default setting of 200ms. This time limit can be customized to users requirement by editing the content in D1000, the wathdog timer register.

Program example:

If the program scan time is over 300ms, users can divide the program into 2 parts. Insert the WDT instruction in between, making scan time of the first half and second half of the program being less than 200ms.



API	Mnemonic	Operands	Function	Controllers			
08	FOR	S	Start of a FOR-NEXT Loop	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	FOR: 3 steps
S					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: The number of times for the loop to be repeated.

API	Mnemonic	Function	Controllers			
09	NEXT	End of a FOR-NEXT Loop	ES2/EX2/ EC5	SS2	SA2 SE	SX2

OP	Descriptions	Program Steps
N/A	No contact to drive the instruction is required.	NEXT: 1 step

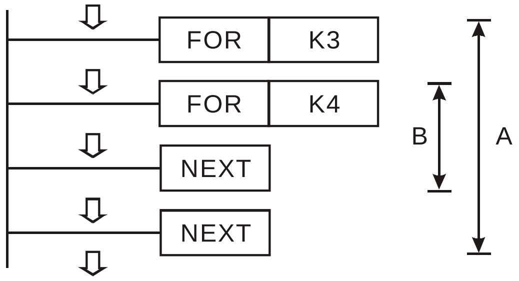
PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Explanations:

1. FOR and NEXT instructions are used when loops are needed. No contact to drive the instruction is required.
2. “N” (number of times loop is repeated) may be within the range of K1 to K32767. If the range N≦K1, N is regarded as K1.
3. An error will occur in the following conditions:
   - NEXT instruction is before FOR instruction.
   - FOR instruction exists but NEXT instruction does not exist..
   - There is a NEXT instruction after the FEND or END instruction.
   - Number of FOR instructions differs from that of NEXT instructinos.
4. FOR~NEXT loops can be nested for maximum five levels. Be careful that if there are too many loops, the increased PLC scan time may cause timeout of watchdog timer and error. Users can use WDT instruction to modify this problem.

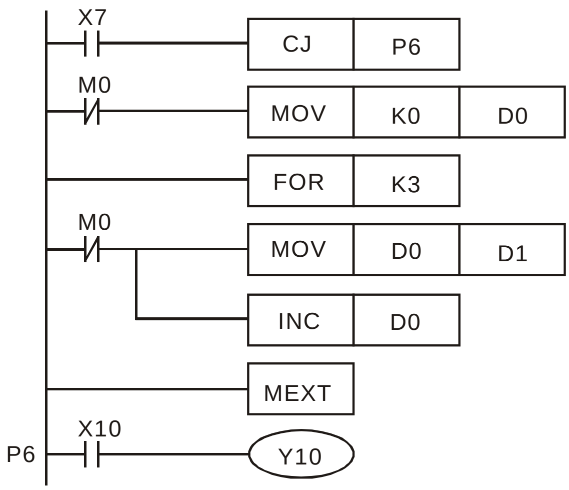
Program example 1:

After program A has been executed for 3 times, it will resume its execution after NEXT instruction. Program B will be executed for 4 times whenever program A is executed once. Therefore, program B will be executed 3 × 4 = 12 times in total.



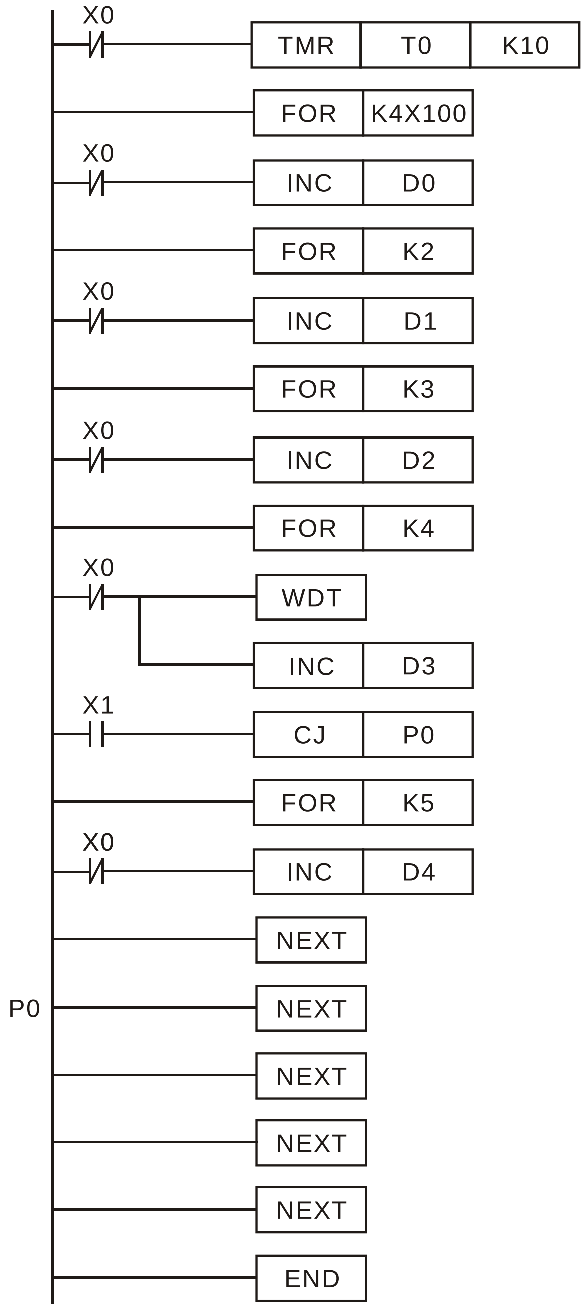
Program example 2:

When X7 = OFF, PLC will execute the program between FOR ~ NEXT. When X7 = ON, CJ instruction jumps to P6 and avoids executing the instructions between FOR ~ NEXT.



Program example 3:

Users can adopt CJ instruction to skip a specified FOR ~ NEXT loop. When X1 = ON, CJ instruction executes to skip the most inner FOR ~ NEXT loop.



3.6.2 Transmission Comparison

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
10	CMP	DCMP	✓	Compare	✓	✓	✓	✓	7	13
11	ZCP	DZCP	✓	Zone compare	✓	✓	✓	✓	9	17
12	MOV	DMOV	✓	Move	✓	✓	✓	✓	5	9
13	SMOV	-	✓	Shift move	✓	✓	✓	✓	11	-
14	CML	DCML	✓	Complement	✓	✓	✓	✓	5	9
15	BMOV	-	✓	Block move	✓	✓	✓	✓	7	-
16	FMOV	DFMOV	✓	Fill move	✓	✓	✓	✓	7	13
17	XCH	DXCH	✓	Exchange	✓	✓	✓	✓	5	9
18	BCD	DBCD	✓	Convert BIN to BCD	✓	✓	✓	✓	5	9
19	BIN	DBIN	✓	Convert BCD to BIN	✓	✓	✓	✓	5	9
343	MOV	DMOV	✓	Move	✓	✓	✓	✓	5	9

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
10	D	CMP	P	S₁ S₂ D	Compare				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*	*	CMP, CMPP: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DCMP, DCMPP: 13 steps
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

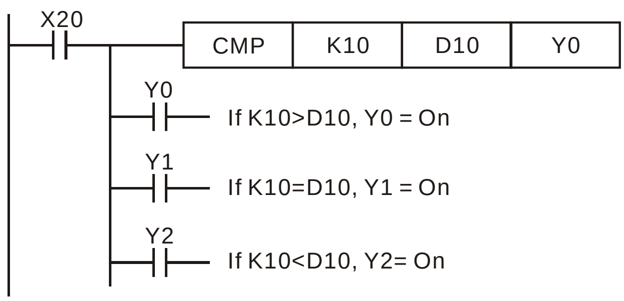
S_1: Comparison Value 1 **S_2**: Comparison Value 2 **D**: Comparison result

Explanations:

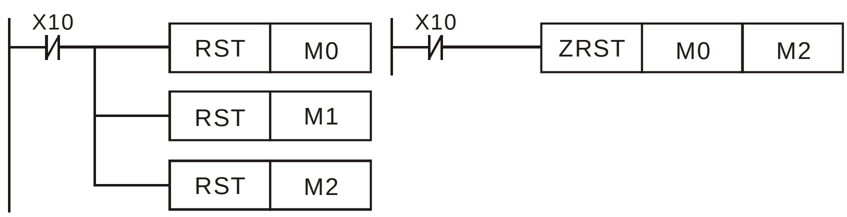
1. The contents of **S_1** and **S_2** are compared and **D** stores the comparison result.
2. The comparison values are signed binary values. If b15=1 in 16-bit instruction or b31=1 in 32-bit instruction, the comparison will regard the value as a negative binary value.
3. Operand **D** occupies 3 continuous devices. **D**, **D** +1, **D** +2 hold the comparison results, **D** = ON if **S_1** > **S_2**, **D** +1 = ON if **S_1** = **S_2**, **D** +2 = ON if **S_1** < **S_2**
4. If operand **S_1**, **S_2** use index register F, only 16-bit instruction is available.

Program example:

1. If **D** is set as Y0, then Y0, Y1, Y2 will display the comparison results as shown below.
2. When X20 = ON, CMP instruction is executed and one of Y0, Y1, Y2 will be ON. When X20 = OFF, CMP instruction is not executed and Y0, Y1, Y2 remain in their previous condition.



3. Use RST or ZRST instruction to reset the comparison result.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
11	D	ZCP	P	S1 S2 S D	Zone Compare				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ZCP, ZCPP: 9 steps DZCP, DZCPP: 17 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	
S_2					*	*	*	*	*	*	*	*	*	*	*	
S					*	*	*	*	*	*	*	*	*	*	*	
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

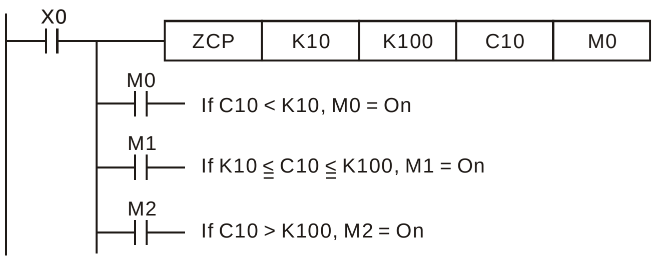
S_1: Lower bound of zone comparison **S_2**: Upper bound of zone comparison **S**: Comparison value **D**: Comparison result

Explanations:

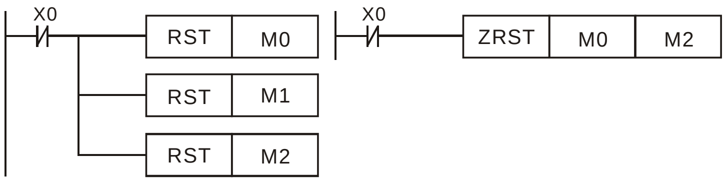
1. **S** is compared with its lower bound **S_1** and upper bound **S_2**. **D** stores the comparison results.
2. The comparison values are signed binary values. If b15=1 in 16-bit instruction or b31=1 in 32-bit instruction, the comparison will regard the value as a negative binary value.
3. Operand **S_1** should be smaller than operand **S_2**. When **S_1** > **S_2**, the instruction takes **S_1** as the 1st comparison value and performs normal comparison similar to CMP instruction.
4. If operand **S_1**, **S_2** , and **S** use index register F, only 16-bit instruction is available.
5. Operand **D** occupies 3 continuous devices. **D**, **D** +1, **D** +2 hold the comparison results, **D** = ON if **S_1** > **S**, **D** +1 = ON if **S_1** ≦ **S** ≦ **S_2**, **D** +2 = ON if **S_2** < **S**

Program example:

1. If **D** is set as M0, then M0, M1, M2 will work as the program example below.
2. When X0 = ON, ZCP instruction is driven and one of M0, M1, M2 is ON. When X0 = OFF, ZCP instruction is not driven and M0, M1, M2 remain in the previous status.



3. Use RST or ZRST instruction to reset the comparison result.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
12	D	MOV	P	(S) (D)	Move				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MOV, MOVP: 5 steps
S					*	*	*	*	*	*	*	*	*	*	*	DMOV, DMOVP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

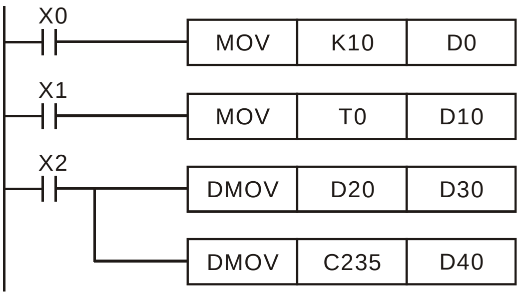
S: Source of data **D**: Destination of data

Explanations:

1. When this instruction is executed, the content of **S** will be moved directly to **D**. When this instruction is not executed, the content of **D** remains unchanged
2. If operand **S** and **D** use index register F, only 16-bit instruction is applicable

Program example:

1. MOV will move a 16-bit value from the source location to the destination.
   a) When X0 = OFF, the content of D0 remains unchanged. If X0 = ON, the data in K10 is moved to D0.
   b) When X1 = OFF, the content of D10 remains unchanged. If X1 = ON, the data of T0 is moved to D10 data register.
2. DMOV will move a 32-bit value from the source location to the destination.
   a) When X2 = OFF, the content of (D31, D30) and (D41, D40) remain unchanged.
   b) When X2 = ON, the data of (D21, D20) is moved to (D31, D30) data register. Meanwhile, the data of C235 is moved to (D41, D40) data register.



API	Mnemonic		Operands	Function	Controllers			
13	SMOV	P	S m1 m2 D n	Shift Move	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SMOV, SMOVP: 11 step
S							*	*	*	*	*	*	*	*	*	
m_1					*	*										
m_2					*	*										
D								*	*	*	*	*	*	*	*	
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: Source device **m_1**: Start digit to be moved from source device **m_2**: Number of digits to be moved **D**: Destination device **n**: Start digit of the destination device for the moved digits

Explanation:

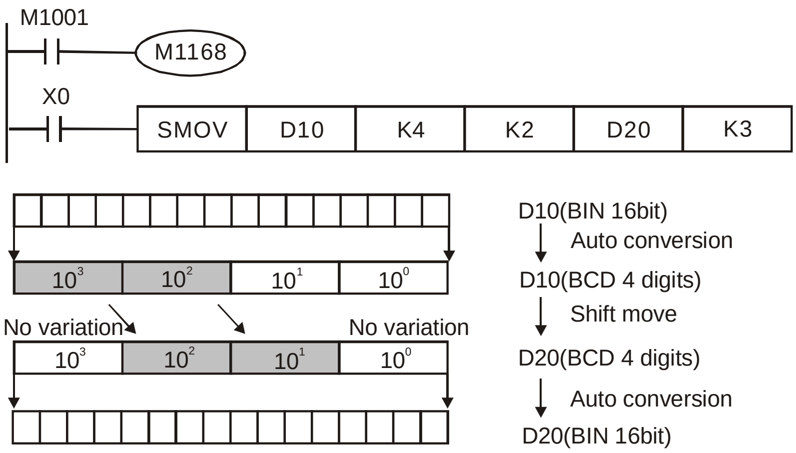
1. This instruction is able to re-allocate or combine data. When the instruction is executed, **m_2** digits of contents starting from digit **m_1** (from high digit to low digit) of **S** will be sent to **m_2** digits starting from digit **n** (from high digit to low digit) of **D**.
2. M1168 is used for designating SMOV working mode. When M1168 = ON, the instruction is in BIN mode. When M1168 = OFF, the instruction is in BCD mode.

Points to note:

1. The range of **m_1**: 1 – 4
2. The range of **m_2**: 1 – **m_1**
3. The range of **n**: **m_2** – 4

Program example 1:

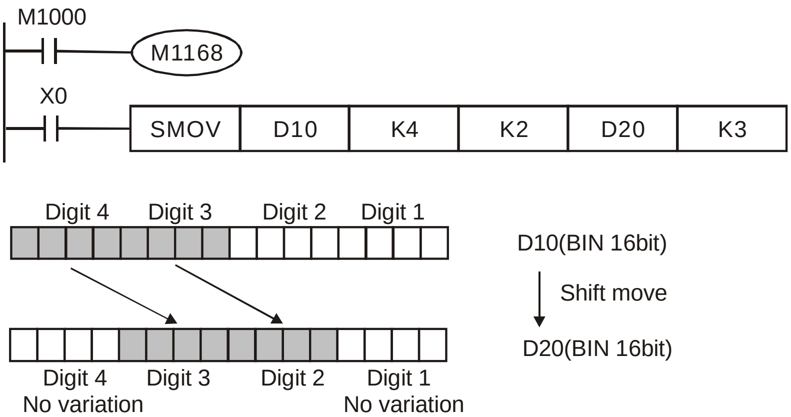
1. When M1168 = OFF (in BCD mode) and X0 = ON, the 4^{th} (thousand) and 3^{rd} (hundred) digit of the decimal value in D10 start to move to the 3^{rd} (hundred) and 2^{nd} (ten) digit of the decimal value in D20. 10^3 and 10^0 of D20 remain unchanged after this instruction is executed.
2. When the BCD value exceeds the range of 0 ~ 9,999, PLC detects an operation error and will not execute the instruction. M1067, M1068 = ON and D1067 stores the error code OE18 (hex).



If D10 = K1234, D20 = K5678 before execution, D10 remains unchanged and D20 = K5128 after execution.

Program example 2:

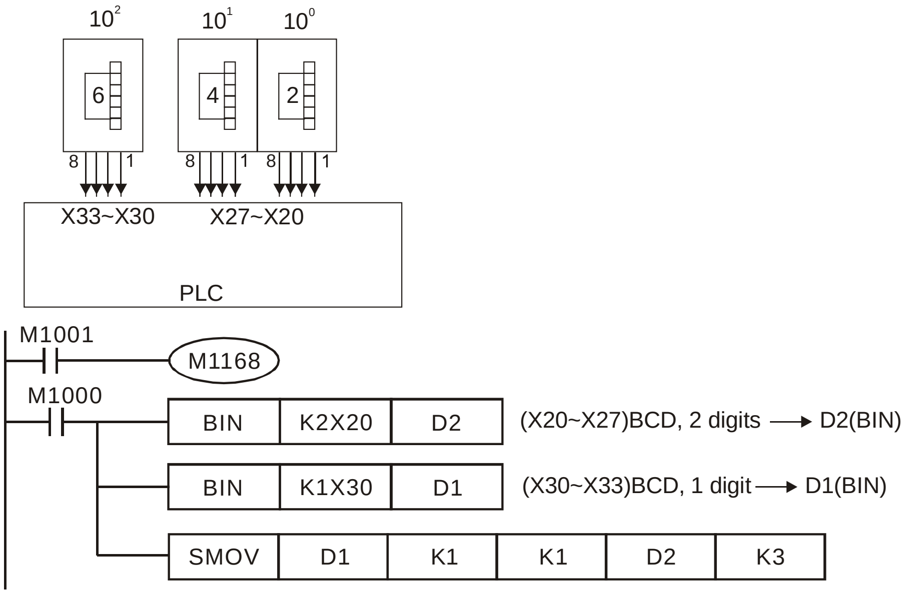
When M1168 = ON (in BIN mode) and SMOV instruction is in use, D10 and D20 will not be converted in BCD format but be moved in BIN format (4 digits as a unit).



If D10 = H1234, D20 = H5678 before execution, D10 remains unchanged and D20 = H5128 after execution.

Program example 3:

1. This instruction can be used to combine the DIP switches connected to the input terminals without continuous numbers.
2. Move the 2 digits of the right DIP switch (X27~X20) to the 2 digits of D2, and the 1 digit of the DIP switch (X33~X30) to the 1st digit of D1.
3. Use SMOV instruction to move the 1st digit of D1 to the 3rd digit of D2 and combine the values from two DIP switches into one set of value.



API	Mnemonic			Operands	Function	Controllers			
14	D	CML	P	S D	Compliment	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	CML, CMLP: 5 steps
S					*	*	*	*	*	*	*	*	*	*	*	DCML, DCMLP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

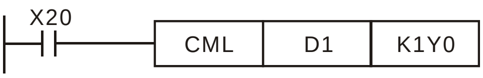
S: Source of data **D**: Destination device

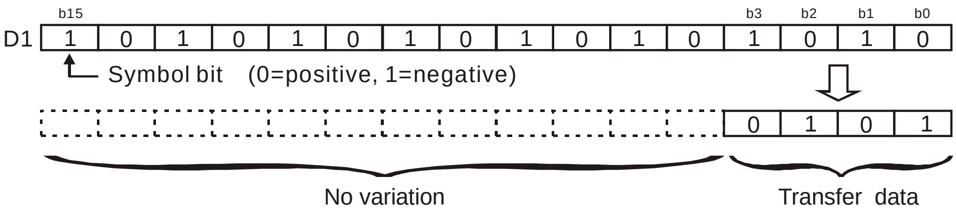
Explanations:

1. The instruction reverses the bit pattern (0→1, 1→0) of all the contents in **S** and sends the contents to **D**.
2. If operand **S** and **D** use index register F, only 16-bit instruction is available

Program example 1:

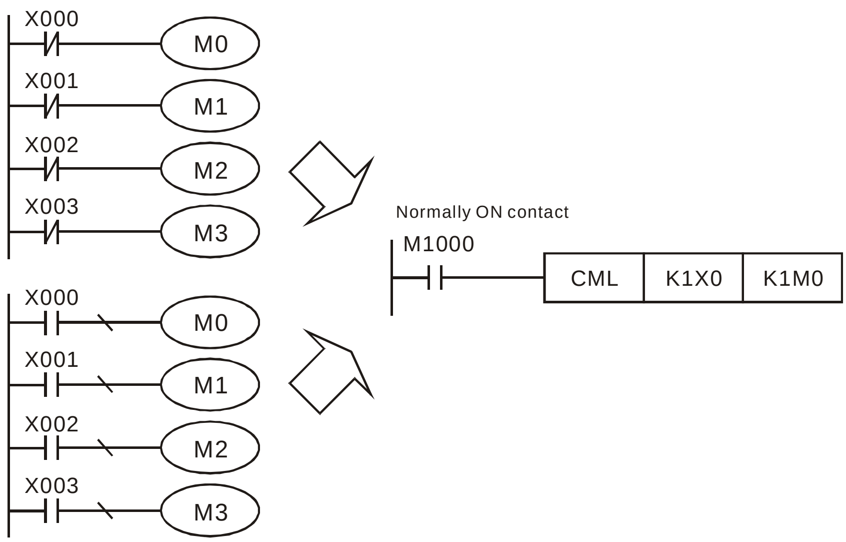
When X10 = ON, b0 ~ b3 in D1 will be inverted and sent to Y0 ~ Y3





Program example 2:

The diagram below can be substituted by the instruction on the right.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
15		BMOV	P	S D n	Block Move				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S							*	*	*	*	*	*	*			BMOV, BMOVP: 7 steps
D								*	*	*	*	*	*			
n					*	*					*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

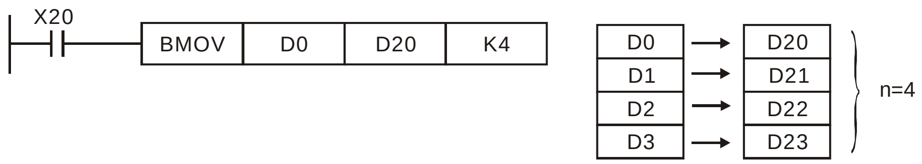
S: Start of source devices **D**: Start of destination devices **n**: Number of data to be moved

Explanations:

1. The program copies a specified block of devices to another destination. Contents in **n** registers starting from **S** will be moved to **n** registers starting from **D**. If **n** exceeds the actual number of available source devices, only the devices that fall within the valid range will be used
2. Range of **n**: 1 ~ 512.

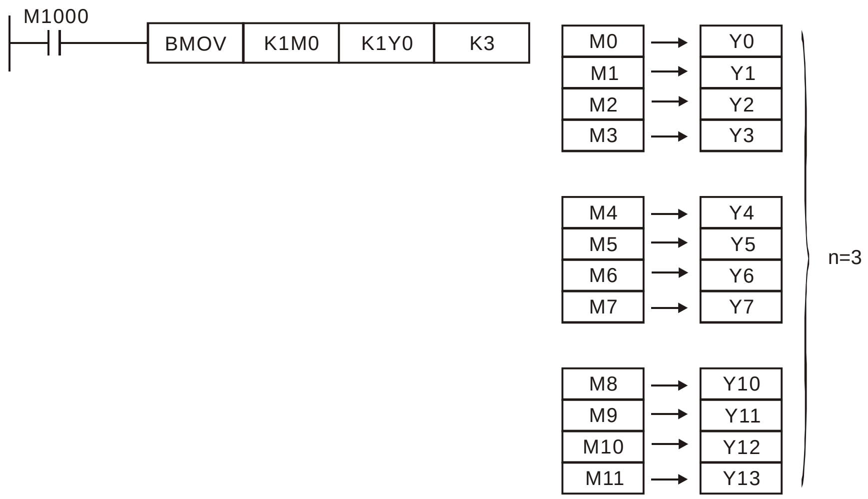
Program example 1:

When X20 = ON, the contents in registers D0 ~ D3 will be moved to the 4 registers D20 ~ D23



Program example 2:

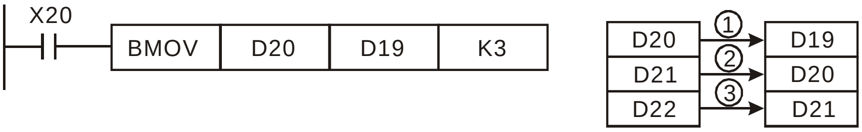
Assume the bit devices KnX, KnY, KnM and KnS are designated for moving, the number of digits of **S** and **D** has to be the same, i.e. their **n** has to be the same.



Program example 3:

In order to prevent the error which results from the overlap between the source devices and the destination devices, the data is transferred in the following way.

1. When **S** > **D**, the BMOV instruction is processed in the order ①→②→③.



2. When **S** < **D**, it is recommended to us the API37 WSFL instruction instead of BMOV.

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
16	D	FMOV	P	S D n	Fill Move				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	FMOV, FMOVP: 7 steps
S					*	*	*	*	*	*	*	*	*	*	*	DFMOV, DFMOVP: 13
D								*	*	*	*	*	*			steps
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

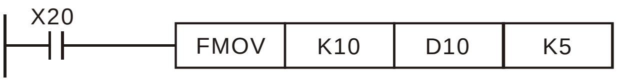
S: Source of data **D**: Destination of data **n**: Number of data to be moved

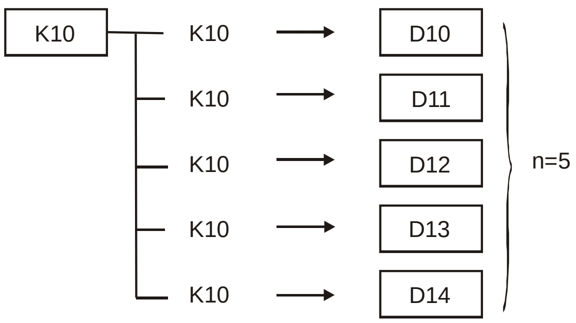
Explanations:

1. The contents in n registers starting from the device designated by **S** will be moved to n registers starting from the device designated by **D**. If n exceeds the actual number of available source devices, only the devices that fall within the valid range will be used
2. If operand **S** use index register F, only 16-bit instruction is available
3. The range of n: 1~ 512

Program example:

When X20 = ON, K10 will be moved to the 5 consecutive registers starting from D10





API	Mnemonic			Operands	Function	Controllers			
17	D	XCH	P	D1 D2	Exchange	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	XCH, XCHP: 5 steps
D_1								*	*	*	*	*	*	*	*	DXCH, DXCHP: 9 steps
D_2								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

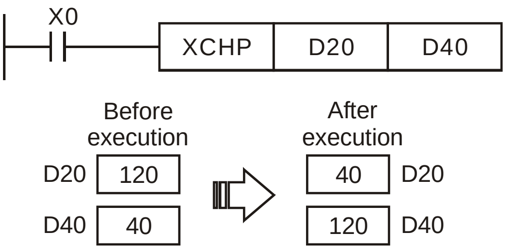
D_1: Device to be exchanged 1 **D_2**: Device to be exchanged 2

Explanations:

1. The contents in the devices designated by **D_1** and **D_2** will exchange
2. It is better to apply a pulse execution for this instruction (XCHP).
3. If operand **D1** and **D2** use index register F, only 16-bit instruction is available.

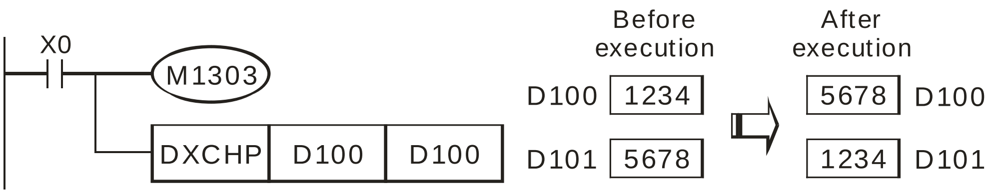
Program example:

When X0=OFF→ON, the contents of D20 and D40 exchange with each other.

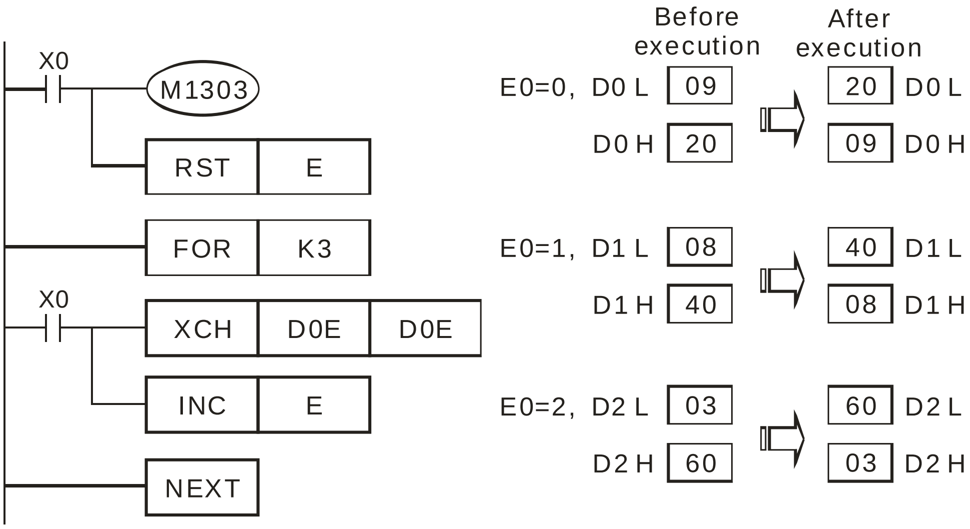


Points to note:

1. As a 16-bit instruction, when the devices designated by **D_1** and **D_2** are the same and M1303 = ON, the upper and lower 8 bits of the designated devices exchange with each other.
2. As a 32-bit instruction, when the devices designated by **D_1** and **D_2** are the same and M1303 = ON, the upper and lower 16 bits in the designated device exchange with each other.
3. When X0 = ON and M1303 = ON, 16-bit contents in D100 and those in D101 will exchange with each other.



4. When X0 = ON and M1303 = ON, the high 8 bits and the low 8 bits in D0 are exchanged, the high 8 bits and the low 8 bits in D1 are exchanged., and the high 8 bits and the low 8 bits in D2 are exchanged.



API	Mnemonic			Operands		Function	Controllers			
18	D	BCD	P	S	D	Convert BIN to BCD	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BCD, BCDP: 5 steps
S							*	*	*	*	*	*	*	*	*	DBCD, DBCDP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: Source of data **D**: Conversion result

Explanations:

1. The content in **S** (BIN value) is converted into BCD value and stored in **D**
2. As a 16-bit (32-bit) instruction, when the conversion result exceeds the range of 0 ~ 9,999 (0 ~ 99,999,999), and M1067, M1068 = ON, D1067 will record the error code 0E18 (hex)
3. If operand **S** and **D** use index register F, only 16-bit instruction is available.
4. Flags: M1067 (Program execution error), M1068 (Execution error locked), D1067 (error code)

Program example:

1. When X0 = ON, the binary value of D10 will be converted into BCD value, and the 1s digit of the conversion result will be stored in K1Y0 (Y0 ~ Y3, the 4 bit devices).

2. If D10=001E (Hex) = 0030 (decimal), the result will be Y0~Y3 = 0000(BIN).

API	Mnemonic			Operands	Function	Controllers			
19	D	BIN	P	(S) (D)	Convert BCD to BIN	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BIN, BINP: 5 steps
S							*	*	*	*	*	*	*	*	*	DBIN, DBINP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

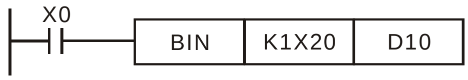
S: Source of data **D**: Conversion result

Explanations:

1. The content in **S** (BCD value) is converted into BIN value and stored in **D**.
2. The valid range of source **S**: BCD (0 to 9,999), DBCD (0 to 99,999,999)
3. If the content of **S** is not a valid BCD value, an operation error will occur, error flags M1067 and M1068 = ON, and D1067 holds error code H0E18.
4. If operand S and D use index register F, only 16-bit instruction is available.
5. Flags: M1067 (Program execution error), M1068 (Execution error locked), D1067 (error code)

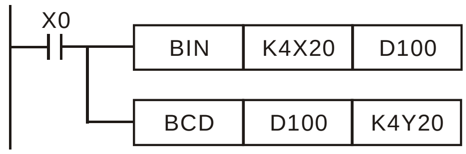
Program example:

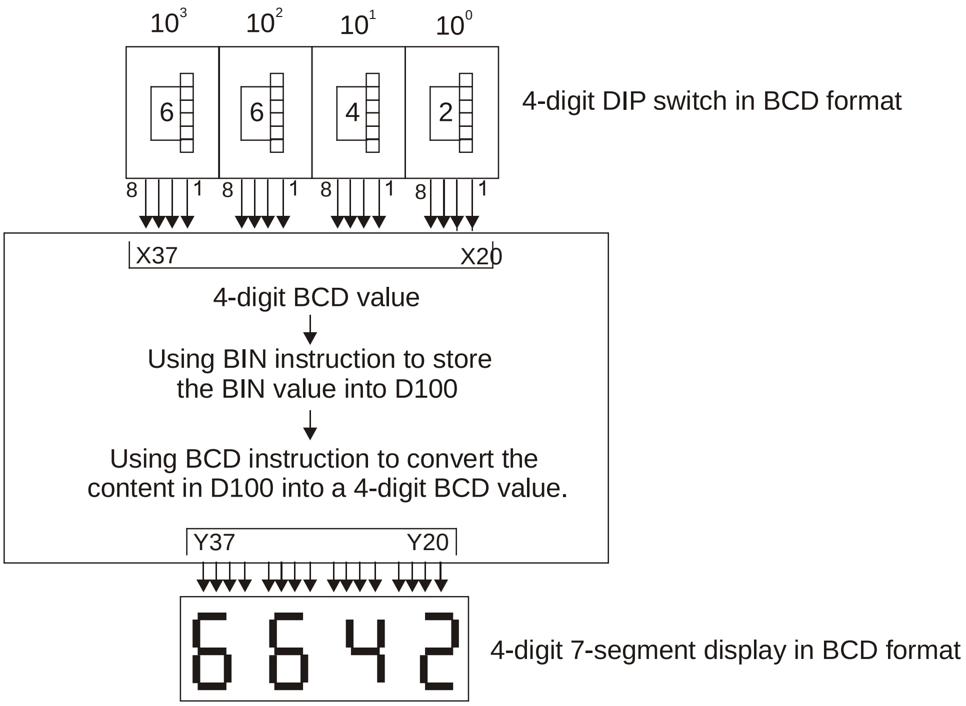
When X0 = ON, the BCD value of K1M0 will be converted to BIN value and stored in D10.



Points to note:

6. When PLC needs to read an external DIP switch in BCD format, BIN instruction has to be first adopted to convert the read data into BIN value and store the data in PLC.
7. On the contrary when PLC needs to display a value on a BCD format 7-segment displayer, BCD instruction is required to convert the internal data into BCD value then sent the value to the displayer.
8. When X0 = ON, the BCD value of K4X20 is converted into BIN value and sent to D100. The BIN value of D100 will then be converted into BCD value and sent to K4Y20.




10^3
10^2
10^1
10^0
6
6
4
2
8
1
4-digit DIP switch in BCD format
X37
X20
4-digit BCD value
Using BIN instruction to store
the BIN value into D100
Using BCD instruction to convert the
content in D100 into a 4-digit BCD value.
Y37
Y20
6642
4-digit 7-segment display in BCD format

API	Mnemonic			Operands	Function	Controllers			
343		$MOV	P	S D	Transferring a string	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	$MOV, $MOVP: 5~12 steps
S											*	*	*			
D											*	*	*			

PULSE				16-bit				32-bit
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	-

Operands:

S: Data source (string can be used) **D**: Data destination

Explanations:

1. Availabilities:

Model	ES2/EX2 /ES2-C	EC5	EC5	ES2-E	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware version	V3.68	V1.00	V1.00	V1.46	V3.02	V3.62	V2.02	V2.02	V2.90

2. Strings can be used in **S**, e.g. “abcd”. 16 words of string can be used.
3. This instruction transfers the string in **S** to **D**, and adds the code 16#00 to the end of the string.
4. If the value in **S** is not in the string format ($), you need to add the ending code 16#00 in the end. Up to 256 words (including the ending code 16#00) can be stored in S.
5. When the ending code16#00 cannot be found in **S** for 256 words in a row or even beyond the device range, the instruction is not executed. And M1067/M1068=ON, the error code 16#0E1A is stored in D1067.
6. When the operand **S** is not a string and the instruction is executed, the string starting with the data in the device specified by **S** (including 16#00) is transferred to **D**. When the instruction is not executed, the data in **D** is unchanged.
7. If **D** is not sufficient to contain the string composed of the values in **S**, the instruction is not executed. And M1067/M1068=ON, the error code 16#0E1A is stored in D1067.
8. Suppose the operand **S** is not a string. When the instruction is executed and the first character in **S** is the code 16#00, 16#00 is still transferred to **D**.

9. When 16#00 appears in the low byte, the execution of the instruction is as follows.

Before the instruction is executed:

	b15~b8	b7~b0
S	16#31	16#30
S+1	16#33	16#32
S+2	16#35	16#34
S+3	16#30	16#00

	B15~b8	b7~b0
D	16#38	16#39
D+1	16#36	16#37
D+2	16#34	16#35
D+3	16#32	16#33

After the instruction is executed:

	b15~b8	b7~b0
S	16#31	16#30
S+1	16#33	16#32
S+2	16#35	16#34
S+3	16#30	16#00

	b15~b8	b7~b0
D	16#31	16#30
D+1	16#33	16#32
D+2	16#35	16#34
D+3	16#00	16#00

16#30 in the high byte is not transferred.

16#32 in the high byte turns into 16#00.

10. When 16#00 appears in the high byte, the execution of the instruction is as follows. The transfer stops when the code 16#00, leaving the remainder of **D** unchanged.

Before the instruction is executed:

	b15~b8	b7~b0
S	16#31	16#30
S+1	16#33	16#32
S+2	16#00	16#34
S+3	16#37	16#36

	b15~b8	b7~b0
D	16#38	16#39
D+1	16#36	16#37
D+2	16#34	16#35
D+3	16#32	16#33

After the instruction is executed:

	b15~b8	b7~b0
S	16#31	16#30
S+1	16#33	16#32
S+2	16#00	16#34
S+3	16#37	16#36

	b15~b8	b7~b0
D	16#31	16#30
D+1	16#33	16#32
D+2	16#00	16#34
D+3	16#32	16#33

When **S** overlaps **D** and the device number of **S** is less than the device number of **D**, the transfer of the data to **D** starts form the ending code 16#00.

Before the instruction is executed:

	b15~b8	b7~b0
D0	16#31	16#30
D1	16#33	16#32
D2	16#35	16#34
D3	16#30	16#00

⇨

	b15~b8	b7~b0
D1	16#33	16#32
D2	16#35	16#34
D3	16#30	16#00
D4	16#38	16#37

After the instruction is executed:

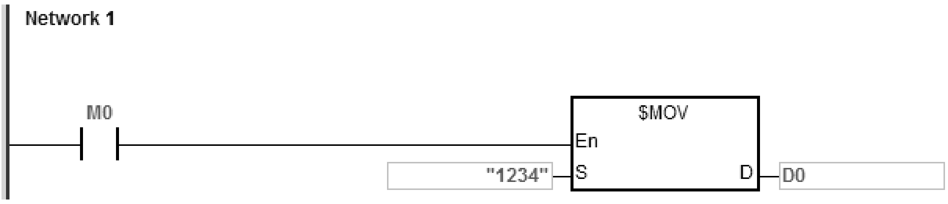
	b15~b8	b7~b0
D0	16#31	16#30
D1	16#33	16#32
D2	16#35	16#34
D3	16#30	16#00

⇨

	b15~b8	b7~b0
D1	16#31	16#30
D2	16#33	16#32
D3	16#35	16#34
D4	16#00	16#00

Example 1

Suppose the data in **S** is the string "1234" (even number of bytes). When M0 is enabled, the data 1234 and the ending code 16#00 is transferred to D0–D3 and 16#00 is added to the high byte in **D**, as follows.



The operand **S**:

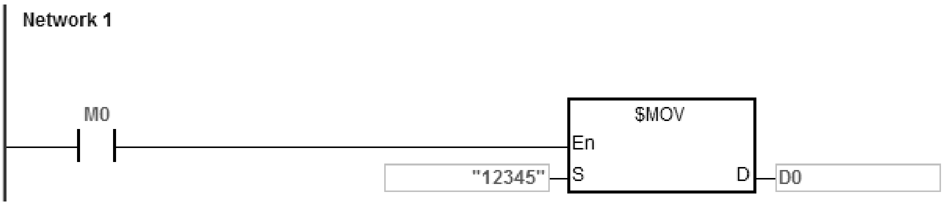
String	'1'	'2'	'3'	'4'
Hexadecimal value	16#31	16#32	16#33	16#34

After the instruction is executed, the data in **D** is as follows.

Device	High byte	Low byte	Note
D0	16#32	16#31	'1'=16#31; '2'=16#32
D1	16#34	16#33	'3'=16#33; '4'=16#34
D2	16#00	16#00	The ending code 16#00 is in the low byte. 16#00 is automatically added in the high byte.
D3	Unchanged	Unchanged	

Example 2

Suppose the data in **S** is the string “12345” (odd number of bytes). When M0 is enabled, the data 12345 is transferred to D0–D3 as follows.



The operand **S**:

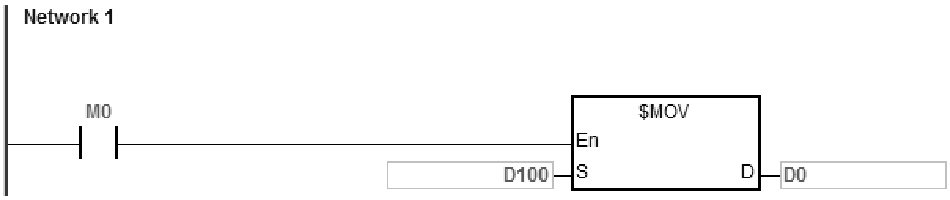
String	‘1’	‘2’	‘3’	‘4’	‘5’
Hexadecimal value	16#31	16#32	16#33	16#34	16#35

After the instruction is executed, the data in the operand **D** is as follows.

Device	**High byte**	**Low byte**	**Note**
D0	16#32	16#31	‘1’=16#31; ‘2’=16#32
D1	16#34	16#33	‘3’=16#33; ‘4’=16#34
D2	16#00	16#35	The ending code 16#00 is in the high byte.
D3	Unchanged	Unchanged	

Example 3

When the data in **S** is not a string and the ending code 16#00 appears in the low byte, the execution of the instruction is as follows.



The operand **S**:

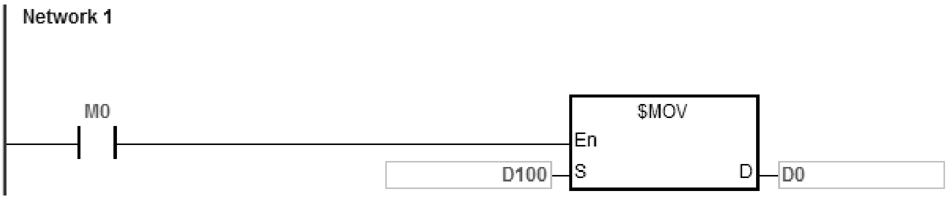
Device	High byte	Low byte	Note
D100	16#31	16#30	‘1’=16#31; ‘0’=16#30
D101	16#33	16#32	‘3’=16#33; ‘2’=16#32
D102	16#35	16#34	‘5’=16#35; ‘4’=16#34
D103	16#30	16#00	‘0’=16#30; 16#00 is the ending code.

After the instruction is executed, the data in the operand **D** is as follows.

Device	High byte	Low byte	Note
D0	16#31	16#30	‘1’=16#31; ‘0’=16#30
D1	16#33	16#32	‘3’=16#33; ‘2’=16#32
D2	16#35	16#34	‘5’=16#35; ‘4’=16#34
D3	16#00	16#00	The ending code 16#00 is in the low byte. 16#00 is automatically added in the high byte.
D4	Unchanged	Unchanged	

Example 4

When the data in **S** is not a string and the ending code 16#00 appears in the high byte, the execution of the instruction is as follows.



The operand **S**:

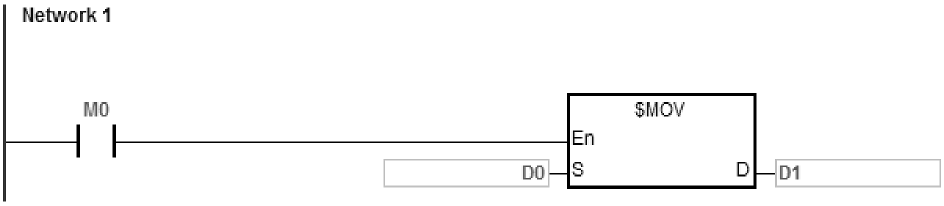
Device	High byte	Low byte	Note
D100	16#31	16#30	‘1’=16#31; ‘0’=16#30
D101	16#33	16#32	‘3’=16#33; ‘2’=16#32
D102	16#00	16#34	16#00 is the ending code. ‘4’=16#34
D103	16#37	16#36	‘7’=16#37; ‘6’=16#36

After the instruction is executed, the data in the operand **D** is as follows.

Device	High byte	Low byte	Note
D0	16#31	16#30	‘1’=16#31; ‘0’=16#30
D1	16#33	16#32	‘3’=16#33; ‘2’=16#32
D2	16#00	16#34	16#00 is the ending code. ‘4’=16#34
D3	Unchanged	Unchanged	

Example 5

When **S** overlaps **D**, and the device number of **S** is less than the device number of **D**, the transfer of the data to **D** starts from the ending code 16#00. The value in S will not be overwritten.



The operand **S**:

Device	High byte	Low byte	Note
D0	16#31	16#30	‘1’=16#31; ‘0’=16#30
D1	16#33	16#32	‘3’=16#33; ‘2’=16#32
D2	16#35	16#34	‘5’=16#35; ‘4’=16#34
D3	16#30	16#00	‘0’=16#30; 16#00 is the ending code.
D4	16#38	16#37	‘8’=16#38; ‘7’=16#37

After the instruction is executed, the data in **D** is as follows.

Device	High byte	Low byte	Note
D1	16#31	16#30	‘1’=16#31; ‘0’=16#30
D2	16#33	16#32	‘3’=16#33; ‘2’=16#32
D3	16#35	16#34	‘5’=16#35; ‘4’=16#34
D4	16#00	16#00	The ending code 16#00 is in the low byte. 16#00 is automatically added in the high byte.
D5	Unchanged	Unchanged	

3.6.3 Four Arithmetic Operations

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
20	ADD	DADD	✓	Addition	✓	✓	✓	✓	7	13
21	SUB	DSUB	✓	Subtraction	✓	✓	✓	✓	7	13
22	MUL	DMUL	✓	Multiplication	✓	✓	✓	✓	7	13
23	DIV	DDIV	✓	Division	✓	✓	✓	✓	7	13
24	INC	DINC	✓	Increment	✓	✓	✓	✓	3	5
25	DEC	DDEC	✓	Decrement	✓	✓	✓	✓	3	5
26	WAND	DAND	✓	Logical Word AND	✓	✓	✓	✓	7	13
27	WOR	DOR	✓	Logical Word OR	✓	✓	✓	✓	7	13
28	WXOR	DXOR	✓	Logical XOR	✓	✓	✓	✓	7	13
29	NEG	DNEG	✓	2’s Complement (Negation)	✓	✓	✓	✓	3	5
114	MUL16	MUL32	✓	16-bit/32-bit Binary Multiplication	✓	✓	✓	✓	7	13
115	DIV16	DIV32	✓	16-bit/32-bit Binary Division	✓	✓	✓	✓	7	13

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
20	D	ADD	P	S1 S2 D	Addition				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ADD, ADDP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DADD, DADDP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S_1: Summand **S_2**: Addend **D**: Sum

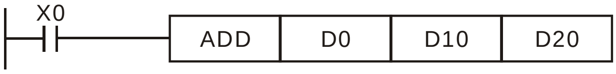
Explanations:

1. This instruction adds **S_1** and **S_2** in BIN format and store the result in **D**.
2. The most significant bit (MSB) is the sign bit of the data. 0 indicates positive and 1 indicates negative. All calculations is algebraically processed, e.g. 3 + (-9) = -6.
3. If **S_1**, **S_2** and **D** use device F, only 16-bit instruction is applicable.
4. Flags: M1020 (Zero flag), M1021 (Borrow flag), M1022 (Carry flag)

Program Example 1:

In 16-bit BIN addition:

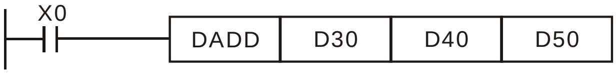
When X0 = ON, the content in D0 will plus the content in D10 and the sum will be stored in D20.



Program Example 2:

In 32-bit BIN addition:

When X0 = ON, the content in (D31, D30) will plus the content in (D41, D40) and the sum will be stored in (D51, D50). D30, D40 and D50 are low word; D31, D41 and D51 are high word



(D31, D30) + (D41, D40) = (D51, D50)

Operation of flags:

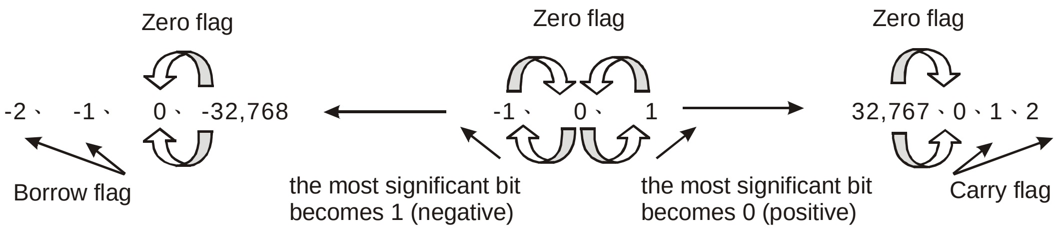
16-bit instruction:

1. If the operation result is “0”, the zero flag M1020 will be ON.
2. If the operation result exceeds -32,768, the borrow flag M1021 will be ON.
3. If the operation result exceeds 32,767, the carry flag M1022 will be ON.

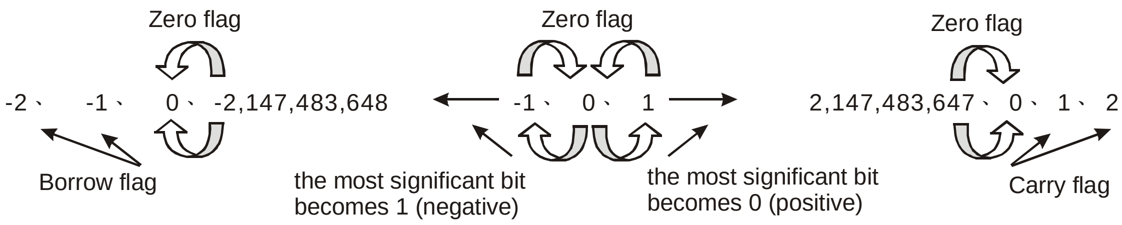
32-bit instruction:

1. If the operation result is “0”, the zero flag, M1020 will be ON.
2. If the operation result exceeds -2,147,483,648, the borrow flag M1021 will be ON.
3. If the operation result exceeds 2,147,483,647, the carry flag M1022 will be ON

16-bit instruction:



32-bit instruction:



API	Mnemonic			Operands	Function	Controllers			
21	D	SUB	P	S1 S2 D	Subtraction	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices X	Y	M	S	Word devices K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*	*	SUB, SUBP: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DSUB, DSUBP: 13 steps
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S_1: Minuend **S_2**: Subtrahend **D**: Remainder

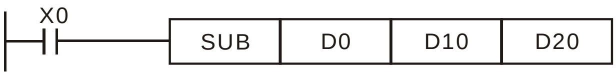
Explanations:

1. This instruction subtracts **S_1** and **S_2** in BIN format and stores the result in **D**
2. The MSB is the sign bit. 0 indicates positive and 1 indicates negative. All calculation is algebraically processed.
3. If **S_1**, **S_2** and **D** use device F, only 16-bit instruction is applicable.
4. Flags: M1020 (Zero flag), M1021 (Borrow flag), M1022 (Carry flag). The flag operations of ADD instruction can also be applied to the subtract instruction.

Program Example 1:

In 16-bit BIN subtraction:

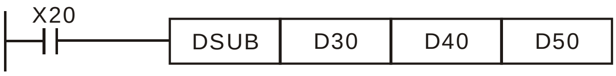
When X0 = ON, the content in D0 will minus the content in D10 and the results will be stored in D20



Program Example 2:

In 32-bit BIN subtraction:

When X10 = ON, the content in (D31, D30) will minus the content in (D41, D40) and the results will be stored in (D51, D50). D30, D40 and D50 are low word; D31, D41 and D51 are high word



(D31, D30) – (D41, D40) = (D51, D50)

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
22	D	MUL	P	S1 S2 D	Multiplication				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MUL, MULP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*		DMUL, DMULP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*		
D								*	*	*	*	*	*	*		

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S_1: Multiplicand **S_2**: Multiplicator **D**: Product

Explanations:

1. This instruction multiplies **S_1** by **S_2** in BIN format and stores the result in **D**. Care should be taken on positive/negative signs of **S_1**, **S_2** and **D** when doing 16-bit and 32-bit operations.
2. MSB = 0, positive; MSB = 1, negative.
3. If operands **S_1**, **S_2** use index F, then only 16-bit instruction is available.
4. If operand **D** use index E, then only 16-bit instruction is available.
5. 16-bit BIN multiplication

S1 S2 D+1 D

b15……………. b00 X b15……………. b00 = b31………… b16 b15………….b00

b15 is the sign bit / b15 is the sign bit / b31 is the sign bit(b15 of D+1)

b15=0,S1 is a positive value / b15=0,S2 is a positive value / b31=0,D(D+1) is a positive value

B15=1,S1 is a negative value / b15=1,S2 is a negative value / b31=1,D(D+1) is a negative value

16-bit value x 16-bit value = 32-bit value

If **D** is specified with a bit device, it can designate K1 ~ K4 to store a 16-bit result. Users can use consecutive 2 16-bit registers to store 32-bit data.

If the product of a 16-bit multiplication must be a 16-bit value (16-bit value x 16-bit value = 16-bit value), users have to use API 114 MUL16/MUL16P. Please refer to the explanation of API 114 MUL16/MUL16P for more information.

6. 32-bit BIN multiplication

S1+1 S1 S2+1 S2 D+3 D+2 D+1 D

b31..b16 b15..b00 X b31..b16 b15..b00 = b63. b48 b47. b32 b31. b16 b15. b00

b31 is the sign bit / b31 is the sign bit / b63 is the sign bit(b15 of D+3)

B31=0,S1(S1+1) is a positive value b31=0,S2(S2+1) is a positive value b63=0, D~(D+3) is a positive value

b31=1,S1(S1+1) is a negative value b31=1,S2(S2+1) is a negative value b63=1, D~(D+3) is a negative value

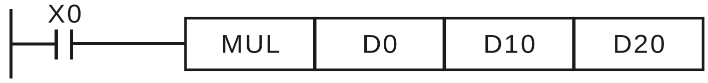
32-bit value x 32-bit value = 64-bit value

If **D** is specified with a word device, it can specify K1~K8 to store a 32-bit result. Users can use 2 consecutive 32-bit registers to store 64-bit data.

If the product of a 32-bit multiplication must be a 32-bit value (32-bit value x 32-bit value = 32-bit value), users have to use API 114 MUL32/MUL32P. Please refer to the explanation of API 114 MUL32/MUL32P for more information.

Program Example:

The 16-bit D0 is multiplied by the 16-bit D10 and brings forth a 32-bit product. The higher 16 bits are stored in D21 and the lower 16-bit are stored in D20. ON/OFF of MSB indicates the positive/negative status of the operation result.



(D0) × (D10) = (D21, D20)

16-bit × 16-bit = 32-bit

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
23	D	DIV	P	S1 S2 D	Division				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DIV, DIVP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*		DDIV, DDIVP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*		
D								*	*	*	*	*	*	*		

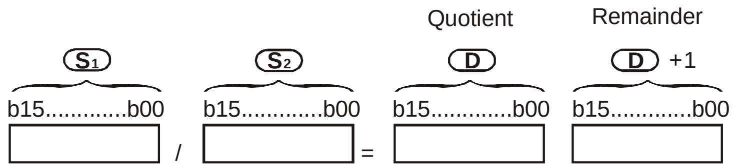
PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S_1: Dividend **S_2**: Divisor **D**: Quotient and remainder

Explanation:

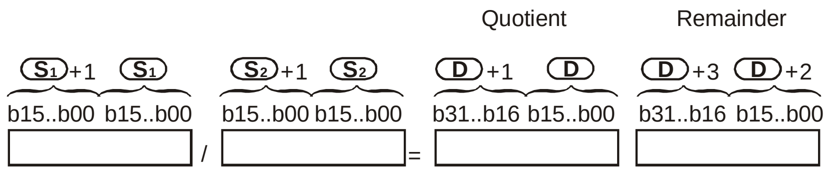
1. This instruction divides **S_1** and **S_2** in BIN format and stores the result in **D**. Care should be taken on positive/negative signs of **S_1**, **S_2** and **D** when doing 16-bit and 32-bit operations.
2. This instruction will not be executed when the divisor is 0. M1067 and M1068 will be ON and D1067 records the error code 0E19 (hex).
3. If operands **S_1**, **S_2** use index F, then only 16-bit instruction is available.
4. If operand **D** use index E, then only 16-bit instruction is available.
5. 16-bit BIN division:



If D is specified with a bit device, it can designate K1 ~ K4 to store a 16-bit result. Users can use consecutive 2 16-bit registers to store 32-bit data of the quotient and remainder.

If users want to store the quotient of a 16-bit division (leave out the remainder), they have to use AP I115 DIV16/DIV16P. Please refer to the explanation of API 115 DIV16/DIV16P for more information.

6. 32-bit BIN division:



If D is specified with a bit device, it can designate K1 ~ K8 to store a 32-bit result. Users can use consecutive 2 32-bit registers to store the quotient and remainder.

If users want to store the quotient of a 32-bit division (leave out the remainder), they have to use AP I115 DIV32/DIV32P. Please refer to the explanation of API 115 DIV32/DIV32P for more information.

Program Example:

When X0 = ON, D0 will be divided by D10 and the quotient will be stored in D20 and remainder in D21. ON/OFF of the MSB indicates the positive/negative status of the result value..

API	Mnemonic			Operands	Function	Controllers			
24	D	INC	P	D	Increment	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	INC, INCP: 3 steps
D								*	*	*	*	*	*	*	*	DINC, DINCP: 5 steps

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

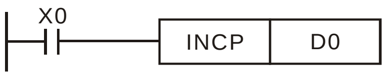
D: Destination device

Explanations:

1. If the instruction is not used in pulse execution mode, the content in the designated device **D** will plus “1” in every scan period
2. When INC is executed, the content in **D** will be incremented. However, in 16-bit instruction, if +32,767 is reached and “1” is added, it will write a value of –32,768 to the destination. In 32-bit instruction, if +2,147,483,647 is reached and “1” is added, it will write a value of -2,147,483,648 to the destination.
3. This instruction is generally used in pulse execution mode (INCP, DINCP).
4. If operand **D** uses index F, only a 16-bit instruction is applicable..
5. The operation results will not affect M1020 ~ M1022.

Program Example:

When X0 is triggered, the content of D0 will be incremented by 1.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
25	D	DEC	P	D	Decrement				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DEC, DECP: 3 steps
D								*	*	*	*	*	*	*	*	DDEC, DDECP: 5 steps

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

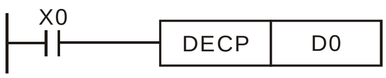
D: Destination device

Explanation:

1. If the instruction is not used in pulse execution mode, the content in the designated device D will minus “1” in every scan whenever the instruction is executed.
2. This instruction is generally used in pulse execution mode (DECP, DDECP).
3. In 16-bit instruction, if –32,768 is reached and “1” is minused, it will write a value of +32,767 to the destination. In 32-bit instruction, if -2,147,483,648 is reached and “1” is minused, it will write a value of +2,147,483,647 to the destination.
4. If operand **D** uses index F, only a 16-bit instruction is applicable.
5. The operation results will not affect M1020 ~ M1022

Program Example:

When X0 is triggered, the value in D0 will be decremented by 1.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
26	WAND	P	S₁ S₂ D	Logical Word AND				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	WAND, WANDP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

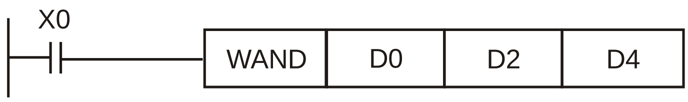
S_1: Source data device 1 **S_2**: Source data device 2 **D**: Operation result

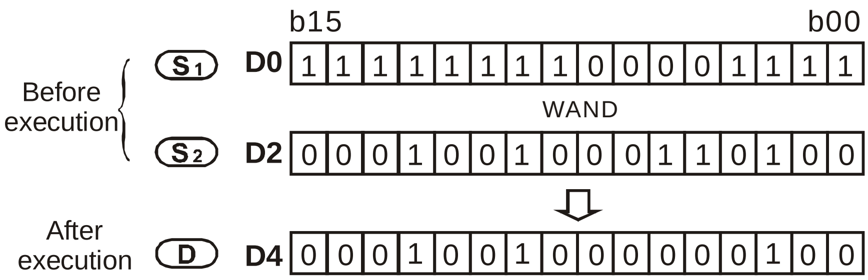
Explanations:

1. This instruction conducts logical AND operation of **S_1** and **S_2** in 16-bit mode and stores the result in **D**
2. For 32-bit operation please refer to DAND instruction..

Program Example:

When X0 = ON, the 16-bit source D0 and D2 are analyzed and the operation result of the logical AND operation is stored in D4.





API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
26		DAND	P	S1 S2 D	Logical DWord AND				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DAND, DANDP: 13 steps
S_1					*	*	*	*	*	*	*	*	*	*		
S_2					*	*	*	*	*	*	*	*	*	*		
D								*	*	*	*	*	*	*		

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

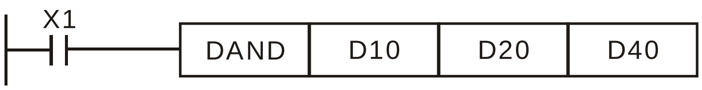
S_1: Source data device 1 **S_2**: Source data device 2 **D**: Operation result

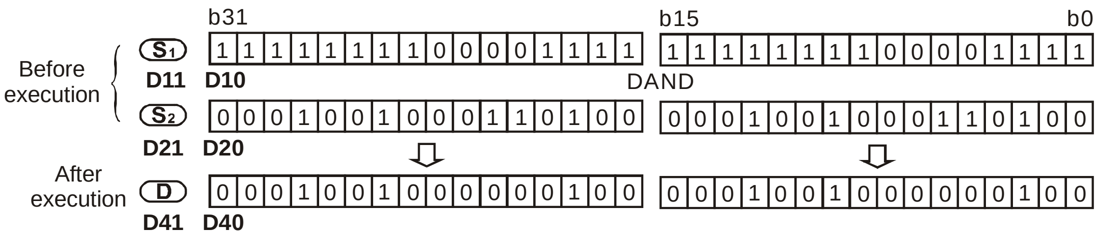
Explanations:

1. Logical double word (32-bit) AND operation.
2. This instruction conducts logical AND operation of **S_1** and **S_2** in 32-bit mode and stores the result in **D.**
3. If operands **S_1**, **S_2**, **D** use index F, only a 16-bit instruction is available.

Program Example:

When X1 = ON, the 32-bit source (D11, D10) and (D21, D20) are analyzed and the result of the logical AND is stored in (D41, D40).





API	Mnemonic		Operands	Function	Controllers			
27	WOR	P	S1 S2 D	Logical Word OR	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																WOR, WORP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

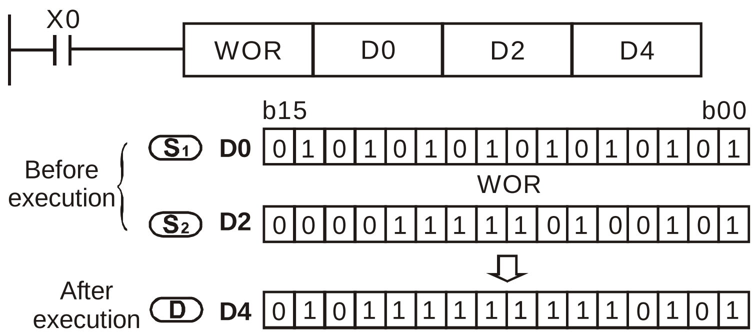
S_1: Source data device 1 **S_2**: Source data device 2 **D**: Operation result

Explanations:

1. This instruction conducts logical OR operation of **S_1** and **S_2** in 16-bit mode and stores the result in **D**.
2. For 32-bit operation please refer to DOR instruction.

Program Example:

When X0 = ON, the 16-bit data source D0 and D2 are analyzed and the result of the logical OR is stored in D4.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
27		DOR	P	S1 S2 D	Logical DWord OR				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DOR, DORP: 13 steps
S_1					*	*	*	*	*	*	*	*	*	*		
S_2					*	*	*	*	*	*	*	*	*	*		
D								*	*	*	*	*	*	*		

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

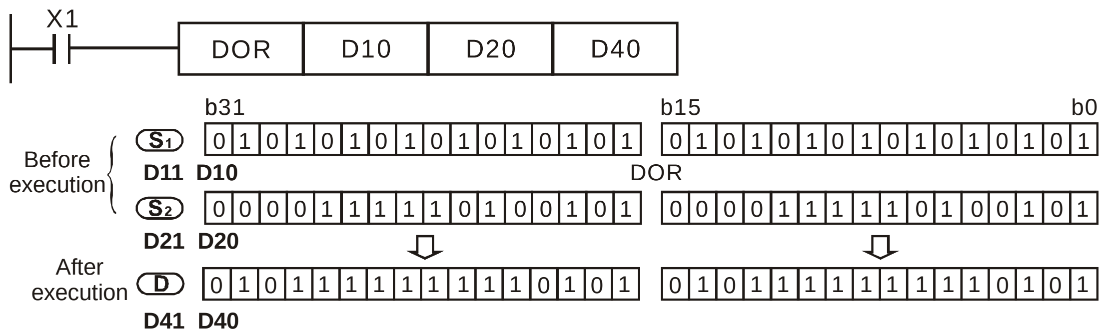
S_1: Source data device 1 **S_2**: Source data device 2 **D**: Operation result

Explanations:

1. Logical double word (32-bit) OR operation.
2. This instruction conducts logical OR operation of **S_1** and **S_2** in 32-bit mode and stores the result in **D**.
3. If operands **S_1**, **S_2**, **D** use index F, then only a 16-bit instruction is available.

Program Example:

When X1 is ON, the 32-bit data source (D11, D10) and (D21, D20) are analyzed and the operation result of the logical OR is stored in (D41, D40).



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
28		WXOR	P	S1 S2 D	Logical Word XOR				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	WXOR, WXORP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

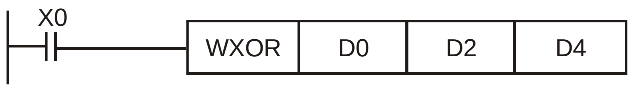
S_1: Source data device 1　　**S_2**: Source data device 2　　**D**: Operation result

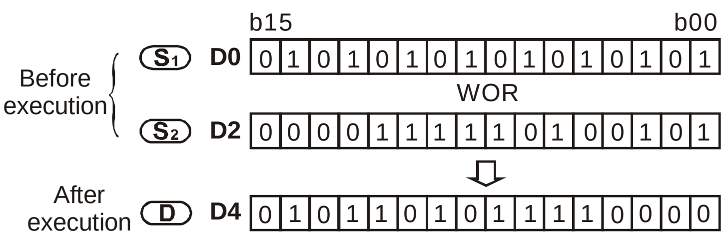
Explanations:

1. This instruction conducts logical XOR operation of **S_1** and **S_2** in 16-bit mode and stores the result in **D**
2. For 32-bit operation please refer to DXOR instruction.

Program Example:

When X0 = ON, the 16-bit data source D0 and D2 are analyzed and the operation result of the logical XOR is stored in D4.





API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
28	D XOR	P	S1 S2 D	Logical DWord XOR				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*		DXOR, DXORP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*		
D								*	*	*	*	*	*	*		

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

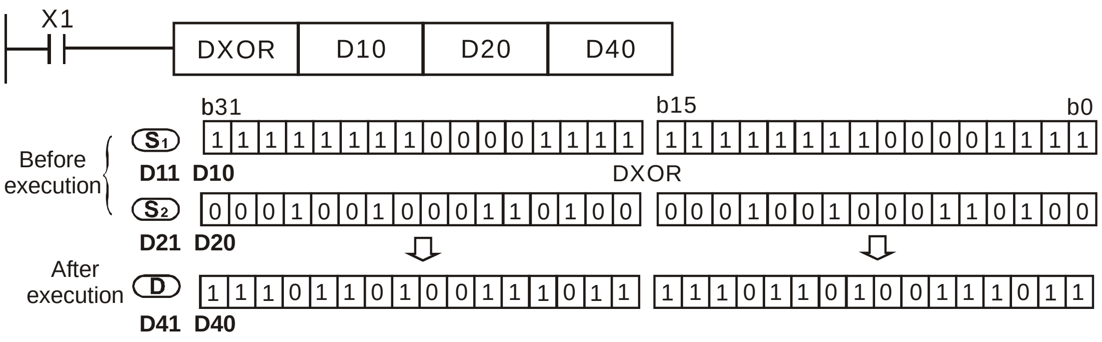
S_1: Source data device 1 **S_2**: Source data device 2 **D**: Operation result

Explanations:

1. Logical double word (32-bit) XOR operation.
2. This instruction conducts logical XOR operation of **S_1** and **S_2** in 32-bit mode and stores the result in **D**
3. If operands **S_1**, **S_2**, **D** use index F, only a 16-bit instruction is available.

Program Example:

When X1 = ON, the 32-bit data source (D11, D10) and (D21, D20) are analyzed and the operation result of the logical XOR is stored in (D41, D40).



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
29	D	NEG	P	D	2's Complement (Negation)				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	NEG, NEGP: 3 steps
D								*	*	*	*	*	*	*	*	DNEG, DNEGP: 5 steps

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

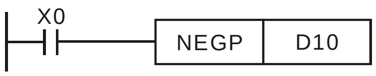
D: Device to store the operation result of 2's Compliment

Explanations:

1. This instruction conducts operation of 2's complement and can be used for converting a negative BIN value into an absolute value.
2. This instruction is generally used in pulse execution mode (NEGP, DNEGP).
3. If operand **D** uses index F, only a 16-bit instruction is available.

Program Example 1:

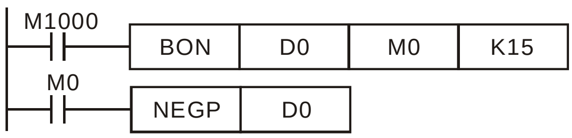
When X0 goes from OFF to ON, the phase of each bit in D10 will be reversed (0→1, 1→0) and then 1 will be added to the Least Significant Bit (LSB) of the register. Operation result will then be stored in D10.



Program Example 2:

To obtain the absolute value of a negative value:

1. When MSB (b15) of D0 is "1", M0 = ON. (D0 is a negative value).
2. When M0 = ON, the absolute value of D0 can be obtained by NEG instruction.



Program Example 3:

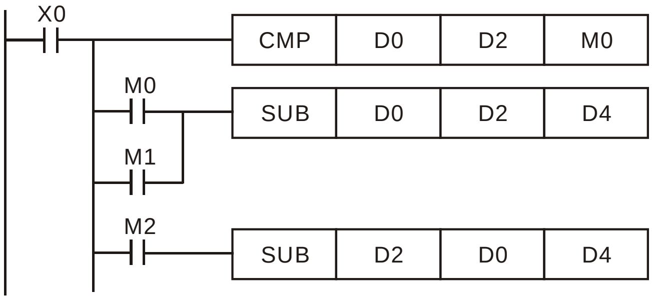
Obtain the absolute value of the remainder of the subtraction. When X0 = ON,

a) If D0 > D2, M0 = ON.

b) If D0 = D2, M1 = ON.

c) If D0 < D2, M2 = ON.

d) D4 is then able to remain positive.



Detailed explanations on negative value and its absolute value

1. MSB = 0 indicates the value is positive while MSB = 1 indicates the value is negative.
2. NEG instruction can be applied to convert a negative value into its absolute value.

(D0=2)
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0

(D0=1)
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

(D0=0)
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

(D0=-1) 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 → ($\overline{D0}$)+1=1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

(D0=-2) 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 → ($\overline{D0}$)+1=2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0

(D0=-3) 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 → ($\overline{D0}$)+1=3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1

(D0=-4) 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 → ($\overline{D0}$)+1=4 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0

(D0=-5) 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 → ($\overline{D0}$)+1=5 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1

⋮

(D0=-32,765) 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 → ($\overline{D0}$)+1=32,765 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1

(D0=-32,766) 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 → ($\overline{D0}$)+1=32,766 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0

(D0=-32,767) 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 → ($\overline{D0}$)+1=32,767 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

(D0=-32,768) 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 → ($\overline{D0}$)+1=-32,768 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Max. absolute value is 32,767

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
114	MUL16 MUL32	P	(S1) (S2) (D)	16-bit Multiplication 32-bit Multiplication				

Type OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S1					*	*	*	*	*	*	*	*	*	*		MUL16, MUL16P:7 steps
S2					*	*	*	*	*	*	*	*	*	*		MUL32, MUL32P:13
D								*	*	*	*	*	*	*		steps

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S1: Multiplicand **S2**: Multiplicator **D**: Product

Explanations:

1. MUL16 and MUL16P are 16-bit instructions. MUL32 and MUL32P are 32-bit instructions.
2. The signed binary value in **S1** is multiplied by the signed binary value in **S2**, and the product is stored in **D**. Notice that it is applicable to normal algebraic regulations.
3. If the sign bit is 0, it represents a positive value. If the sign bit is 1, it represents a negative value.
4. The models which are supported are DVP-ES2/EX2 v. 3.22, DVP-SS2 v. 3.20, DVP-SA2/SX2 v. 2.66, and DVP-SE v.1.60 (and above).
5. 16-bit binary multiplication

   (S1) b15................ b0 X (S2) b15................ b0 = (D) b15................ b0

   b15 is a sign bit. b15 is a sign bit. b15 is a sign bit.

   16-bit value×16-bit value=16-bit value

   If **D** is a bit device, users can use K1~K4, and form 16 bits. **D** only occupies 16 bits.
6. 32-bit binary multiplication

   (S1)+1 b31.......b16 (S1) b15.........b0 X (S2)+1 b31.......b16 (S2) b15.........b0 = (D)+1 b31.......b16 (D) b15.........b0

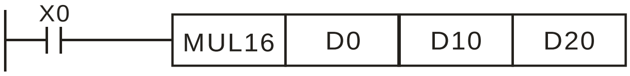
   b31 is a sign bit b31 is a sign bit b31 is a sign bit

   32-bit value×32-bit value=32-bit value

   If **D** is a word device, users can use K1~K8, and forms 32 bits. **D** only occupies 32 bits.

Program Example 1:

The 16-bit value K100 in D0 is multiplied by the 16-bit value K200 in D10, and the product is stored in D20. Whether the product is a positive value or a negative value depends on the leftmost bit (bit 15) in D20. If bit 15 in D20 is 0, the product stored in D20 is a positive value. If bit 15 in D20 is 1, the product stored in D20 is a negative value.



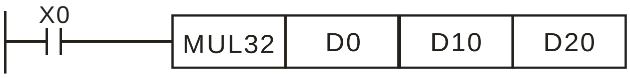
16-bit value×16-bit value=16-bit value

⇨ D0 × D10 = D20

⇨ D0=K100, D10=K200, D20=K20,000

Program Example 2:

The 32-bit value K10,000 in (D1, D0) is multiplied by the 32-bit value K20,000 in (D11, D10), and the product is stored in (D21, D20). Whether the product is a positive value or a negative value depends on the leftmost bit (bit 31) in (D21, D20). If bit 31 in (D21, D20) is 0, the product stored in (D21, D20) is a positive value. If bit 31 in (D21, D20) is 1, the product stored in (D21, D20) is a negative value.



32-bit value×32-bit value=32-bit value

⇨ (D1,D0) ×(D11,D10) = (D21,D20)

⇨ (D1,D0)=K10,000, (D11,D10)=K20,000, (D21, D20)=K200,000,000

Note:

1. If the product of a 16-bit multiplication is not a 16-bit signed value available, and is greater than the maximum 16-bit positive value (K32767), only the low 16 bits of the product will be stored, and the carry flag M1022 will be ON. If the product of a 16-bit multiplication is not a 16-bit signed value available, and is less than the minimum 16-bit negative value (K-32768), only the low 16 bits of the product will be stored, and the carry flag M1022 will be ON.
2. If users need a complete result of a 16-bit multiplication (a 32-bit value), they have to use API22 MUL/MULP. Please refer to the explanation of API22 MUL/MULP for more information.
3. If the product of a 32-bit multiplication is not a 32-bit signed value available, and is greater than the maximum 32-bit positive value (K2147483647), only the low 32 bits of the product will be stored, and the carry flag M1022 will be ON. If the product of a 32-bit multiplication is not a 32-bit signed value available, and is less than the minimum 32-bit negative value (K-2147483648), only the low 32 bits of the product will be stored, and the carry flag M1022 will be ON.
4. If users need a complete result of a 32-bit multiplication (a 64-bit value), they have to use API22 DMUL/DMULP. Please refer to the explanation of API22 DMUL/DMULP for more information.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
115	DIV16 DIV32	P	S1 S2 D	16-bit binary division 32-bit binary division				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*		DIV16, DIV16P: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*		DIV32, DIV32P: 13 steps
D								*	*	*	*	*	*	*		

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Dividend **S_2**: Divisor **D**: Quotient

Explanations:

1. DIV16 and DIV16P are 16-bit instructions. DIV32 and DIV32P are 32-bit instructions.
2. The singed binary value in **S_1** is divided by the signed binary value in **S_2**, and the quotient is stored in **D**. It is not a normal algebraic regulation. Notice the sign bits in **S_1**, **S_2**, and **D** in 16-bit binary division and 32-bit binary division.
3. If the divisor is 0, the instruction will not be executed, M1067 and M1068 will be ON, and the error code in D1067 will be H0E19.
4. The models which are supported are DVP-ES2/EX2 v. 3.22, DVP-SS2 v. 3.20, DVP-SA2/SX2 v. 2.66, and DVP-SE v. 1.60 (and above).
5. 16-bit binary division

S1		S2		D
b15................b0	/	b15................b0	=	b15................b0
b15 is a sign bit.		b15 is a sign bit.		b15 is a sign bit.

   If **D** is a bit device, users can use K1~K4, and form 16 bits. **D** only occupies 16 bits.

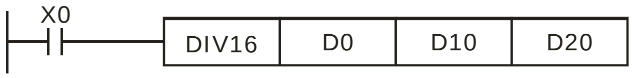
6. 32-bit bianry division

S1+1 S1		S2+1 S2		Quotient D+1 D	Remainder D+3 D+2
b15.....b0 b15.....b0	/	b15.....b0 b15.....b0	=	b15.....b0 b15.....b0	b15.....b0 b15.....b0

   If **D** is a word device, users can use K1~K8, and forms 32 bits. **D** only occupies 32 bits.

Program Example 1:

When X0 is ON, the dividend K103 in D0 is divided by the divisor K5 in D10, and the quotient is stored in D20. Whether the quotient is a positive value or a negative value depends on the leftmost bit in D20.



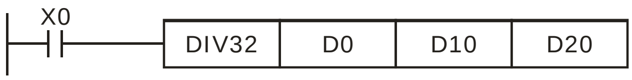
D0/D10=D20

⇨ K103/K5=K20, the remainder is K3.

⇨ D20=K20 (The remainder is left out.)

Program Example 2:

When X0 is ON, the dividend K81,000 in (D1, D0) is divided by the divisor K40,000 in (D11, D10), and the quotient is stored in (D21, D20). Whether the quotient is a positive value or a negative value depends on the leftmost bit in (D21, D20).



(D1,D0)/(D11,D10)=(D21,D20)

⇨ K81,000/K40,000=K2, The remainder is K1,000.

⇨ (D21,D20)=K2 (The remainder is left out.)

Note:

1. If users want to store the remainder of a 16-bit bianry division, they have to use API23 DIV/DIVP. Please refer to the explanation of API23 DIV/DIVP for more information.
2. If users want to store the remainder of a 32-bit bianry division, they have to use API23 DDIV/DDIVP. Please refer to the explanation of API23 DDIV/DDIVP for more information.

3.6.4 Rotation and Displacement

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
30	ROR	DROR	✓	Rotate right	✓	✓	✓	✓	5	9
31	ROL	DROL	✓	Rotate left	✓	✓	✓	✓	5	9
32	RCR	DRCR	✓	Rotate right with carry	✓	✓	✓	✓	5	9
33	RCL	DRCL	✓	Rotate left with carry	✓	✓	✓	✓	5	9
34	SFTR	-	✓	Bit shift right	✓	✓	✓	✓	9	-
35	SFTL	-	✓	Bit shift left	✓	✓	✓	✓	9	-
36	WSFR	-	✓	Word shift right	✓	✓	✓	✓	9	-
37	WSFL	-	✓	Word shift left	✓	✓	✓	✓	9	-
38	SFWR	-	✓	Shift register write	✓	✓	✓	✓	7	-
39	SFRD	-	✓	Shift register read	✓	✓	✓	✓	7	-

API	Mnemonic			Operands	Function	Controllers			
30	D	ROR	P	(D) (n)	Rotation Right	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ROR, RORP: 5 steps
D								*	*	*	*	*	*	*	*	DROR, DRORP: 9 steps
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

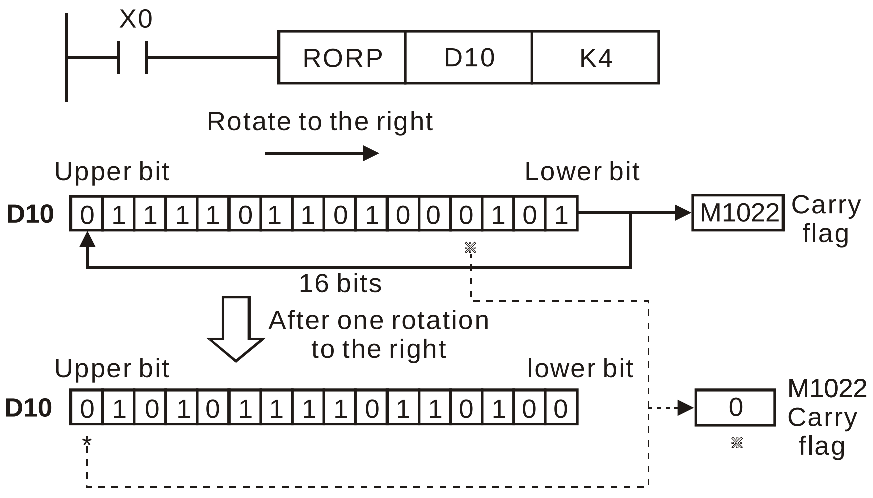
D: Device to be rotated **n**: Number of bits to be rotated in 1 rotation

Explanations:

1. This instruction rotates bit status of the device **D** to the right for **n** bits
2. The status of the last bit rotated (marked with ※) is copied to the carry flag M1022 (Carry flag)
3. This instruction is generally used in pulse execution mode (RORP, DRORP).
4. If operand **D** uses index F, only a 16-bit instruction is available.
5. If operand **D** is specified as KnY, KnM or KnS, only K4 (16-bit) or K8 (32-bit) is valid.
6. Valid range of operand **n**: 1≤ **n** ≤16 (16-bit), 1≤ **n** ≤32 (32-bit)

Program Example:

When X0 goes from OFF to ON, the 16 bits (4 bits as a group) in D10 will rotate to the right, as shown in the figure below. The bit marked with ※ will be sent to carry flag M1022..



API	Mnemonic			Operands	Function	Controllers			
31	D	ROL	P	D n	Rotate Left	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ROL, ROLP: 5 steps
D								*	*	*	*	*	*	*	*	DROL, DROLP: 9 steps
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

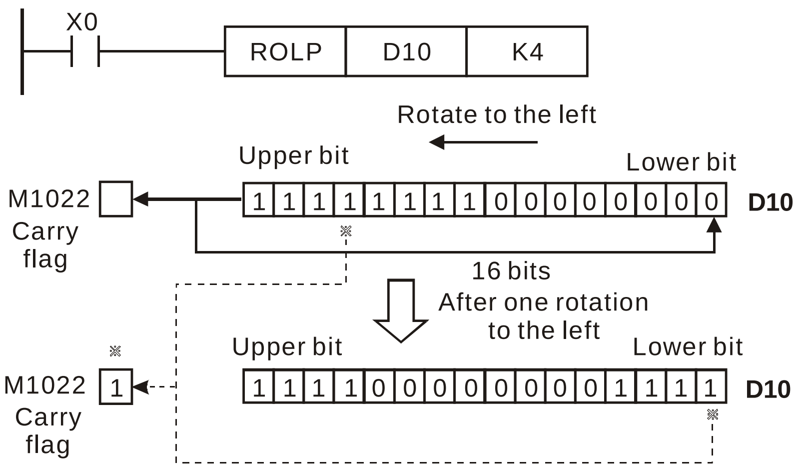
D: Device to be rotated **n**: Number of bits to be rotated in 1 rotation

Explanation:

1. This instruction rotates bit status of the device **D** to the left for **n** bits
2. The status of the last bit rotated (marked with ※) is copied to the carry flag M1022.
3. This instruction is generally used in pulse execution mode (ROLP, DROLP).
4. If operand **D** uses index F, only a 16-bit instruction is available.
5. If operand **D** is specified as KnY, KnM or KnS, only K4 (16-bit) or K8 (32-bit) is valid.
6. Valid range of operand **n**: 1≤ **n** ≤16 (16-bit), 1≤ **n** ≤32 (32-bit)

Program Example:

When X0 goes from OFF to ON, all the 16 bits (4 bits as a group) in D10 will rotate to the left, as shown in the figure below. The bit marked with ※ will be sent to carry flag M1022.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
32	D	RCR	P	(D) (n)	Rotation Right with Carry				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
D								*	*	*	*	*	*	*	*	RCR, RCRP: 5 steps
n					*	*										DRCR, DRCRP: 9 steps

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

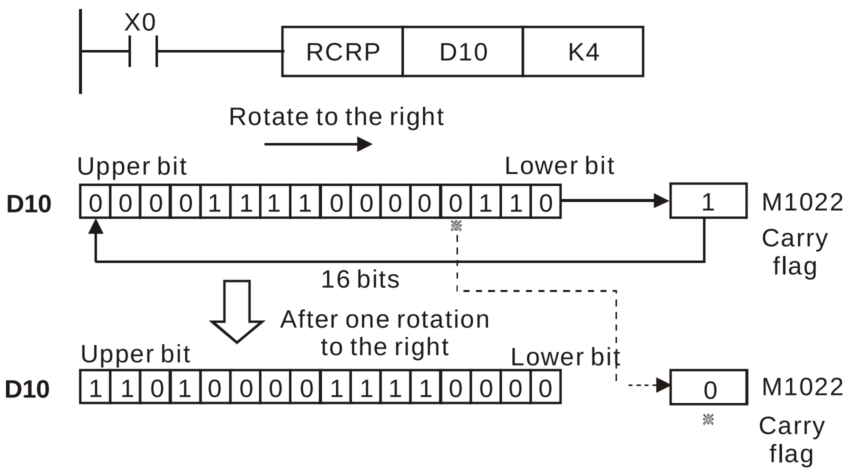
D: Device to be rotated **n**: Number of bits to be rotated in 1 rotation

Explanation:

1. This instruction rotates bit status of the device **D** together with M1022 to the right for **n** bits.
2. The status of the last bit rotated (marked with ※) is moved to the carry flag M1022.
3. This instruction is generally used in pulse execution mode (RCRP, DRCRP).
4. If operand **D** uses index F, only a 16-bit instruction is available.
5. If operand **D** is specified as KnY, KnM or KnS, only K4 (16-bit) or K8 (32-bit) is valid.
6. Valid range of operand **n**: 1≤ **n** ≤16 (16-bit), 1≤ **n** ≤32 (32-bit)

Program Example:

When X0 goes from OFF to ON, the 16 bits (4 bits as a group) in D10 together with carry flag M1022 (total 17 bits) will rotate to the right, as shown in the figure below. The bit marked with ※ will be moved to carry flag M1022



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
33	D	RCL	P	D n	Rotation Left with Carry				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	RCL, RCLP: 5 steps
D								*	*	*	*	*	*	*	*	DRCL, DRCLP: 9 steps
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

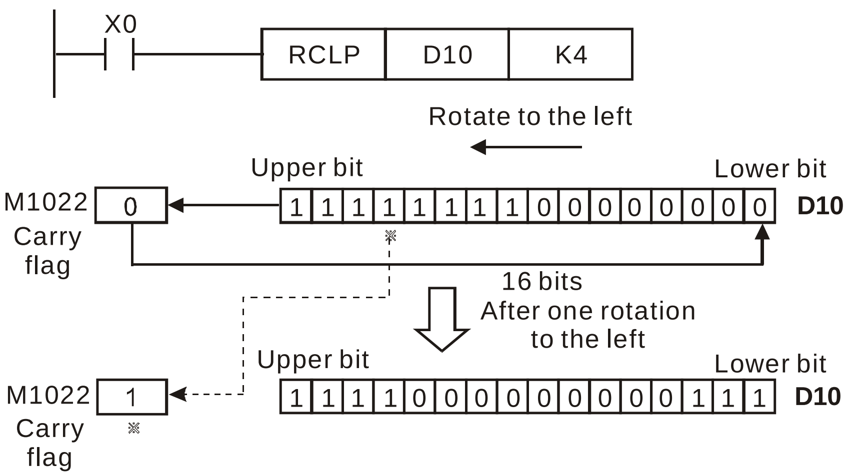
D: Device to be rotated **n**: Number of bits to be rotated in 1 rotation

Explanations:

1. This instruction rotates bit status of the device **D** together with M1022 to the left for **n** bits.
2. The status of the last bit rotated (marked with ※) is moved to the carry flag M1022.
3. This instruction is generally used in pulse execution mode (RCLP, DRCLP).
4. If operand **D** uses index F, only a 16-bit instruction is available.
5. If operand **D** is specified as KnY, KnM or KnS, only K4 (16-bit) or K8 (32-bit) is valid.
6. Valid range of operand **n**: 1≤ **n** ≤16 (16-bit), 1≤ **n** ≤32 (32-bit)

Program Example:

When X0 goes from OFF to ON, the 16 bits (4 bits as a group) in D10 together with carry flag M1022 (total 17 bits) will rotate to the left, as shown in the figure below. The bit marked with ※ will be sent to carry flag M1022.



API	Mnemonic			Operands	Function	Controllers			
34		SFTR	P	S D n1 n2	Bit Shift Right	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SFTR, SFTRP: 9 steps
S	*	*	*	*												
D		*	*	*												
n_1					*	*										
n_2					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: Start No. of source device **D**: Start No. of destination device **n_1**: Length of data to be shifted **n_2**: Number of bits to be shifted as a group

Explanation:

1. This instruction performs a right shift from source device of **n_2** bits starting from **S** to destination device of **n_1** bits starting from **D.**
2. This instruction is generally used in pulse execution mode (SFTRP).
3. Valid range of operand **n1**, **n2** : 1≤ **n2** ≤ **n1** ≤1024

Program Example:

1. When X0 is rising edge triggered, SFTR instruction shifts X0~X4 into 16 bit data M0~M15 and M0~M15 also shift to the right with a group of 4 bits.
2. The figure below illustrates the right shift of the bits in one scan.

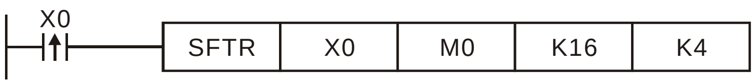
❶ M3~M0 → Carry

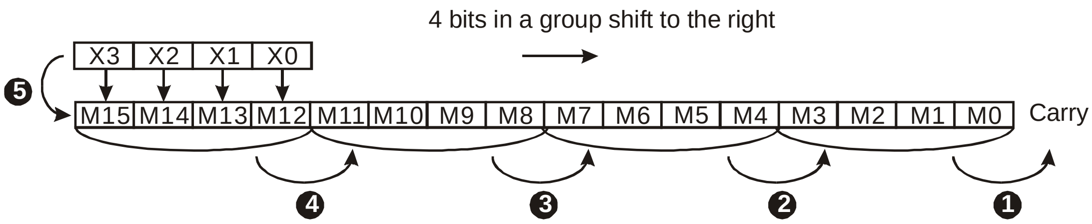
❷ M7~M4 → M3~M0

❸ M11~M8 → M7~M4

❹ M15~M12 → M11~M8

❺ X3~X0 → M15~M12 completed





API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
35	SFTL	P	(S) (D) (n1) (n2)	Bit Shift Left				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S	*	*	*	*												SFTL, SFTLP: 9 steps
D		*	*	*												
n_1					*	*										
n_2					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: Start No. of source device **D**: Start No. of destination device **n_1**: Length of data to be shifted **n_2**: Number of bits to be shifted as a group

Explanations:

1. This instruction performs a left shift from source device of **n_2** bits starting from **S** to destination device of **n_1** bits starting from **D**
2. This instruction is generally used in pulse execution mode (SFTLP).
3. Valid range of operand **n1**, **n2** : 1≤ **n2** ≤ **n1** ≤1024

Program Example:

1. When X0 is rising edge triggered, SFTL instruction shifts X0~X4 into 16-bit data M0~M15 and M0~M15 also shift to the left with a group of 4 bits.
2. The figure below illustrates the left shift of the bits in one scan

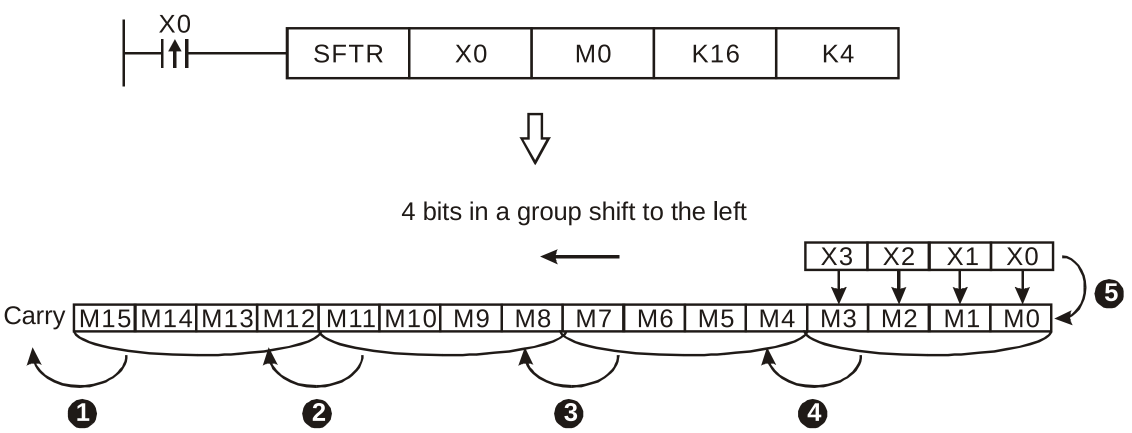
❶ M15~M12 → Carry

❷ M11~M8 → M15~M12

❸ M7~M4 → M11~M8

❹ M3~M0 → M7~M4

❺ X3~X0 → M3~M0 completed



API	Mnemonic		Operands	Function	Controllers			
36	WSFR	P	S D n1 n2	Word Shift Right	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S							*	*	*	*	*	*	*			WSFR, WSFRP: 9 steps
D								*	*	*	*	*	*			
n_1					*	*										
n_2					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: Start No. of source device **D**: Start No. of destination device $\mathbf{n_1}$: Length of data to be shifted $\mathbf{n_2}$: Number of devices to be shifted as a group

Explanations:

1. This instruction performs a right shift from source device of $\mathbf{n_2}$ registers starting from **S** to destination device of $\mathbf{n_1}$ registers starting from **D**.
2. This instruction is generally used in pulse execution mode (WSFRP).
3. The type of devices designated by **S** and **D** has to be the same, e.g. K_nX, K_nY, K_nM, and K_nS as a category and T, C, and D as another category
4. Provided the devices designated by **S** and **D** belong to K_n type, the number of digits of K_n in **S** and **D** has to be the same.
5. Valid range of operand **n1**, **n2** : 1≤ **n2** ≤ **n1** ≤512

Program Example 1:

1. When X0 is triggered, WSFRP instruction shifts D10~D13 into data stack D20~D35 and D20~D35 also shift to the right with a group of 4 registers.
2. The figure below illustrates the right shift of the registers in one scan.

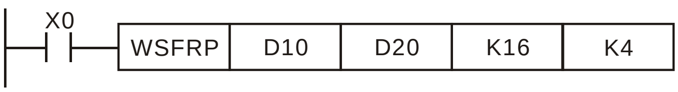
❶ D23~D20 → Carry

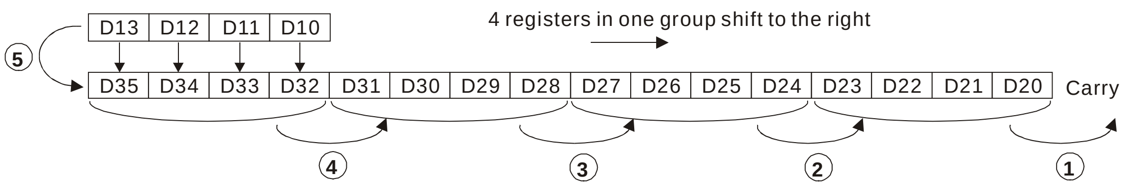
❷ D27~D24 → D23~D20

❸ D31~D28 → D27~D24

❹ D35~D32 → D31~D28

❺ D13 ~D10 → D35~D32 completed





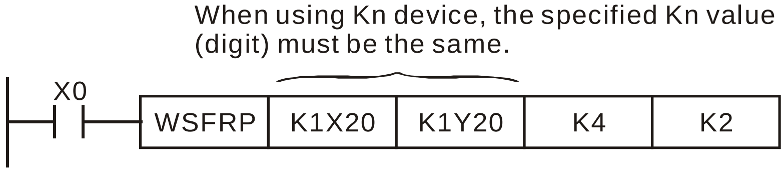
Program Example 2:

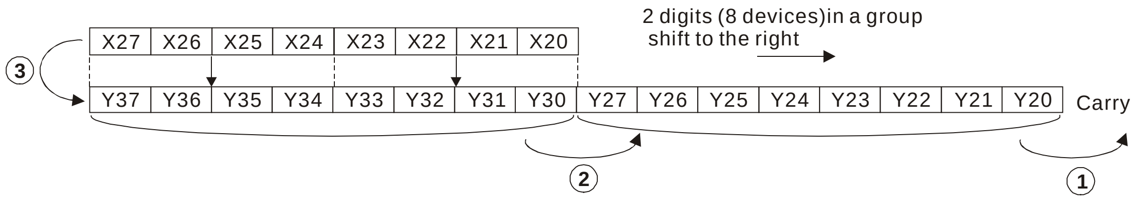
1. When X0 is triggered, WSFRP instruction shifts X20~X27 into data stack Y20~Y37 and Y20~Y37 also shift to the right with a group of 4 devices.
2. The figure below illustrates the right shift of the devices in one scan

❶ Y27~Y20 → carry

❷ Y37~Y30 → Y27~Y20

❸ X27~X20 → Y37~Y30 completed





API	Mnemonic		Operands	Function	Controllers			
37	WSFL	P	S D n1 n2	Word Shift Left	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S							*	*	*	*	*	*	*			WSFL, WSFLP: 9 steps
D								*	*	*	*	*	*			
n_1					*	*										
n_2					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: Start No. of source device **D**: Start No. of destination device $\mathbf{n_1}$: Length of data to be shifted $\mathbf{n_2}$: Number of devices to be shifted as a group

Explanations:

1. This instruction performs a left shift from source device of $\mathbf{n_2}$ registers starting from **S** to destination device of $\mathbf{n_1}$ registers starting from **D**.
2. This instruction is generally used in pulse execution mode (WSFLP).
3. The type of devices designated by **S** and **D** has to be the same, e.g. K_nX, K_nY, K_nM, and K_nS as a category and T, C, and D as another category
4. Provided the devices designated by **S** and **D** belong to K_n type, the number of digits of K_n in **S** and **D** has to be the same.
5. Valid range of operand **n1**, **n2** : 1≤ **n2** ≤ **n1** ≤512

Program Example:

1. When X0 is triggered, WSFLP instruction shifts D10~D13 into data stack D20~D35 and D20~D35 also shift to the left with a group of 4 registers.
2. The figure below illustrates the left shift of the words in one scan

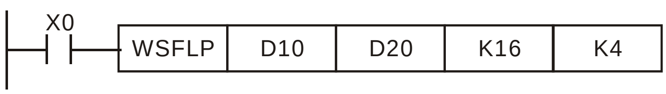
❶ D35~D32 → Carry

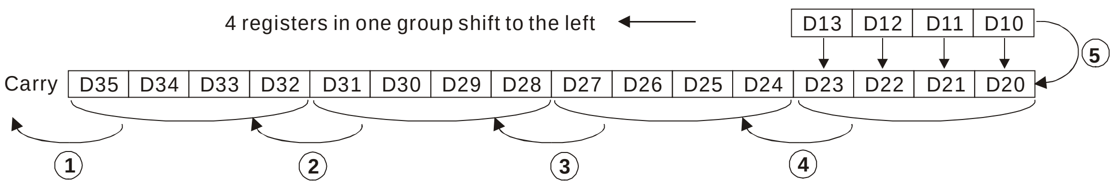
❷ D31~D28 → D35~D32

❸ D27~D24 → D31~D28

❹ D23~D20 → D27~D24

❺ D13~D10 → D23~D20 completed





API	Mnemonic		Operands	Function	Controllers			
38	SFWR	P	(S) (D) (n)	Shift Register Write	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SFWR, SFWRP: 7 steps
S					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*			
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: Source device **D**: Head address of data stack **n**: Length of data stack

Explanations:

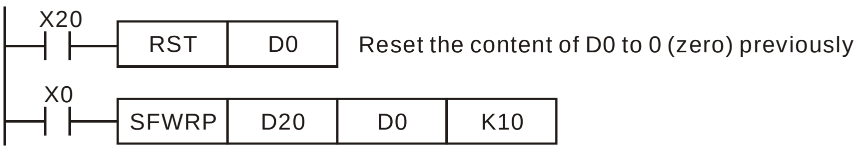
1. This instruction defines the data stack of **n** words starting from **D** as a "first-in, first out (FIFO)" data stack and specifies the first device as the pointer (**D**). When SFWRP is executed, content in pointer pluses 1, and the content in **S** will be written into the device designated by the pointer. When the content in pointer exceeds **n-1**, the instruction stops and carry flag M1022= ON.
2. This instruction is generally used in pulse execution mode (SFWRP).
3. Valid range of operand **n**: 2≤ **n** ≤512

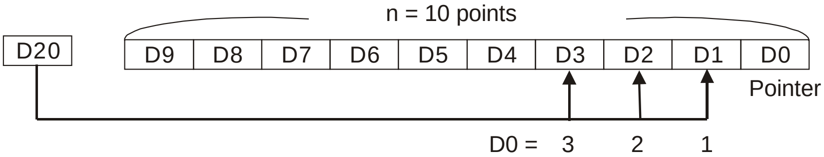
Program Example:

1. First, reset the content of D0. When X0 goes from OFF to ON, the content of D0 (pointer) becomes 1, and D20 is written into D1. If the content of D20 is changed and X0 is triggered again, pointer D0 becomes 2, and the content of D20 is then written into D2.
2. P The figure below illustrates the shift and writing process of the instruction.

   ❶ The content of D0 becomes 1.

   ❷. The content of D20 is written into D1.





Points to note:

This instruction can be used together with API 39 SFRD for the reading/writing of "first-in, first-out" stack data.

API	Mnemonic		Operands	Function	Controllers
39	SFRD	P	S D n	Shift Register Read	ES2/EX2/EC5 SS2 SA2 SE SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SFRD, SFRDP: 7 steps
S								*	*	*	*	*	*			
D								*	*	*	*	*	*	*	*	
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

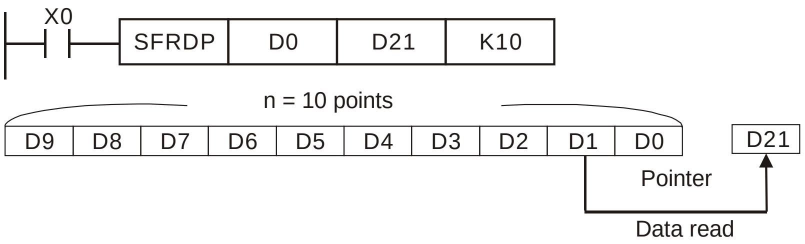
S: Head address of data stack **D**: Destination device **n**: Length of data stack

Explanation:

1. This instruction defines the data stack of **n** words starting from **S** as a FIFO data stack and specifies the first device as the pointer (**S**). The content of pointer indicates current length of the stack. When SFRDP is executed, first data (**S+1**) will be read out to **D**, all data in this stack moves up to fill the read device and content in pointer minuses 1. When the content in pointer = 0, the instruction stops and carry flag M1022= ON
2. This instruction is generally used in pulse execution mode (SFRDP).
3. Valid range of operand **n**: 2≤ **n** ≤512

Program Example:

1. When X0 goes from OFF to ON, D9~D2 are all shifted to the right and the pointer D0 is decremented by 1 when the content of D1 is read and moved to D21.
2. The figure below illustrates the shift and reading of the instruction.
   ❶ The content of D1 is read and moved to D21.
   ❷ D9~D2 are all shifted to the right.
   ❸ The content of D0 is decremented by 1.



3.6.5 Data Processing

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
40	ZRST	-	✓	Zone reset	✓	✓	✓	✓	5	-
41	DECO	-	✓	Decode	✓	✓	✓	✓	7	-
42	ENCO	-	✓	Encode	✓	✓	✓	✓	7	-
43	SUM	DSUM	✓	Sum of Active bits	✓	✓	✓	✓	5	9
44	BON	DBON	✓	Check specified bit status	✓	✓	✓	✓	7	13
45	MEAN	DMEAN	✓	Mean	✓	✓	✓	✓	7	13
46	ANS	-	-	Timed Annunciator Set	✓	✓	✓	✓	7	-
47	ANR	-	✓	Annunciator Reset	✓	✓	✓	✓	1	-
48	SQR	DSQR	✓	Square Root	✓	✓	✓	✓	5	9
49	FLT	DFLT	✓	Floating point	✓	✓	✓	✓	5	9

API	Mnemonic			Operands	Function	Controllers			
40		ZRST	P	D_1 D_2	Zone Reset	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ZRST, ZRSTP: 5 steps
D_1		*	*	*							*	*	*			
D_2		*	*	*							*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

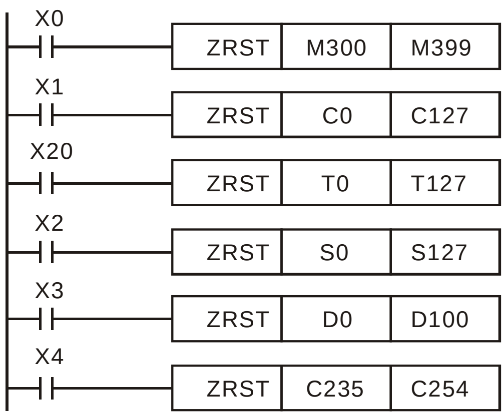
D_1: Starting device of the reset range **D_2**: End device of the reset range

Explanations:

1. When the instruction is executed, range **D_1** to **D_2** will be reset.
2. Operand **D_1** and **D_2** must be the same data type, Valid range: **D_1** ≦ **D_2**
3. When **D_1** > **D_2**, only operand designated by **D_2** will be reset.
4. This instruction is generally used in pulse execution mode (ZRSTP).

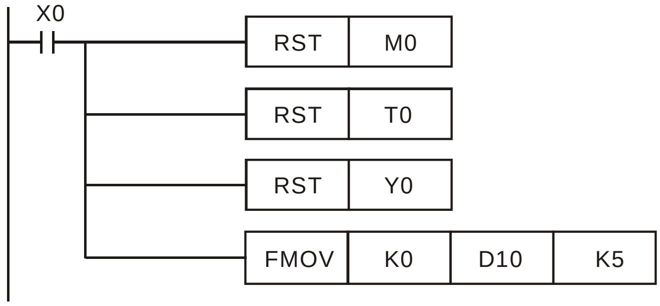
Program Example:

1. When X0 = ON, M300 to M399 will be reset.
2. When X1 = ON, C0 to C127 will all be reset, i.e. present value = 0 and associated contact/ output will be reset as well.
3. When X20 = ON, T0 to T127 will all be reset, i.e. present value = 0 and associated contact/ output will be reset as well.
4. When X2 = ON, the steps of S0 to S127 will be reset.
5. When X3 = ON, the data of D0 to D100 will be reset.
6. When X4 = ON, C235 to C254 will all be reset, i.e. present value = 0 and associated contact/ output will be reset as well.



Points to note:

1. Bit devices Y, M, S and word devices T, C, D can be individually reset by RST instruction.
2. For clearing multiple devices, API 16 FMOV instruction can be used to send K0 to word devices T, C, D or bit devices KnY, KnM, KnS.



API	Mnemonic		Operands	Function	Controllers			
41	DECO	P	S D n	Decode	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DECO, DECOP: 7 steps
S	*	*	*	*	*	*					*	*	*	*	*	
D		*	*	*							*	*	*	*	*	
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC52	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

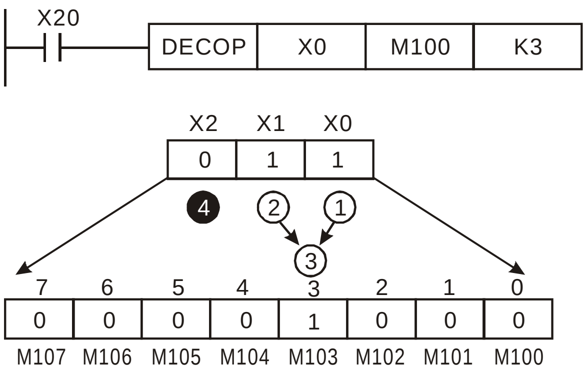
S: Source device to be decoded **D**: Device for storing the result **n**: Number of consecutive bits of **S**

Explanation:

1. The instruction decodes the lower "**n**" bits of **S** and stores the result of "2^n" bits in **D**.
2. This instruction is generally used in pulse execution mode (DECOP).
3. When operand **D** is a bit device, **n** = 1~8, when operand **D** is a word device, **n** = 1~4

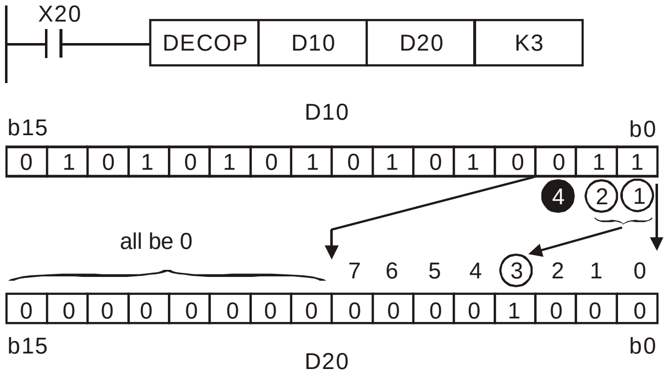
Program Example 1:

1. When **D** is used as a bit device, **n** = 1 ~ 8. Errors will occur if **n** = 0 or **n** > 8.
2. If **n** = 8, the decoded data is 2^8= 256 bits data.
3. When X20 goes from OFF to ON, the data of X0~X2 will be decoded to M100~M107.
4. If the source data is 3, M103 (third bit from M100) = ON.
5. After the execution is completed, X20 is turned OFF. The decoded results or outputs will retain their operation.



Program Example 2:

1. When **D** is used as a word device, n = 1 ~ 4. Errors will occur if n = 0 or n > 4.
2. When **n** = 4, the decoded data is 2^4 = 16 bits.
3. When X20 goes from OFF to ON, the data in D10 (b2 to b0) will be decoded and stored in D20 (b7 to b0). The unused bits in D20 (b15 to b8) will be set to 0.
4. The lower 3 bits of D10 are decoded and stored in the lower 8 bits of D20. The higher 8 bits of D20 are all 0.
5. After the execution is completed, X20 is turned OFF. The decoded results or outputs will retain their operation.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/EC5	SS2	SA2 SE	SX2
42	ENCO	P	(S) (D) (n)	Encode				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DECO, DECOP: 7 steps
S	*	*	*	*							*	*	*	*	*	
D											*	*	*	*	*	
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2/EC5	SS2	SA2 SE	SX2	ES2/EX2/EC5	SS2	SA2 SE	SX2	ES2/EX2/EC5	SS2	SA2 SE	SX2

Operands:

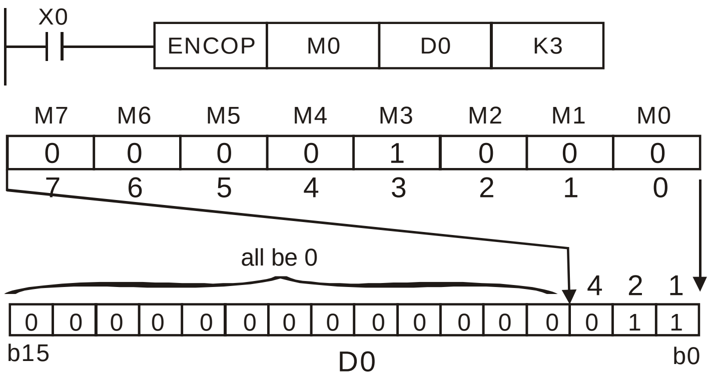
S: Source device to be encoded **D**: Device for storing the result **n**: Number of consecutive bits of **S**

Explanation:

1. The instruction encodes the lower "2^n" bits of source **S** and stores the result in **D**.
2. They highest active bit in **S** has the priority for encoding operation.
3. This instruction is generally used in pulse execution mode (ENCOP).
4. When operand **S** is a bit device, **n**=1~8, when operand **S** is a word device, **n**=1~4
5. If no bits in **S** is active (1), M1067, M1068 = ON and D1067 records the error code 0E1A (hex).

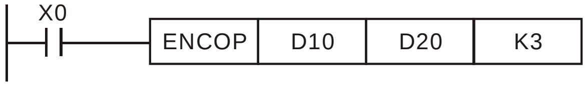
Program Example 1:

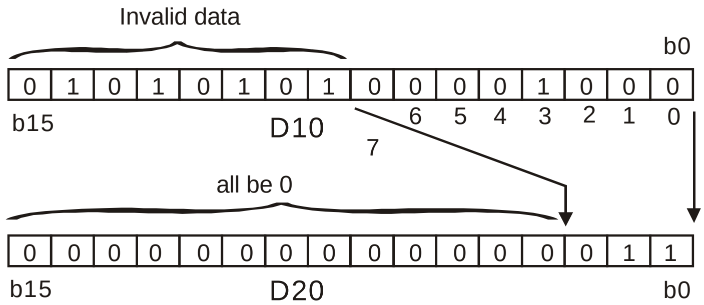
1. When **S** is used as a bit device, **n** = 1 ~ 8. Errors will occur if **n** = 0 or **n** > 8.
2. f **n** = 8, the decoded data is 2^8= 256 bits data.
3. When X0 goes from OFF to ON, the data in (M0 to M7) will be encoded and stored in lower 3 bits of D0 (b2 to b0). The unused bits in D0 (b15 to b3) will be set to 0.
4. After the execution is completed, X0 is turned OFF and the data in **D** remains unchanged.



Program Example 2:

1. When **S** is used as a word device, n = 1 ~ 4. Errors will occur if n = 0 or n > 4.
2. When **n** = 4, the decoded data is 2^4 = 16 bits data.
3. When X0 goes from OFF to ON, the 2^3 bits (b0 ~ b7) in D10 will be encoded and the result will be stored in the lower 3 bits of D20 (b2 to b0). The unused bits in D20 (b15 to b3) will be set to 0.
4. After the execution is completed, X0 is turned OFF and the data in **D** remains unchanged





API	Mnemonic			Operands	Function	Controllers			
43	D	SUM	P	S D	Sum of Active bits	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S					*	*	*	*	*	*	*	*	*	*	*	SUM, DSUMP: 5 steps
D											*	*	*	*	*	DSUM, DSUMP: 9 steps

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

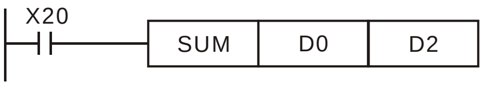
S: Source device **D**: Destination device for storing counted value

Explanation:

1. This instruction counts the total active bits in **S** and store the value in **D.**
2. **D** will occupy two registers when using in 32-bit instruction.
3. If operand **S**, **D** use index F, only a 16-bit instruction is available.
4. If there is no active bits, zero flag M1020 =ON.

Program Example:

When X20 = ON, all active bits in D0 will be counted and the result will be stored in D2.



0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0

D0

3

D2

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
44	D	BON	P	(S) (D) (n)	Check specified bit status				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S					*	*	*	*	*	*	*	*	*	*	*	BON, BONP: 7 steps
D		*	*	*												DBON, DBONP: 13 steps
n					*	*					*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

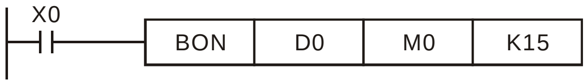
S: Source device **D**: Device for storing check result **n**: Bit number to be checked

Explanation:

1. The instruction checks the status of designated bit (specified by **n**) in **S** and stores the result in **D**
2. If operand **S** uses index F, only 16-bit instruction is available.
3. Valid range of operand **n** : **n** = 0~15 (16-bit), **n** = 0~31 (32-bit)

Program Example:

1. When X0 = ON, and bit15 of D0 = "1", M0 will be ON. If the bit15 is "0", M0 is OFF.
2. When X0 is OFF, M0 will retain its previous status.



b15 ... b0

0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	M0=Off

D0

b15 ... b0

1	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	M0=On

D0

API	Mnemonic			Operands	Function	Controllers			
45	D	MEAN	P	(S) (D) (n)	Mean	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MEAN, MEANP: 7 steps
S							*	*	*	*	*	*	*			DMEAN, DMEANP: 13
D								*	*	*	*	*	*	*	*	steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

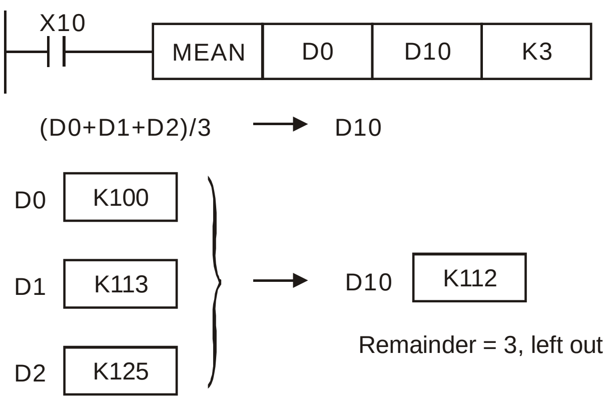
S: Source device **D**: Destination for storing result **n**: Number of consecutive device from **S**

Explanations:

1. The instruction obtains the mean value from **n** consecutive registers from **S** and stores the value in **D**.
2. Remainders in the operation will be ignored.
3. If **S** is not within the valid range, only those addresses within the valid range will be processed.
4. If **n** is out of the valid range (1~64), PLC will determine it as an “instruction operation error”.
5. If operand **D** uses index F, only a 16-bit instruction is available.
6. Valid range of operand **n** : **n** = 1~64

Program Example:

When X10 = ON, the contents in 3 (n = 3) registers starting from D0 will be summed and then divided by 3 to obtain the mean value. The result will be stored in D10 and the remainder will be left out



API	Mnemonic	Operands	Function	Controllers			
46	ANS	S m D	Timed Annunciator Set	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ANS: 7 steps
S											*					
m					*											
D				*												

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: Alarm timer **m**: Time setting **D**: Alarm

Explanations:

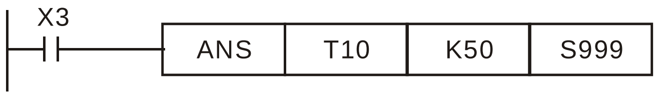
1. ANS instruction is used to drive the output alarm device in designated time.
2. Operand **S** valid range: T0~T183

   Operand **m** valid range: K1~K32,767 (unit: 100 ms)

   Operand **D** valid range: S912~S1023
3. Flag: M1048 (ON: Alarm is active), M1049 (ON: Alarm monitoring is enabled)
4. See ANR instruction for more information

Program Example:

If X3 = ON for more than 5 sec, alarm step relay S999 will be ON. S999 will remains ON after X3 is reset. (T10 will be reset, present value = 0)



API	Mnemonic			Function	Controllers			
47		ANR	P	Annunciator Reset	ES2/EX2/ EC5	SS2	SA2 SE	SX2

OP	Descriptions	Program Steps
N/A	Instruction driven by contact is necessary.	ANR, ANRP: 1 steps

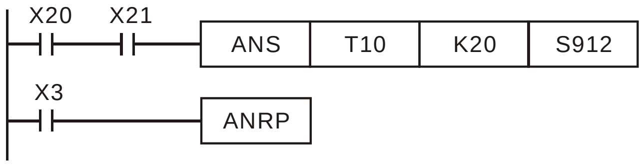
PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Explanations:

1. ANR instruction is used to reset an alarm.
2. When several alarm devices are ON, the alarm with smaller number will be reset.
3. This instruction is generally used in pulse execution mode (ANRP).

Program Example:

1. If X20 and X21 are ON at the same time for more than 2 sec, the alarm S912 will be ON. If X20 or X21 is reset, alarm S912 will remain ON but T10 will be reset and present value is cleared.
2. If X20 and X21 are ON less than 2 sec, the present value of T10 will be cleared.
3. When X3 goes from OFF → ON, activated alarms S912 will be reset.
4. When X3 goes from OFF → ON again, the alarm device with second lower number will be reset.



Points to note:

Flags:

1. M1048 (indicating alarm status): When M1049 = ON, enabling any of the alarm S912~S1023 turns M1048 ON.
2. M1049 (Enabling alarm monitoring): When M1049 = ON, D1049 will automatically hold the lowest alarm number in active alarms.

Application example of alarm device (production line):

X0 = Forward switch　　X1 = Backward switch

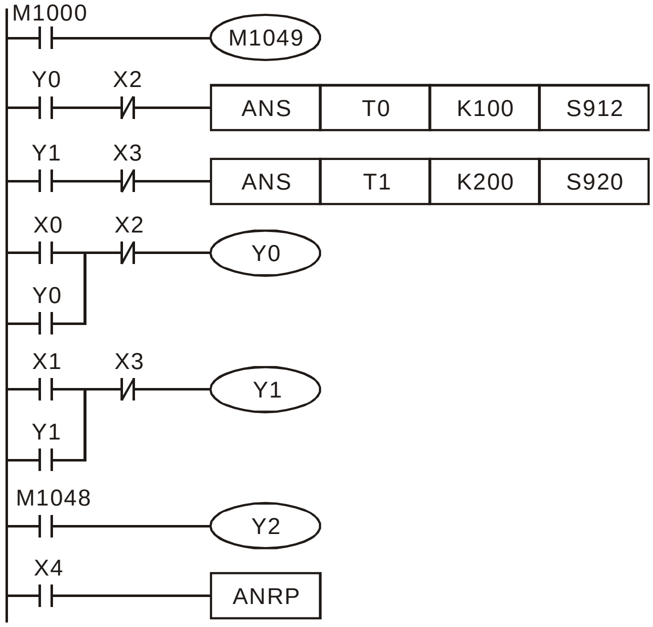
X2 = Front position switch　　X3 = Back position switch

X4 = Alarm reset button

Y0 = Forward　　Y1 = Backward

Y2 = Alarm indicator

S912 = Forward alarm　　S920 = Backward alarm



3. M1048 and D1049 are valid only when M1049 = ON.
4. When Y0 = ON for more than 10 sec and the product fails to reach the front position X2, S912 = ON
5. When Y1 = ON for more than 10 sec and the product fails to reach the back position X3, S920= ON.
6. When backward switch X1 = ON and backward device Y1 = ON, Y1 will go OFF only when the product reaches the back position switch X3.
7. Y2 is ON when any alarm is enabled.
8. Whenever X4 is ON, 1 active alarm will be reset. If there are several active alarms, the reset will start from the alarm with the lowest number and then the alarm with second lower number, etc.

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
48	D	SQR	P	(S) (D)	Square Root				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SQR, SQRP: 5 steps
S					*	*							*			DSQR, DSQRP: 9 steps
D													*			

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

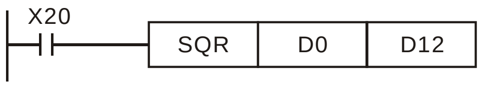
S: Source device **D**: Device for storing the result

Explanation:

1. This instruction performs a square root operation on **S** and stores the result in **D**.
2. **S** can only be a positive value. Performing a square root operation on a negative value will result in an error and the instruction will not be executed. The error flag M1067 and M1068 = ON and D1067 records error code H0E1B.
3. The operation result **D** should be integer only, and the decimal will be left out. When decimal is left out, borrow flag M1021 = ON.
4. When the operation result **D** = 0, zero flag M1020 = ON.

Program Example:

When X20 = ON, square root of D0 will be stored in D12.



$\sqrt{D0} \rightarrow D12$

API	Mnemonic			Operands	Function	Controllers			
49	D	FLT	P	S D	Floating Point	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		FLT, FLTP: 5 steps
S													*				DFLT, DFLTP: 9 steps
D													*				

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

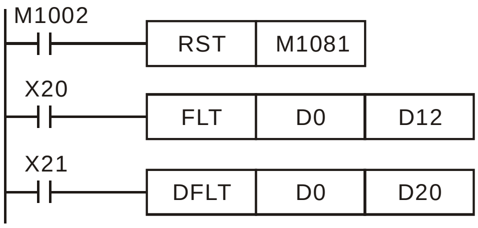
S: Source device   **D**: Device for storing the conversion result

Explanations:

1. When M1081 = OFF, the source **S** is converted from BIN integer to binary floating point value. At this time, 16-bit instruction FLT occupies 1 register for **S** and 2 registers for **D**.
   a) If the absolute value of the conversion result ＞ max. floating value, carry flag M1022 = ON.
   b) If the absolute value of the conversion result ＜ min. floating value, carry flag M1021 = ON.
   c) If conversion result is 0, zero flag M1020 = ON.
2. When M1081 is ON, the source **S** is converted from binary floating point value to BIN integer. (Decimal ignored). At this time, 16-bit instruction FLT occupies 2 registers for **S** and 1 register for **D**. The operation is same as instruction INT.
   a) If the conversion result exceeds the available range of BIN integer in **D** (for 16-bit: -32,768 ~ 32,767; for 32-bit: -2,147,483,648 ~ 2,147,483,647), **D** will obtain the maximum or minimum value and carry flag M1022 = ON.
   b) If the decimal is ignored, borrow flag M1021=ON.
   c) If the conversion result = 0, zero flag M1020=ON.
   d) After the conversion, **D** stores the result in 16 bits.

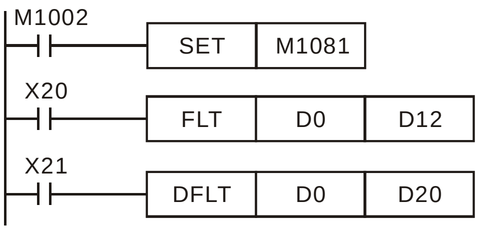
Program Example 1:

1. When M1081 = OFF, the BIN integer is converted into binary floating point value.
2. When X20 = ON, D0 is converted to D13, D12 (floating point).
3. When X21 = ON, D1, D0 are converted to D21, D20 (floating point).
4. Assume D0 is K10. When X10 is ON, the converted 32-bit value will be H41200000 and stored in 32-bit register D12 (D13)
5. If 32-bit register D0 (D1)=K100,000, X21 = ON. 32-bit of floating point after conversion will be H47C35000 and it will be saved in 32-bit register D20 (D21)



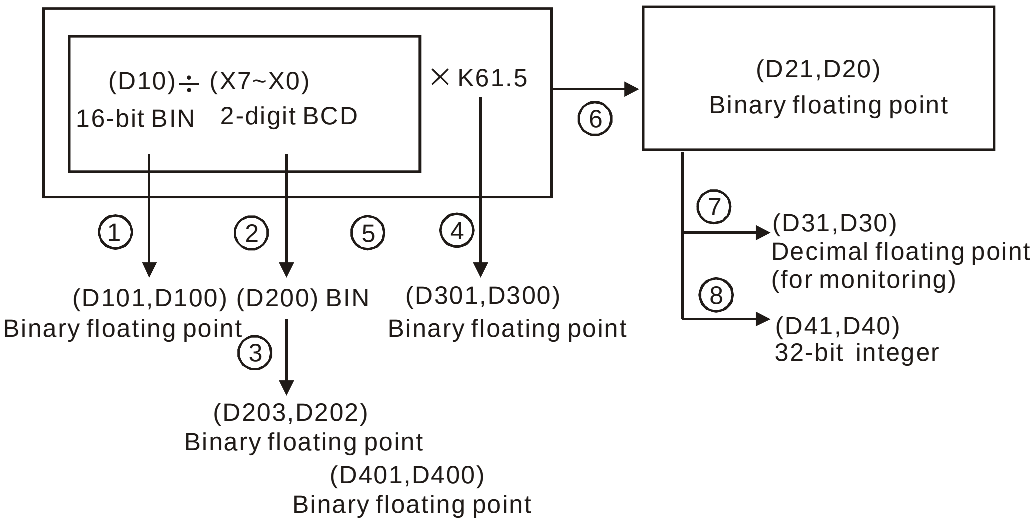
Program Example 2:

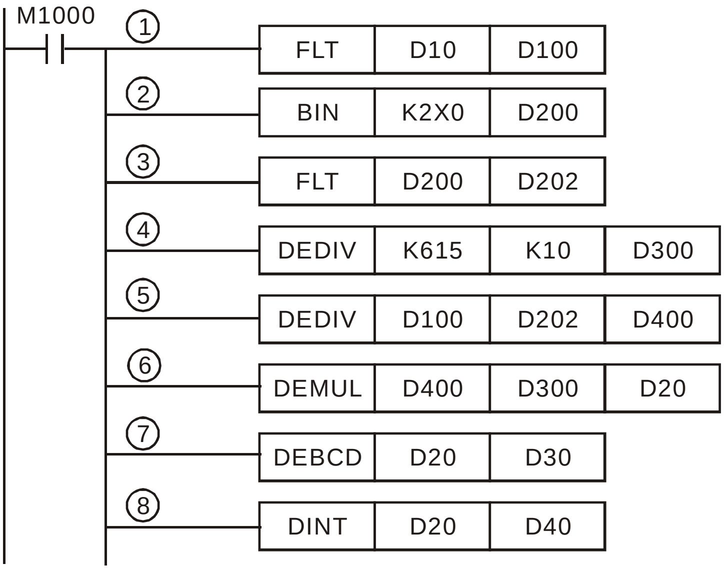
1. When M1081 = ON, the source data is converted from floating point value to BIN integer. (Decimal ignored)
2. When X20 = ON, D1 and D0 (floating point) are converted to D12 (BIN integer). If D0 (D1) = H47C35000, the result will be 100,000 which exceeds the available range of BIN integer in 16-bit register D12. In this case the result will be D12 = K32767, and M1022 = ON
3. When X21 = ON, D1 and D0 (floating point) are converted to D21, D20 (BIN integer). If D0 (D1) = H47C35000, the result is 100,000 and will be saved in 32-bit register D20 (D21).



Program Example 3:

Apply FLT instruction to complete the following operation





1. Covert D10 (BIN integer) to D101, D100 (floating point).
2. Covert the value of X7~X0 (BCD value) to D200 (BIN value).
3. Covert D200 (BIN integer) to D203, D202 (floating point).
4. Save the result of K615 ÷ K10 to D301, D300 (floating point).
5. Divide the floating point:

   Save the result of (D101, D100) ÷ (D203, D202) to D401, D400 (floating point).
6. Multiply floating point:

   Save the result of (D401, D400) × (D301, D300) to D21, D20 (floating point).
7. Covert floating point (D21, D20) to decimal floating point (D31, D30).
8. Covert floating point (D21, D20) to BIN integer (D41, D40).

3.6.6 High Speed Processing

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
50	REF	-	✓	Refresh	✓	✓	✓	✓	5	-
51	REFF	-	✓	Refresh and filter adjust	✓	✓	✓	✓	3	-
52	MTR	-	-	Input Matrix	✓	✓	✓	✓	9	-
53	-	DHSCS	-	High speed counter SET	✓	✓	✓	✓	-	13
54	-	DHSCR	-	High speed counter RESET	✓	✓	✓	✓	-	13
55	-	DHSZ	-	High speed zone compare	ES2 EX2	✓	✓	✓	-	17
56	SPD	-	-	Speed detection	✓	✓	✓	✓	7	-
57	PLSY	DPLSY	-	Pulse output	✓	✓	✓	✓	7	13
58	PWM	-	-	Pulse width modulation	✓	✓	✓	✓	7	-
59	PLSR	DPLSR	-	Pulse ramp	✓	✓	✓	✓	9	17

3.6.6.1 Output Points for High Speed Instructions

API	Mnemonic		PULSE	Applicable to	ES2/EX2/SS2 SA2/SX2/SE	ES2/EX2/SS2 SA2/SX2/SE	EC5
	16 bits	32 bits		Output Number	Y0, Y2	Y1, Y3	Y0, Y2, Y4, Y6
50	REF	-	✓		✓	✓	✓
57	PLSY	DPLSY	-		✓	✓	✓
58	PWM	-	-		✓	✓	✓
59	PLSR	DPLSR	-		✓	✓	✓

API	Mnemonic		Operands	Function	Controllers
50	REF	P	D n	Refresh	ES2/EX2/EC5, SS2, SA2 SE, SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
D	*	*														REF, REFP: 5 steps
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

D: Start device for I/O refresh **n**: Number of devices for I/O refresh

Explanations:

1. PLC updates I/O status between END instruction and the start of next program scan. If an immediate I/O refresh is needed, REF can be applied for performing I/O refresh immediately.
2. **D** can only be a multiple of 10, i.e. X0 or Y0, and the instruction is NOT applicable for I/O points on DIO modules.
3. Only the I/O points on MPU can be specified for operand D for I/O refresh.
   - When **D** specifies X0 and **n** ≦ 8, only X0~X7 will be refreshed. If **n** > 8, all I/O points on MPU will be refreshed.
   - When **D** specifies Y0 and **n** = 8, only Y0~X7 will be refreshed. If **n** > 8, all I/O points on MPU will be refreshed.
   - When **D** specifies X10 or Y10, I/O points on MPU except for X0~X7 or Y0~Y3 will all be refreshed regardless of **n** value, i.e. only status of X0~X7 or Y0~Y3 remains.
4. For EX2/SX2 MPU only: If M1180 = ON and REF instruction executes, PLC will read the A/D value and update the read value to D1110~D1113. If M1181 = ON and REF instruction executes, PLC will output the D/A value in D1116 and D1117 immediately. When A/D or D/A values are refreshed, PLC will reset M1180 or M1181 automatically.
5. Range for **n (ES2/EX2):** 4 ~ total I/O points on MPU. **n** should always be a multiple of 4.
6. Range for **n (SS2/SA2/SE/SX2):** 8 ~ total I/O points on MPU.
7. The function to update pulse number immediately is only available for the following modules and firmware, ES2, EX2, ES2-C: V3.60, ES2-E: V1.40, 28SA2, 12SA2, SX2: V3.0, 26SE: V1.92 and later.

Output Device	Y0	Y1	Y2	Y3
Refresh current position of output	M1672	M1673	M1674	M1675
Pulse output number	D1030/D1031	D1032/D1033	D1336/D1337	D1338/D1339

A. Normally, PLC only refreshes pulse output when the pulse instruction is executed. You can use output pulse to check the pulse number but if the program is big, it may cause a bigger different result in such a long scan.

B. When executing REF instruction with M1672-M1675, it can refresh the pulse output immediately. And when REF instruction works with M1672-M1675 flags, it is only used to refresh the pulse number not to refresh the actual inputs and outputs.

C. Refer to program example 5 for reference.

8. The function to update pulse number immediately is available for EC5: V1.00 and later.

Output Device	Y0	Y2	Y4	Y6
Refresh current position of output	M1672	M1674	M1676	M1677
Pulse output number	D1030/D1031	D1336/D1337	D1375/D1376	D1377/D1378

Program Example 1:

When X0 = ON, PLC will refresh the status of input points X0 ~ X7 immediately without delay.

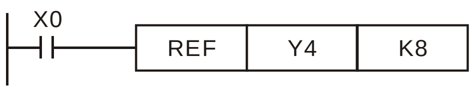
Program Example 2:

When X0 = ON, the 4 output signals on Y0 ~ Y3 will be sent to output terminals immediately before the program proceeds to END instruction.

Program Example 3:

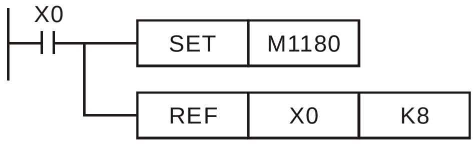
When X0 = ON, I/O points starting from X10 or Y4 will all be refreshed.

或



Program Example 4:

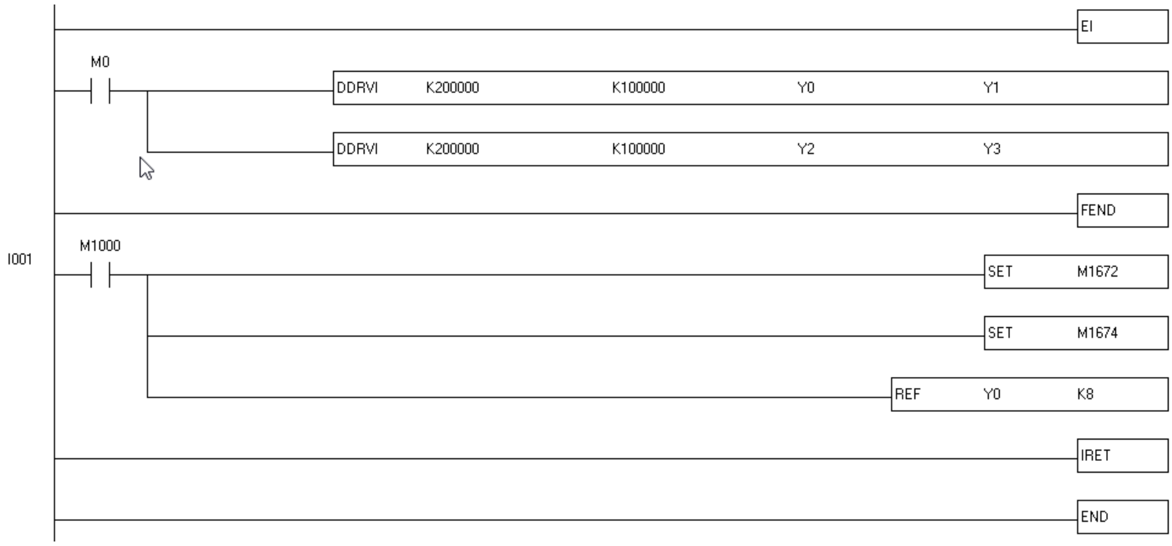
For DVP-EX2/SX2 only: When X0 = ON and M1180 = ON, A/D signal in D1110~D1113 will be refreshed immediately regardless of the settings of operands **D** and **n**



Program Example 5:

When M0 is ON, executing DDRVI instruction to output pulses. When an external interrupt occurs in X0, the program refreshes the pulse number immediately in D1030, D1031 and D1336, D1337. No need to wait for the scan.

3



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
51		REFF	P	n	Refresh and Filter Adjust				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	REFF, REFFP: 3 steps
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

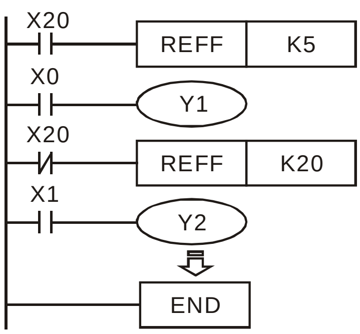
n: Response time (unit: ms)

Explanation:

1. PLC provides digital input filters to avoid interference. The response time (**n)** of X0 ~ X7 input filters can be adjusted by REFF instruction. The instruction sets the value specified in **n** to D1020 (X0 ~ X7 input filter time) directly. The instruction sets the value specified in **n** to D1021 (X10 ~ X17response time) for models including 28SS2 V3.42 /28SA2 V3.0 /26SE V1.0 and later versions.
2. When PLC turns from OFF to ON or the END instruction is reached, the response time is dictated by the value of D1020.
3. During program execution, the value in D1020 can be changed by using MOV instruction.
4. When using REFF instruction during program execution, the modified response time will be move to D1020 and refreshed until next program scan..
5. Range of **n**: = K2 ~ K20.

Program Example:

1. When the power of PLC turns from OFF to ON, the response time of X0~X7 inputs is specified by the value in D1020.
2. When X20 = ON, REFF K5 instruction is executed, response time changes to 5 ms and takes affect the next scan.
3. When X20 = OFF, the REFF instruction will not be executed, the response time changes to 20ms and takes affect the next scan.



Points to note:

Response time is ignored (no delay) when input points are occupied by external interrupts, high-speed counters or SPD instruction.

API	Mnemonic	Operands	Function	Controllers			
52	MTR	S D1 D2 n	Input Matrix	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S	*															MTR: 9 steps
D_1		*														
D_2		*	*	*												
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S: Head address of input device **D₁**: Head address of output device **D₂**: Head address of matrix scan **n**: Number of arrays in the matrix

Explanations:

1. **S** is the source device of the matrix input and occupies 8 consecutive points.

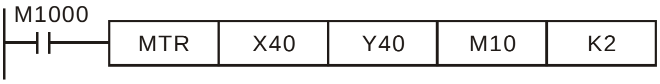
D_1 is the trigger device (transistor output Y) to read input signals and occupies **n** consecutive points

D_2 is the head address of the matrix which stores the read status from inputs

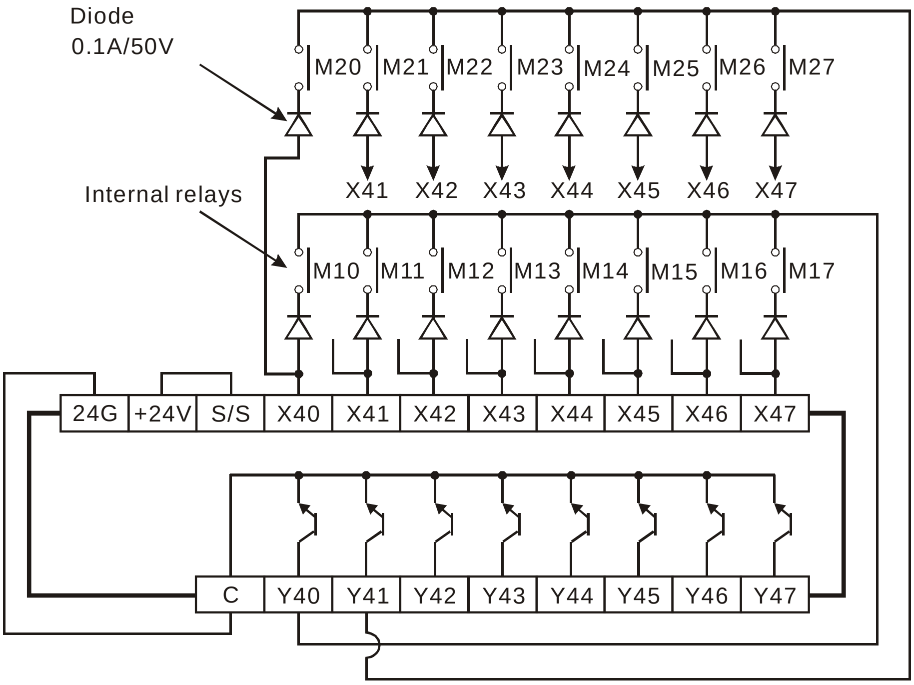
2. This instruction allows 8 continuous input devices starting from **S** to be used **n** times, which means the operation result can be displayed with a matrix table starting from **D_2**. Each set of 8 input signals are grouped into an "array" and there are **n** number of arrays. Each array is selected to be read by triggering output devices starting from **D_1**. The result is stored in a matrix-table which starts at corresponding head address **D_2**.
3. Maximum 8 arrays can be specified (**n** = 8) to obtain 64 input points (8 × 8 = 64).
4. The processing time of each array is approximately 25ms, i.e. an 8 array matrix would cost 200ms to finish reading. In this case, input signals with ON/OFF speed faster than 200ms are not applicable in the matrix input.
5. It is recommended to use special auxiliary relay M1000 (normally open contact).
6. Whenever this instruction finishes a matrix scan, M1029 will be ON for one scan period..
7. There is no limitation on the number of times for using the instruction, but only one instruction can be executed in the same time.
8. Flag: M1029, execution completed flag.

Program Example:

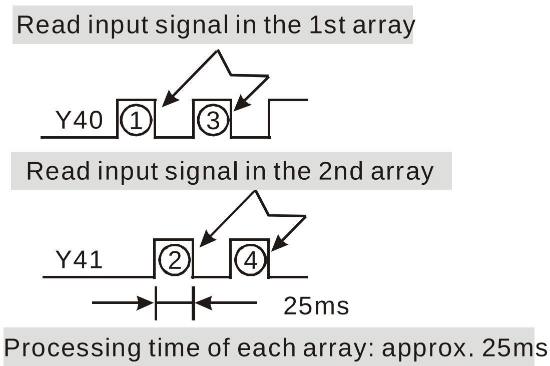
When PLC runs, MTR instruction executes. The status of input points X40~X47 is read 2 times in the driven order of output points Y40 and Y41, i.e. 16 signals will be generated and stored in internal relay M10~M17 and M20~M27.



The figure below illustrates the external wiring of the 2-array matrix input loop constructed by X40 ~ X47 and Y40 ~ Y41. The 16 switches correspond to the internal relays M10 ~ M17, M20 ~ M27. The wiring should be applied with MTR instruction.



When output Y40 is ON, only inputs in the first array are read. The results are stored in auxiliary relays M10~M17. After Y40 goes OFF, Y41 turns ON. This time only inputs in the second array are read. The results are stored in M20~M27.



Points to note:

1. Operand **S** must be a multiple of 10, e.g. 00, 10, 20, which means X0, X10… etc. and occupies 8 continuous devices.
2. Operand **D_1** should be a multiple of 10, i.e. 00, 10, 20, which means Y0, Y10… etc. and occupies **n** continuous devices
3. Operand **D_2** should be a multiple of 10, i.e. 00, 10, which means M0, M10, S0, S10… etc.
4. Valid range of **n** = 2~8

API	Mnemonic			Operands	Function	Controllers			
53	D	HSCS		(S1) (S2) (D)	High Speed Counter Set	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
S_1					*	*	*	*	*	*	*	*	*	*		DHSCS: 13 steps
S_2												*				
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S_1: Comparative value **S_2**: No. of high-speed counter **D**: Compare result

Explanations:

1. Functions related to high-speed counters adopt an interrupt process; therefore, devices specified in **D** which indicates comparison results are updated immediately. This instruction compares the present value of the designated high-speed counter **S_2** against a specified comparative value **S_1**. When the current value in counters equals **S_1**, device in **D** will be ON even when values in **S_1** and **S_2** are no longer equal.
2. If **D** is specified as Y0~Y3, Y4 and Y6, when the instruction is executed and the count value equals to **S_1** , the compare result will immediately output to the external outputs Y0~Y3, Y4 and Y6. However, other Y outputs will still be updated till the end of program. Also, M and S devices, not affected by the program scan time, will be immediate updated as the Y devices specified by this instruction. For EC5, the available Y outputs are Y0, Y2, Y4 and Y6.
3. Operand **D** can designate I0□0, □=1~8
4. High speed counters include software high speed counters and hardware high speed counters. In addtiion, there are also two types of comparators including software comparators and hardware comparators. For detailed explanations of high speed counters please refer to section 2.12 in this manual.
5. Explanations on software comparators for DHSCS/DHSCR instruction:
   - There are 6 software comparators for the high-speed compare Set/Reset.
   - There are 6 software comparators available corresponding to associated high speed counter interrupts. Numbers of the applied interrupts should also be specified correctly in front of the associated interrupt subroutines in the program.
   - When programming DHSCS and DHSCR instructions, the total of Set/Reset comparisons for both instructions can not be more than 6, otherwise syntax check error will occur.

➢ Table of settings for the high-speed interrupts of the software counters and software comparators:

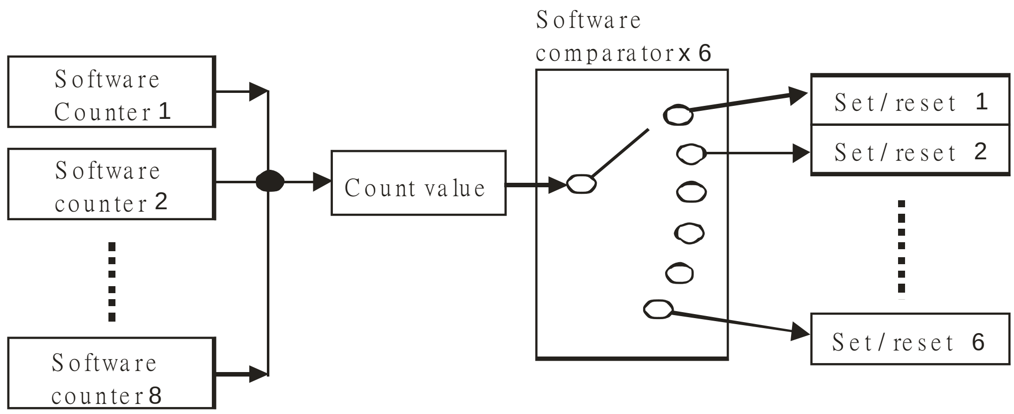
Counter	C232	C233	C234	C235	C236	C237
DHSCS High-speed interrupt	I010	I050	I070	I010	I020	I030
High-speed comparator Set	C232~C242 share 6 software comparators					

Counter	C238	C239	C240	C241	C242
DHSCS High-speed interrupt	I040	I050	I060	I070	I080
High-speed comparator Set	C232~C242 share 6 software comparators				

➢ DVP-SS2/SA2/SX2/SE does not support the software high speed counter C232.

➢ For DVP-12SE (FW: V1.xx), C253 and C254 are software high speed counters. The high-speed interrupt is I030. For DVP-12SE (FW: V2.00), C253 and C254 are upgraded to hardware speed counters.

➢ Block diagram of software counters and comparators:



6. Explanations on hardware comparators DHSCS/DHSCR instruction:

➢ There are 2 groups of hardware comparators provided respectively for 2 groups of hardware counters (A group and B group), and each group shares 4 comparators with individual Compare Set/Reset function.

➢ When programming DHSCS and DHSCR instructions, the total of Set/Reset comparisons for both instructions can not be more than 4, otherwise syntax check error will occur.

➢ Each high-speed counter interrupt occupies an associated hardware comparator, consequently the interrupt number can not be repeated. Also, I010~I040 can only be applied for group A comparators and I050~I080 for group B.

➢ If DCNT instruction enables C243 as high speed counter (group A) and DHSC/DHSC instruction uses C245 as high speed counter (group A) at the same time, PLC takes C243 as the source counter automatically and no syntax check error will be detected.

- Designers have to specify the comparison value of a hardware comparator before they enable a comparison instruction. If the comparison value of a hardware comparator has to be changed after a comparison instruction is enabled, it is suggested that users should disable the comparison instruction first. After the users specify a new comparison value, the users can enable the comparison instruction again.
- If users want to change the value of a hardware comparator without disabling the high-speed comparison instruction which is being used, they have to check whether the model used support this operation. The models which support this operation are listed below.

Model name	ES2/EX2	EC5	SS2	SA2	SX2	SE
Version	V3.20	V1.00	V3.00	V2.60	V2.40	V1.00

Note: If the comparative value changes, it will not be stored in the hardware comparator until the instruction is scanned.

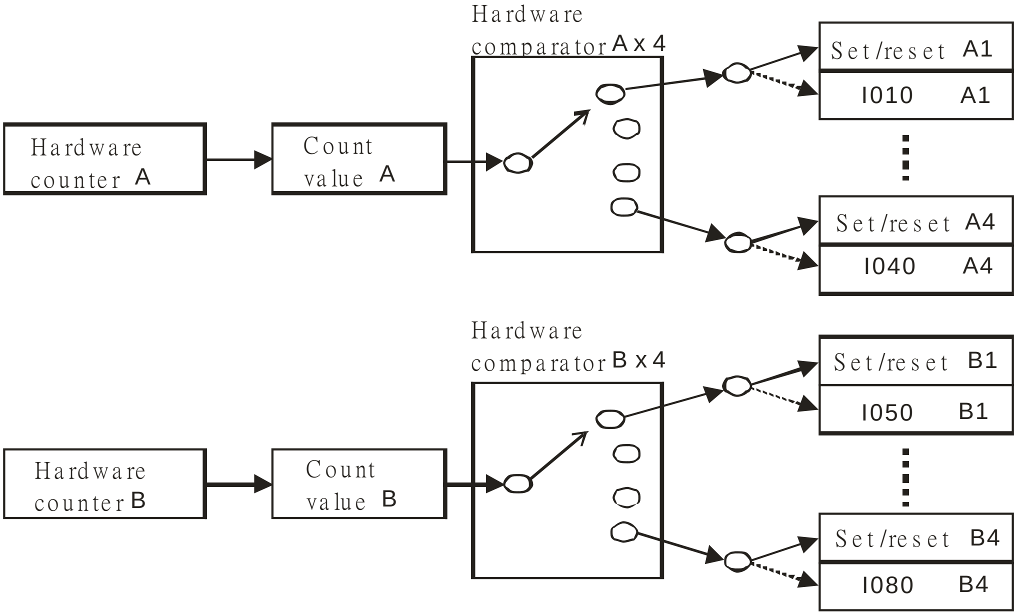
- Table of settings for the high-speed interrupts of hardware counters and comparators: applicable to DVP-12SE (FW V2.00 or later) and DVP-26SE.

Hardware counter	A group				B group			
	A1	A2	A3	A4	B1	B2	B3	B4
Counter No.	C243, C245~C248, C251,C252				C244, C249, C250, C253, C254			
High-speed counter interrupt	I010	I020	I030	I040	I050	I060	I070	I080
High-speed compare Set/Reset	Share 4 hardware comparators for group A				Share 4 hardware comparators for group B			

- Table of settings for the high-speed interrupts of hardware counters and comparators: only applicable to DVP-12SE (FW V1.xx).

Hardware counter	A group		B group	
	A1	A2	B1	B2
Counter No.	C243, C245~C248, C251,C252		C244	
High-speed counter interrupt	I010	I020	I050	I060
Hi-speed compare Set/Reset	Share 2 hardware comparators for group A		Share 2 hardware comparators for group B	

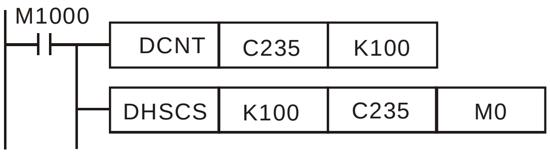
- Block diagram of hardware counters and comparators:



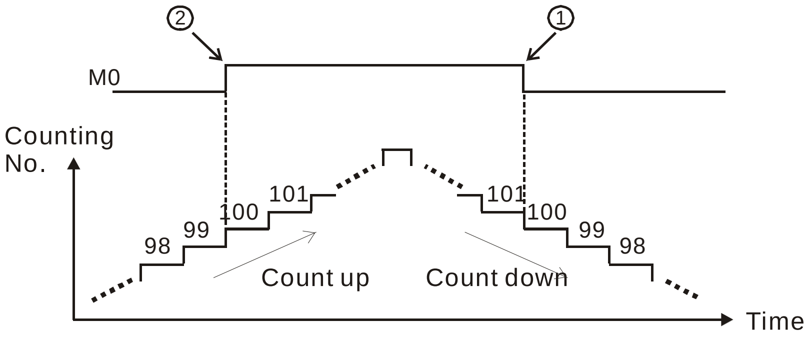
7. Difference between software and hardware comparators, applicable to DVP-12SE (FW V2.00 or later) and DVP-26SE:
   - 6 comparators are available for software counters while 8 comparators are available for 2 groups of hardware counters (4 comparators for each group)
   - Output timing of software comparator → count value equals to comparative value in both counting up/down modes.
   - Output timing of the hardware comparator with firmware version 1.xx → count value equals to comparative value+1 in counting-up mode; count value equals to comparative value -1 in counting-down mode.
   - Output timing of the hardware comparator with firmware version 2.00 and above → count value equals to comparative value in both counting up/down modes.
8. Difference between software and hardware comparators, applicable to DVP-12SE FW V1.xx:
   - 6 comparators are available for software counters while 4 comparators are available for 2 groups of hardware counters (2 comparators for each group)
   - Output timing of software comparator → count value equals to comparative value in both counting up/down modes.
   - Output timing of the hardware comparator → count value equals to comparative value+1 in counting-up mode; count value equals to comparative value -1 in counting-down mode.
   - Once the instruction is executed, the comparative value cannot be changed. If you need to modify the comparative value, you need to close and then open the high-speed comparison instruction again.

Program Example 1:

Set/reset M0 by applying software comparator

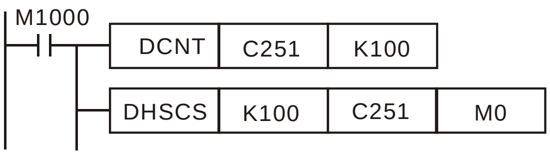


- When value in C235 varies from 99 to100, DHSCS instruction sets M0 ON. (M1235 = OFF, C235 counts up)
- When value in C235 varies from 101 to100, DHSCR instruction resets M0. (M1235 = ON, C235 counts down)
- Timing diagram for the comparison:

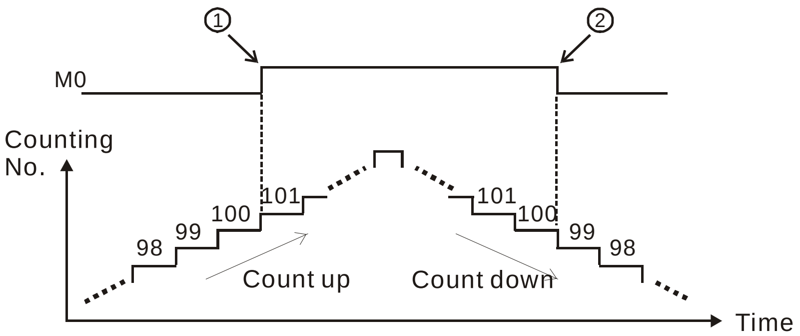


Program Example 2:

Set/reset M0 by applying hardware comparator

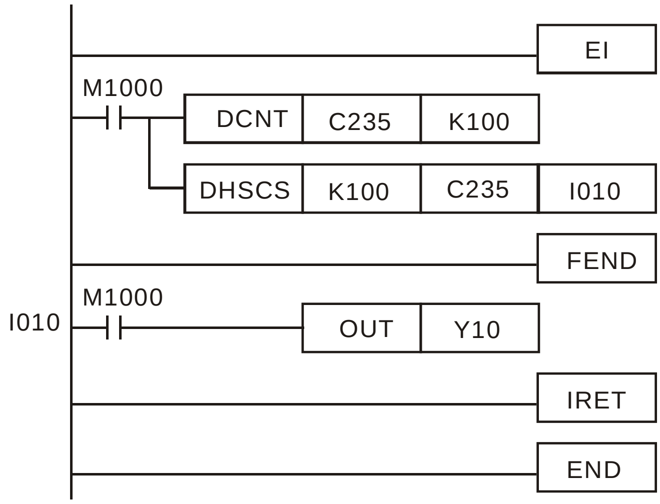


- When C251 counts up and the value in C251 varies from 100 to101, DHSCS instruction sets M0 ON.
- When C251 counts down and the value in C251 varies from 100 to 99, DHSCR instruction resets M0.
- Timing diagram for the comparison:



Program Example 3:

Executes interrupt subroutine by applying software comparator.



- When value in C235 varies from 99 to100, interrupt subroutine triggered by I010 executes immediately to set Y0 ON.

Points to note:

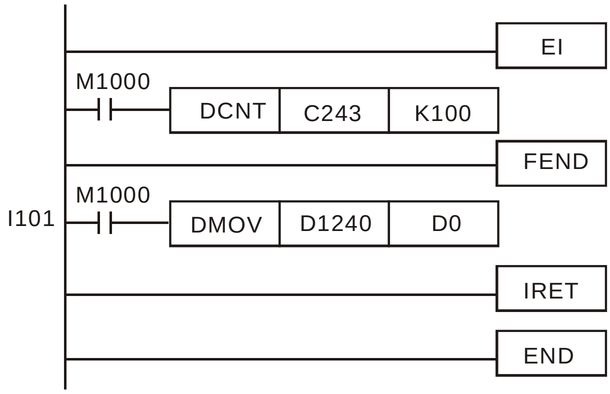
- If operand **D** is specified as S, M or Y0~Y3 for the above high speed comparison, the compare result will immediately output to the external points Y0~Y3 (Y0~Y5 for SS2/SX2). However, if D is specified as Y4~Y337, external outputs will be updated till the end of program (delay for one scan cycle).

1. Count value storage function of high speed interrupt:
   - When X1, X3, X4 and X5 is applied for reset function and associated external interrupts are disabled, users can define the reset function as Rising/Falling-edge triggered by special M relays specified in the table: Applicable Software High Speed Counters. However, if external interrupts are applied, the interrupt instructions have the priority in using the input points. In addition, PLC will move the current data in the counters to the associated data registers below then reset the counters
   - When X0 (counter input) and X1 (external Interrupt I100/I101) work with C243, the count value will be moved to D1240 and D1241 when interrupt occurs and then the counter will be reset.
   - When X2 (counter input) and X3 (external Interrupt I300/I301) work with C244, the count value will be moved to D1242 and D1243 when interrupt occurs and then the counter will be reset.
   - When X0 (counter input) and X4 (external Interrupt I400/I401) work with C246, C248, C252, the count value will be moved to D1240 and D1241 when interrupt occurs and then the counter will be reset.

- When X2 (counter input) and X5 (external Interrupt I500/I501) work with C244, C250, C254, the count value will be moved to D1242 and D1243 when interrupt occurs and then the counter will be reset.

Special D	D1241, D1240				D1243, D1242		
Counter	C243	C246	C248	C252	C244	C250	C254
Interrupt	X1(I100/I101)	X4(I400/I401)			X3(I300/I301)	X5(I500/I501)	

Program Example 4:



- If interrupt I101 is triggered from input point X1 while C243 is counting, I101 interrupt subroutine executes immediately and the count value in C243 will be moved to D0. After this, C243 is reset.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
54	D	HSCR	S1 S2 D	High Speed Counter Reset				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DHSCR: 13 steps
S_1					*	*	*	*	*	*	*	*	*	*		
S_2												*				
D		*	*	*								*				

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC52	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S_1: Comparative value **S_2**: No. of high speed counter **D**: Comparison result

Explanations:

1. DHSCR compares the current value of the counter **S_2** against a compare value **S_1**. When the counters current value changes to a value equal to **S_1**, then device **D** is reset to OFF. Once reset, even if the compare result is no longer unequal, **D** will still be OFF.
2. If **D** is specified as Y0~Y3, Y4 and Y6, when the instruction is executed and the count value equals to **S_1** , the compare result will immediately output to the external outputs Y0~Y3, Y4 and Y6. However, other Y outputs will still be updated till the end of program. Also, M and S devices, not affected by the program scan time, will be immediate updated as the Y devices specified by this instruction. For EC5, the available Y outputs are Y0, Y2, Y4 and Y6.
3. Operand **D** can be specified with high speed counters C232~C254 (SS2/SA2/SX2/SE does not support C232) the same as **S_2**. .
4. High speed counters include software high speed counters and hardware high speed counters. In addtiion, there are also two types of comparators including software comparators and hardware comparators. For detailed explanations of high speed counters please refer to section 2.12 in this manual.
5. For explanations on software counters and hardware counters, please refer to API53 DHSCS.
6. For program examples, please refer to Program Example1 and 2 in API53 DHSCS.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
55	D	HSZ	(S1) (S2) (S) (D)	High Speed Zone Compare				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DHSZ: 17 steps
S_1					*	*	*	*	*	*	*	*	*	*		
S_2					*	*	*	*	*	*	*	*	*	*		
S												*				
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

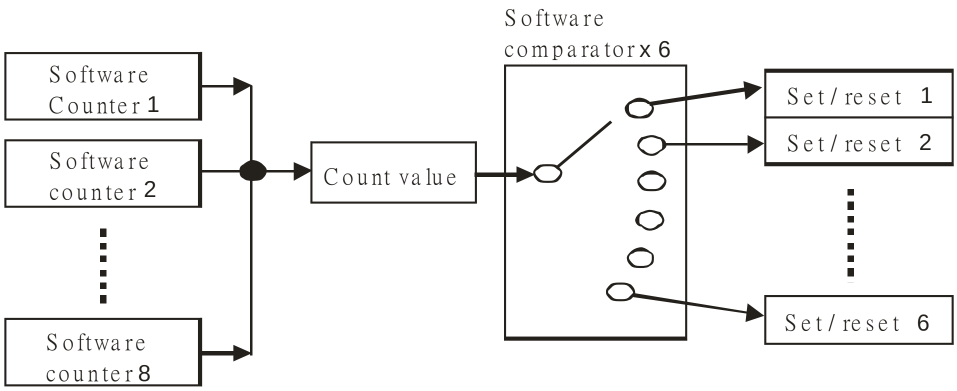
S_1: Lower bound of the comparison zone **S_2**: Upper bound of the comparison zone **S**: No. of high speed counter **D**: Comparison result (3 consecutive devices)

Explanations:

1. **S_1** should be equal to or smaller than **S_2** (**$S_1 \leqq S_2$**).
2. If **D** is specified as Y0~Y3 in this instruction, the compare result will immediately output to the external outputs Y0~Y3. However, other Y outputs will still be updated till the end of program. Also, M and S devices, not affected by the program scan cycle, will be immediately updated as well.
3. High speed counters include software high speed counters and hardware high speed counters. In addtiion, there are also two types of comparators including software comparators and hardware comparators. For detailed explanations of high speed counters please refer to section 2.12 in this manual.
4. Explanations on software comparators for DHSZ instruction
   - Corresponding table for software counters and comparators:

Counter	C232	C233	C234	C235	C236	C237	C238	C239	C240	C241	C242
Hi-speed compare Set/Reset	Share 6 software comparators										

- Block diagram of software counters and comparators:



- There are 6 software zone comparators available exclusively for zone compare operation, hence the limit of 6 comparisons for zone compare does not include the comparisons of DHSCS and DHSCR.
- SS2/SA2/SX2/SE does not support software counter C232.

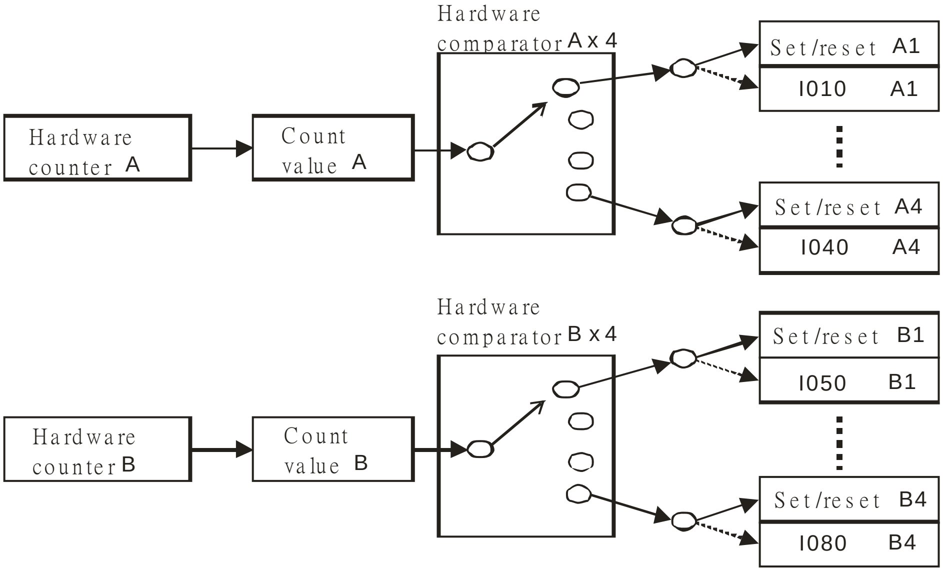
5. Explanations on hardware comparators for HSZ instruction:
   - Corresponding table for hardware counters and comparators (It is not applicable to VEP-12SE):

Hardware counter	A group				B group			
	A1	A2	A3	A4	B1	B2	B3	B4
Counter No.	C243, C245~C248, C251,C252				C244, C249, C250, C253, C254			
High-speed compare Set/Reset	Shares 4 hardware comparators for group A				Shares 4 hardware comparators for group B			

   - Corresponding table for hardware counters and comparators (It is only applicable to VEP-12SE):

Hardware counter	A group		B group	
	A1	A2	B1	B2
Counter No.	C243, C245~C248, C251,C252		C244	
High-speed compare Set/Reset	Shares 2 hardware comparators for group A		Shares 2 hardware comparators for group B	

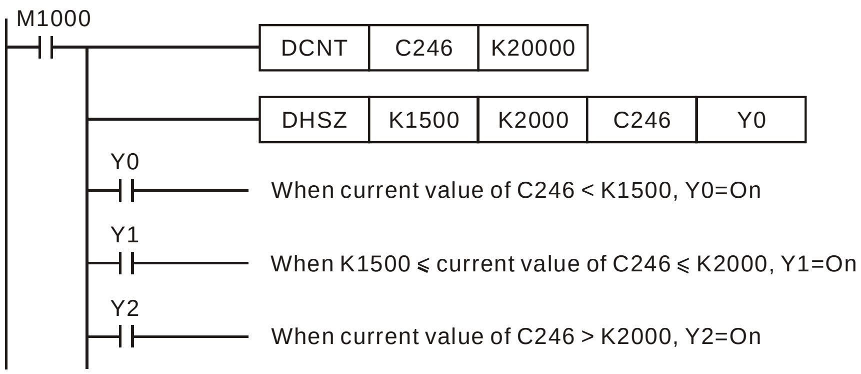
- Block diagram of hardware counters and comparators:



- The two groups can only be used once for each group, occupying 2 comparators. For example, when DHSZ instruction uses A3 and A4 of group A comparators, only the other 2 comparators (A1, A2) are available for DHSCS and DHSCR instructions.
- When DHSCS uses I030 or I040, comparators A3 and A4 are no longer available for DHSZ instruction. Also, when DHSCS uses I070 or I080, comparators B3 and B4 are no longer available for DHSZ instruction. If comparators are used repeatedly, the syntax error will be detected on the instruction behind.
- For DVP-SE, if DHSZ instruction uses hardware comparators, two hardware comparators are used. DHSCS instruction and DHSCR instruction can not use the same hardware comparators.

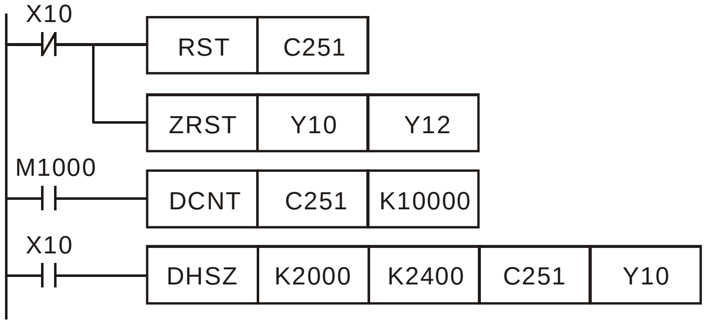
Program Example 1: (Applying Hardware High Speed Counter)

1. When **D** is specified as Y0, then Y0~Y2 will be occupied automatically.
2. When DHSZ is executed, the instruction compares the current value in C246 with the upper/lower bound (1500/2000) of the comparison zone, and Y0~Y2 will be ON according to the comparison result.

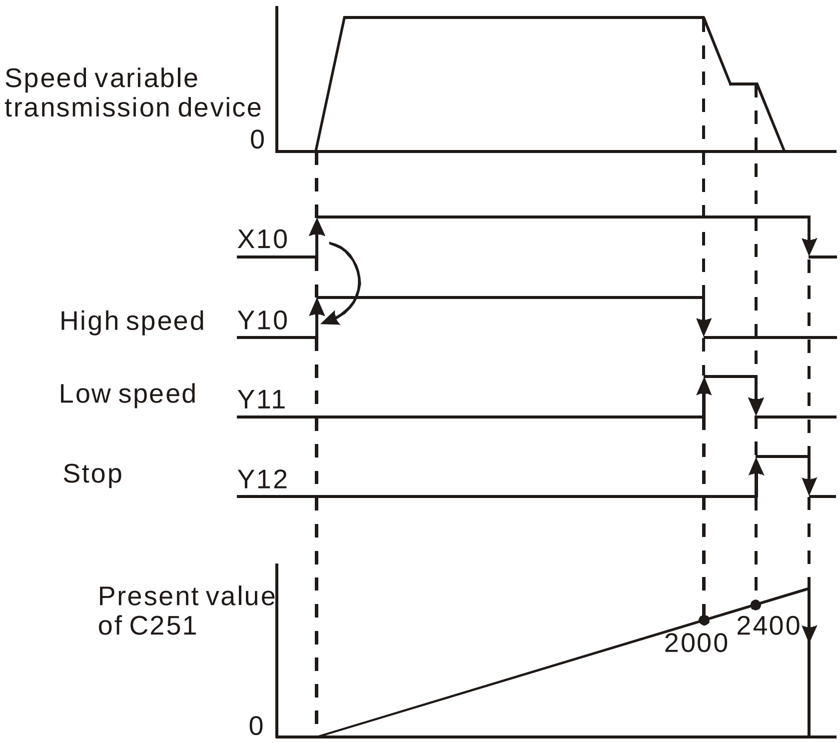


Program Example 2: (Applying DHSZ instruction for performing ramp down operation)

1. C251 is AB-phase high speed counter. When X10 = ON, DHSZ compare the present value with K2000. Present value≦K2000, Y10 = ON.
2. When X10 = OFF, Y10~Y12 are reset.



Timing diagram



API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
56	SPD	S1 S2 D	Speed Detection				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S1	*															SPD: 7 steps
S2					*	*	*	*	*	*	*	*	*	*	*	
D											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S1: External pulse input **S2**: Pulse receiving time (ms) **D**: Detected result (5 consecutive devices)

Explanations:

1. The instruction counts the number of pulses received at input terminal **S1** during the time **S2** (ms) and stores the result in the register **D**.
2. ES2/EX2 before V0.92. External pulse input terminals designated in **S1** :

Available input points	X0, X2	X1 (X0/X1)	X6, X7
Input mode	1-phase input (Supports single frequency)	AB-phase input (Supports quadruple frequency)	1-phase input (Supports single frequency)
Max frequency	100KHz	5KHz	10KHz

3. ES2/EX2/EC5: V1.00 or later. External pulse input terminals designated in **S1** :

Available input points	X0, X2	X1 (X0/X1), X3 (X2/X3) X5 (X4/X5), X7 (X6/X7)	X4, X6
Input mode	1-phase input (Supports single frequency)	AB-phase input (Supports quadruple frequency)	1-phase input (Supports single frequency)
Max frequency	EC5: 50KHz ES2: 100KHz	5KHz	10KHz

4. SS2/SA2/SX2/12SE. External pulse input terminals designated in **S1** :

Available input points	X0, X2	X1 (X0/X1), X3 (X2/X3) X5 (X4/X5), X7 (X6/X7)	X4, X6
Input mode	1-phase input (Supports single frequency)	AB-phase input (Supports quadruple frequency)	1-phase input (Supports single frequency)
Max frequency	SA2/SE/SX2: 100kHz SS2: 20kHz	5KHz. X1(X0/X1) of SA2/12SE: 30kHz	10KHz

5. **D** occupies 5 consecutive registers, **D** + 1 and **D** store the results of previous pulse detection; **D** +3 and **D** + 2 store the current accumulated number of pulses; **D** + 4 store the current time remaining (max. 32,767ms).

6. If any of the X0~7 is used in SPD instruction, its associated high-speed counters or external interrupts cannot be used in this instruction or other instructions. The same input can only be used once at a time.
7. For ES2/EX2 before V0.92: when X0, X2, X6 and X7 are used, they will be detected as 1-phase input. When X1 is used, X0(A) and X1(B) will be applied together as AB-phase input.
8. For EC5/SS2/SA2/SX2/SE/ES2/EX2: V1.00 or later: when X0, X2, X4 and X6 are used, they will be detected as 1-phase input. When X1, X3, x5, X7 are used, X0, X2, X4, X6 will be applied together as AB-phase input.
9. This instruction is mainly used to obtain the value of rotation speed and the results in D are in proportion to the rotation speed. Rotation speed **N** can be calculated by the following equation

$$N=\frac{60(D0)}{nt}\times 10^{3}(rpm)$$

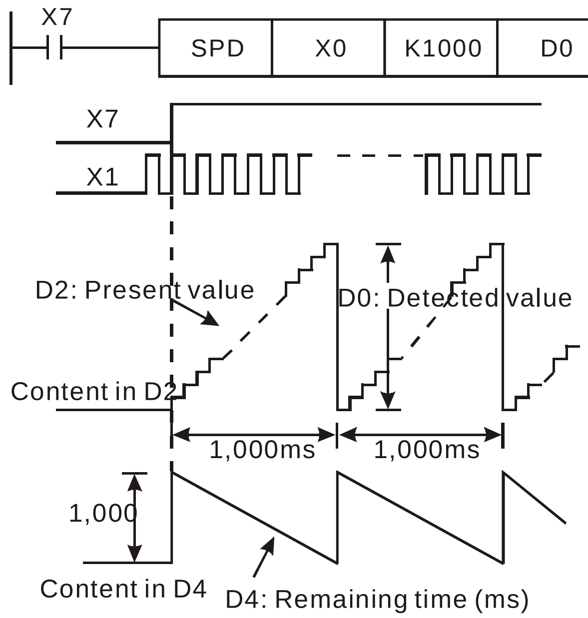
N: Rotation speed

n: The number of pulses produced per rotation

t: Detecting time specified by $\mathbf{S_2}$ (ms)

Program Example:

1. When X7 = ON, D2 stores the high-speed pulses at X0 for 1,000ms and stops automatically. The results are stored in D0, D1.
2. When the 1000ms of counting is completed, D2 will be reset. When X7 turns ON again, D2 starts counting again.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
57	D	PLSY		S1 S2 D	Pulse Output				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	PLSY: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DPLSY: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D		*														

PULSE				16-bit				32-bit			
ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Operands:

S_1: Pulse output frequency **S_2**: Number of output pulses **D**: Pulse output device (Y0 ~ Y3, Y4, Y6 available)

Explanations:

(For ES2 / EX2 / SS2 / SA2 / SE / SX2 Series PLC CPU)

1. When PLSY instruction has been executed, the specified quantity of pulses **S_2** will be output through the pulse output device **D** at the specified pulse output frequency **S_1**
2. **S_1** specifies the pulse output frequency

Output frequency range of MPU			
range	Output	Y0, Y2	Y1, Y3
	16-bit instruction	SS2: 0~10,000Hz ES2/EX2/SA2/SX2/SE: 0~32,767 Hz	0~10,000Hz
	32-bit instruction	SS2: 0~10,000Hz ES2/EX2/SA2/SX2/SE: 0~100,000 Hz	0~10,000Hz
If frequency equals or smaller than 0Hz is specified, pulse output will be disabled. If frequency bigger than max frequency is specified, PLC will output with max frequency.			

3. **S_2** specifies the number of output pulses.
   16-bit instruction: -32,768~32,767. 32-bit instruction: -2,147,483,648~2,147,483,647.
   When **S_2** is specified as K0, the pulse will be output continuously regardless of the limit of pulse number.
4. When D1220/D1221 = K0, K1, K2 or K3, the positive/negative sign of **S_2** denotes pulse output direction (Positive/negative).
5. Four pulse output modes:

Mode / Output	D1220						D1221					
	K0		K1	K2	K3		K0		K1	K2	K3#	
Y0	Pulse		Pulse	A	CW							
Y1		Pulse	Dir	B		Pulse						
Y2							Pulse		Pulse	A	CCW	
Y3								Pulse	Dir	B		Pulse

Pulse: Pulse A: A phase pulse CW: clockwise
Dir: Direction B: B phase pulse CCW: Counter-clockwise

Note #: Available for 12SE: FW V2.02 or later. When D1220 is specified as K3, D1221 is invalid.

6. Pulse output flags:

Output device	Y0	Y1	Y2	Y3
Completed Flag	M1029	M1030	M1102	M1103
Immediately pause	M1078	M1079	M1104	M1105
0.01~10Hz output	M1190	M1191	M1192	M1193

a) M1029 = ON after Y0/Y1 (D1220=K1, pulse/Dir) output is completed.
   M1102 = ON after Y2/Y3 (D1221=K1, pulse/Dir) output is completed.
   M1029 = ON after the Y0/Y2 (D1220 = K3, CW/CCW) output is completed.

b) The execution completed flag M1029, M1030, M1102, and M1103 should be manually reset by users after pulse output is completed.

c) When PLSY / DPLSY instruction is OFF, the pulse output completed flags will all be reset.

d) When M1190~M1193 = ON, the available output range for PLSY Y0~Y3 is 0.01~10Hz.

7. While the PLSY instruction is being executed, the output will not be affected if **S_2** is changed. To change the pulse output number, stop the PLSY instruction, then change the pulse number.
8. **S_1** can be changed during program execution and the change will take effects until the modified PLSY instruction is being executed.
9. The ratio of OFF time and ON time of the pulse output is 1:1.
10. If operand **S_1**, **S_2** use index F, only 16-bit instruction is available.
11. There is no limitation on the times of using this instruction, however the program allows only **4** instructions (PLSY, PWM, PLSR) to be executed at the same time. If Y1 is used for several high speed pulse output instructions, PLC will output according to the execution order of these instructions.

(For EC5 Series PLC CPU)

1. When PLSY instruction has been executed, the specified quantity of pulses **S_2** will be output through the pulse output device **D** at the specified pulse output frequency **S_1**
2. **S_1** specifies the pulse output frequency

Output frequency range of EC5 Series PLC CPU		
range	Output	Y0, Y2, Y4, Y6
	16-bit instruction	0~32,767 Hz
	32-bit instruction	0~50,000 Hz
If frequency equals or smaller than 0Hz is specified, pulse output will be disabled. If frequency bigger than max frequency is specified, PLC will output with max frequency.		

3. **S_2** specifies the number of output pulses.
   16-bit instruction: -32,768~32,767. 32-bit instruction: -2,147,483,648~2,147,483,647.
   When **S_2** is specified as K0, the pulse will be output continuously regardless of the limit of pulse number.

4. When D1220/D1221/D1341/D1342= K0, K1,or K2, the positive/negative sign of **S2** denotes pulse output direction (Positive/negative).

5. Four pulse output modes:

Mode / Output	D1220			D1221			D1341			D1342		
	K0	K1	K2	K0	K1	K2	K0	K1	K2	K0	K1	K2
Y0	Pulse	Pulse	A									
Y1		Dir	B									
Y2				Pulse	Pulse	A						
Y3					Dir	B						
Y4							Pulse	Pulse	A			
Y5								Dir	B			
Y6										Pulse	Pulse	A
Y7											Dir	B

Pulse: Pulse   A: A phase pulse
Dir: Direction   B: B phase pulse

6. Pulse output flags:

Output device	Y0	Y2	Y4	Y6
Completed Flag	M1029	M1102	M1321	M1322
Immediately pause	M1078	M1104	M1310	M1311

a) M1029 = ON after Y0/Y1 (D1220=K1, pulse/Dir) output is completed.
M1102 = ON after Y2/Y3 (D1221=K1, pulse/Dir) output is completed.
M1321 = ON after the Y4/Y5 (D1341 = K1, pulse/Dir) output is completed.
M1322 = ON after the Y6/Y7 (D1342 = K1, pulse/Dir) output is completed.

b) The execution completed flag M1029, M1030, M1321, and M1322 should be manually reset by users after pulse output is completed.

c) When PLSY / DPLSY instruction is OFF, the pulse output completed flags will all be reset.

7. While the PLSY instruction is being executed, the output will not be affected if **S2** is changed. To change the pulse output number, stop the PLSY instruction, then change the pulse number.

8. **S1** can be changed during program execution and the change will take effects until the modified PLSY instruction is being executed.

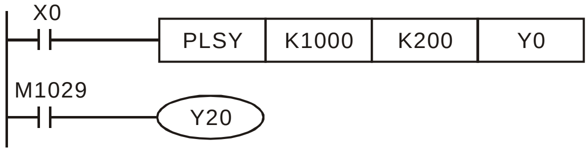
9. The ratio of OFF time and ON time of the pulse output is 1:1.

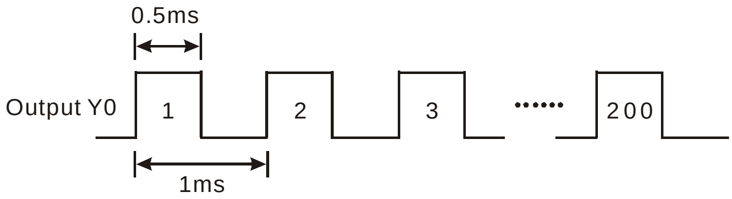
10. If operand **S1**, **S2** use index F, only 16-bit instruction is available.

11. There is no limitation on the times of using this instruction, however the program allows only **4** instructions (PLSY, PWM, PLSR) to be executed at the same time. If Y1 is used for several high speed pulse output instructions, PLC will output according to the execution order of these instructions.

Program Example:

1. When X0 = ON, 200 pulses of 1kHz are generated from output Y0, after the pulse output has been completed, M1029 = ON to set Y20.
2. When X0 = OFF, pulse output Y0 will immediately stop. When X0 turns ON again, the pulse output will start from the first pulse.





Points to note:

1. Description of associated flags:

M1029: M1029 = ON when Y0 pulse output is completed.

M1030: M1030 = ON when Y1 pulse output is completed. (Not available for EC5)

M1102: M1102 = ON when Y2 pulse output is completed.

M1103: M1103 = ON when Y3 pulse output is completed. (Not available for EC5)

M1321: M1321= ON when Y4 pulse output is completed.

M1322: M1322 = ON when Y6 pulse output is completed.

M1078: Y0 pulse output pause (immediately)

M1079: Y1 pulse output pause (immediately) (Not available for EC5)

M1104: Y2 pulse output pause (immediately)

M1105: Y3 pulse output pause (immediately) (Not available for EC5)

M1310: Y4 pulse output pause (immediately)

M1311: Y6 pulse output pause (immediately)

M1190: Se t Y0 high speed output as 0.01~10Hz. (Not available for DVP-12SE / EC5)

M1191: Se t Y1 high speed output as 0.01~10Hz. (Not available for DVP-12SE / EC5)

M1192: Se t Y2 high speed output as 0.01~10Hz. (Not available for DVP-12SE / EC5)

M1193: Se t Y3 high speed output as 0.01~10Hz. (Not available for DVP-12SE / EC5)

M1347: Auto reset Y0 when high speed pulse output completed.

M1348: Auto reset Y1 when high speed pulse output completed. (Not available for EC5)

M1524: Auto reset Y2 when high speed pulse output completed.

M1525: Auto reset Y3 when high speed pulse output completed. (Not available for EC5)

M1526: Auto reset Y4 when high speed pulse output completed.

M1527: Auto reset Y6 when high speed pulse output completed.

M1538: Indicating pause status of Y0

M1539: Indicating pause status of Y1 (Not available for EC5)

M1540: Indicating pause status of Y2

M1541: Indicating pause status of Y3 (Not available for EC5)

M1542: Indicating pause status of Y4

M1543: Indicating pause status of Y6

2. Description of associated special D registers:

   D1030: Present number of Y0 output pulses (Low word).

   D1031: Present number of Y0 output pulses (High word).

   D1032: Present number of Y1 output pulses (Low word). (Not available for EC5)

   D1033: Present number of Y1 output pulses (High word). (Not available for EC5)

   D1336: Present number of Y2 output pulses (Low word).

   D1337: Present number of Y2 output pulses (High word).

   D1338: Present number of Y3 output pulses (Low word). (Not available for EC5)

   D1339: Present number of Y3 output pulses (High word). (Not available for EC5)

   D1375: Present number of Y4 output pulses (Low word).

   D1376: Present number of Y4 output pulses (High word).

   D1377: Present number of Y6 output pulses (Low word).

   D1378: Present number of Y6 output pulses (High word).

   D1220: Output mode selection for pulse output (Y0,Y1), refer to the instruction explanations.

   D1221: Output mode selection for pulse output (Y2, Y3), refer to the instruction explanations.

   D1341: Output mode selection for pulse output (Y4, Y), refer to the instruction explanations.

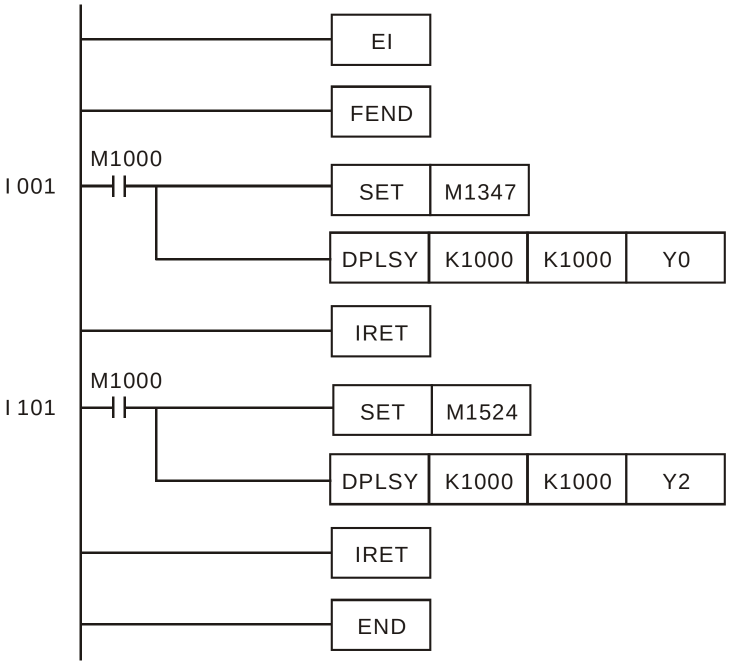
   D1342: Output mode selection for pulse output (Y6, Y7), refer to the instruction explanations.

3. More explanations for M1347,M1348, M1524, M1525, M1526, M1527:

   Generally when pulse output is completed, PLSY instruction has to be reset so that the instruction can start pulse output one more time. When M1347, M1348, M1524, M1525, M1526 and M1527 is enabled, the associated output terminals will be reset automatically when pulse output is completed, i.e. the PLSY instruction is reset. When PLC scans PLSY instruction again, the pulse output starts automatically. In addition, PLC scans the mentioned 6 flags after END instruction, hence PLSY instruction in continuous pulse output mode requires a delay time of one scan cycle for next pulse output operation.

   The function is mainly used in subroutines or interrupts which require high speed pulse output. Here are some examples:

Program Example 1:

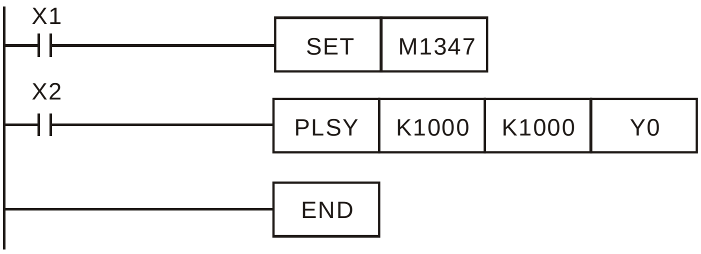


Explanations:

a) Whenever I001 is triggered, Y0 will output 1,000 pulses; whenever I101 is triggered, Y2 will output 1,000 pulses.

b) When pulse output is completed, there should be an interval of at least one scan cycle before next pulse output operation is triggered. .

Program Example 2:



Explanation:

When both X1 and X2 are ON, Y0 pulse output will operate continuously. However, there will be a delay of approx. 1 scan cycle every 1000 pulses.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
58	PWM	S1 S2 D	Pulse Width Modulation				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*	*	PWM: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D		*														

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Pulse output width (ms)   **S_2**: Pulse output cycle (ms)

D: Pulse output device (Y0, Y1, Y2,Y3, Y4, Y6)

Explanations:

1. **S_1** is specified as pulse output width (t). **S_2** is specified as pulse output cycle (T).

   Rule: **S1** ≦ **S2**. (It is not applicable to EC5 and DVP-12SE: FW V1.xx.)

Reference Table for Output Cycle and Output Width					
Range of pulse output width / cycle	Output	Y0	Y2	Y1	Y3
	t	0~10000		0~32767	
	T	1~10000		1~32,767	
Flag for switching unit		M1112	M1113	M1070	M1071
		On=10us；Off=100us		On=100us；Off=1ms	
Flag for high-speed output		M1116 is ON. (Unit: 1us)		M1117 is ON. (Unit: 10us)	

2. **S_1** is specified as pulse output width (t). **S_2** is specified as pulse output cycle (T).

   Rule: **S1** ≦ **S2**. (for DVP-12SE: FW V1.xx.)

Reference Table for Output Cycle and Output Width					
Range of pulse output width / cycle	Output	Y0	Y1	Y2	Y3
	t	0~10000	0~32767		
	T	1~10000	1~32767		
Flag for switching unit		M1112	M1070	M1113	M1071
		On=10us Off=100us	On=100us Off=1ms		

S_1 is specified as pulse output width (t). **S_2** is specified as pulse output cycle (T).

Rule: **S1** ≦ **S2**. (for EC5)

Reference Table for Output Cycle and Output Width					
Range of pulse output width / cycle	Output	Y0	Y2	Y4	Y6
	t	0~10000			
	T	1~10000			
Flag for switching unit		M1112	M1113	M1530	M1531
		On=10 us Off=100 us			

3. Pulse output devices for operand D: Y0, Y1, Y2, Y3. For EC5, the available pulse output devices for operand D are Y0, Y2, Y4 and Y6.
4. When several pulse output instructions (PLSY, PWM, PLSR) use Y1 or Y3 as the output device in the same scan cycle, PLC will perform the instruction which is executed first.
5. When $\mathbf{S_1} \leqq 0$, $\mathbf{S_2} \leqq 0$ or $\mathbf{S_1} > \mathbf{S_2}$, errors will occur (M1067 and M1068 will not be ON) and no output will be generated from pulse output devices. When $\mathbf{S_1} = \mathbf{S_2}$, the pulse output device will be ON continuously.
6. $\mathbf{S_1}$, $\mathbf{S_2}$ can be changed when PWM instruction is being executed. But the output time unit (M flags) cannot be changed during the execution of PWM instruction. You need to set the output time unit before the execution of PWM instruction.
7. When M1112 = ON, the unit of Y0 output pulse is 10μs, when M1112 = OFF, the unit is 100μs.
8. When M1070 = ON, the unit of Y1 output pulse is 100μs, when M1070 = OFF, the unit is 1ms. (Not available for EC5)
9. When M1113 = ON, the unit of Y2 output pulse is 10μs, when M1113 = OFF, the unit is 100μs. (Not available for DVP-12SE.)
10. When M1113 = ON, the unit of Y2 output pulse is 100μs, when M1113 = OFF, the unit is 1ms. (It is only applicable to DVP-12SE.)
11. When M1071 = ON, the unit of Y3 output pulse is 100μs, when M1071 = OFF, the unit is 1ms. (Not available for EC5)
12. For EC5, when M1530 = ON, the unit of Y4 output pulse is 10 μs, when M1530 = OFF, the unit is 100 μs.
13. For EC5, when M1531 = ON, the unit of Y6 output pulse is 10 μs, when M1531 = OFF, the unit is 100 μs.
14. When M1116 is ON, M1112 and M1113 do not work. The time unit of the pulse output through Y0 and Y2 is 1μs. Availabilities are shown below:

Model	ES2/EX2	ES2-C	ES2-E	12SA2	SX2	SS2	12SE	26SE	28SA2
Version	V3.00	V3.20	V1.00	2.60	2.40	2.80	V2.02	V1.0	V2.88

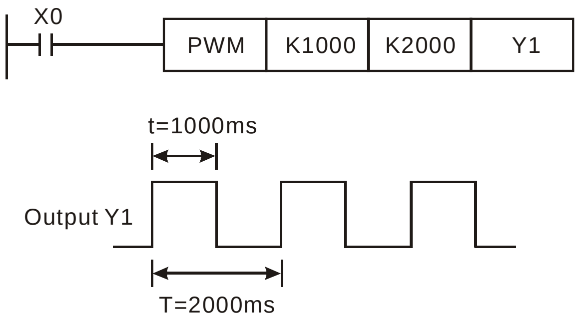
15. When M1117 is ON, M1070 and M1071 do not work. The time unit of the pulse output through Y1 and Y3 is 10μs. Availabilities are shown below:

Model	ES2/EX2	ES2-C	ES2-E	12SA2	SX2	SS2	12SE	26SE	28SA2
Version	V3.00	V3.20	V1.00	2.60	2.40	2.80	V2.02	V1.0	V2.88

16. If M1116 for DVP-SS2 is enabled, the minimum pulse output width should be larger than 20. Otherwise, due to the limitations on the hardware bandwidth of Y0 and Y2, the output result is not the correct time width.

Program Example:

When X0 = ON, Y1 output the pulse as shown opposite. When X0 = OFF, output Y1 turns OFF.



Note:

1. Flag description:

M1070: For PWM instruction, when M1070 = ON, the unit for Y1 output pulse to switch is 100 µs, when M1070 = OFF, the unit is 1 ms. (Not available for EC5)

M1071: For PWM instruction, when M1071 = ON, the unit for Y3 output pulse to switch is 100 µs, when M1070 = OFF, the unit is 1 ms. (Not available for EC5)

M1112: For PWM instruction, when M1112 = ON, the unit for Y0 output pulse to switch is 10 µs (for SE: 100 µs), when M1112 = OFF, the unit is 100 µs (for SE: 1 ms). (Not available for EC5)

M1113: For PWM instruction, when M1113 = ON, the unit for Y2 output pulse to switch is 10 µs, when M1113 = OFF, the unit is 100 µs.

M1530: For PWM instruction, when M1530 = ON, the unit for Y4 output pulse to switch is 10 µs, when M1530 = OFF, the unit is 100 µs. (Applicable to EC5)

M1531 For PWM instruction, when M1530 = ON, the unit for Y4 output pulse to switch is 10 µs, when M1530 = OFF, the unit is 100 µs. (Applicable to EC5)

M1116: For PWM instruction, when M1116 = ON, the unit for Y0 and Y2 output pulse is 1 µs, and switching functions of M1112 and M1113 are not available. (Not available for EC5)

M1117: For PWM instruction, when M1117 = ON, the unit for Y1 and Y3 output pulse is 10 µs, and switching functions of M1070 and M1071 are not available. (Not available for EC5)

2. Special D registers description:

D1030	PV of Y0 pulse output (Low word)
D1031	PV of Y0 pulse output (High word)
D1032:	Low word of the present value of Y1 pulse output
D1033	High word of the present value of Y1 pulse output
D1336	PV of Y2 pulse output (Low word)
D1337	PV of Y2 pulse output (High word)
D1338:	Low word of the present value of Y3 pulse output.

D1339:	High word of the present value of Y3 pulse output
D1375:	Low word of the present value of Y4 pulse output.
D1376:	High word of the present value of Y4 pulse output
D1377:	Low word of the present value of Y6 pulse output.
D1378:	High word of the present value of Y6 pulse output

API		Mnemonic	Operands	Function	Controllers			
59	D	PLSR	S1 S2 S3 D	Pulse Ramp	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*	*	PLSR: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DPLSR: 17 steps
S_3					*	*	*	*	*	*	*	*	*	*	*	
D		*														

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Maximum frequency (Hz) **S_2**: Number of pulses **S_3**: Ramp up/down time (ms)

D: Pulse output device (Y0, Y1, Y2, Y3, Y4 and Y6 are available) (EC5/DVP-12SE: FW V1.xx does not support Y1 and Y3.)

Explanations:

(For ES2 / EX2 / SS2 / SA2 / SE / SX2 Series PLC CPU)

1. PLSR instruction performs a frequency ramp up/down process when positioning. Speed ramp up process is activated between static status to the target speed. Pulse output persists in target speed before getting close to target position. When target position is near, speed ramp down process executes, and pulse output stops when target position is achieved.
2. Set range of **S_1** pulse output frequency:

Range of **S_1** pulse output frequency:			
Output frequency:	Output	Y0, Y2	Y1, Y3
	16-bit	SS2: 6~10,000Hz ES2/EX2/SA2/SX2/SE: 6~32,767Hz	6~10,000Hz
	32-bit	SS2: 6~10,000Hz ES2/EX2/SA2/SX2/SE: 0~100,000Hz	6~10,000Hz
If frequency smaller than 6Hz is specified, PLC will output 6Hz. If frequency bigger than max frequency is specified, PLC will output with max frequency.			

3. When output device is specified with Y0, Y2, the start/end frequency of Y0 is set by D1340 and start/end frequency of Y2 is set by D1352.
4. When output device is specified with Y1, Y3, the start/end frequency is 0Hz.
5. When D1220/D1221 = K1 or K2, positive/negative sign of S2 denotes pulse output direction.
6. PLSR instruction supports two modes of pulse output as below list.

Mode / Output	D1220			D1221		
	K0		K1	K0		K1
Y0	Pulse		Pulse			
Y1		Pulse	Dir			
Y2				Pulse		Pulse
Y3					Pulse	Dir

PLSR instruction does NOT support the value of pulse=0, e.g. infinite pulse output.

7. When assigning Y0 and Y2 output mode as Pulse, i.e. D1220 = K0, D1221 = K0, the available range for **S2** is 1~32,767 (16-bit instruction) and 1~2,147,483,647 (32-bit instruction).

8. When assigning Y0 and Y2 output mode as Pulse/Dir, i.e. D1220 = K1, D1221 = K1, the available range for **S2** is 1~32,767 or -1~-32,768 (16-bit instruction) and 1~2,147,483,647 or -1~-2,147,483,648 (32-bit instruction)

9. When assigning output device as Y1 and Y3, the available range for **S2** is 1~32,767 (16-bit instruction) and 1~2,147,483,647 (32-bit instruction).

10. **S3**: Ramp up/down time (unit: ms, min. 20ms).

When assigning output device as Y1 and Y3, the set value of ramp up and ramp down time should be the same.

When assigning output device as Y0 and Y2, and if:

- M1534 = OFF (Y0) and M1535 = OFF (Y2), the ramp up and ramp down time should be the same.
- M1534 = ON and M1535 = ON, then **S3** specifies ramp up time only. The ramp down time is specified by value set in D1348 (Y0) and D1349 (Y2).

11. When M1257 = OFF, ramp up/down curve of Y0 and Y2 is straight line. When M1257 = ON, ramp up/down curve will be S curve. The ramp up/down curve of Y1 and Y3 is fixed as straight line

12. The output will not be affected if **S1**, **S2** or **S3** are changed when PLSR instruction is being executed. PLSR instruction has to be stopped if changing values in **S1**, **S2** or **S3** is required.

13. Flags for indicating pulse output status:

Output	Y0	Y1	Y2	Y3
Completion	M1029	M1030	M1102	M1103
Immediately Pause	M1078	M1079	M1104	M1105

a) When pulse output on Y0/Y1 specified as Pulse/Dir (D1220 = K1) is completed, completion flag M1029 = ON.

b) When pulse output on Y2/Y3 specified as Pulse/Dir (D1221 = K1) is completed, completion flag M1102 = On。

c) When PLSR/DPLSR instruction is activated again, the completion flags will automatically be reset.

14. During the ramp up process, the pulse numbers (frequency x time) of each speed shift may not all be integer values, but PLC will operate integer value only. In this case, the omitted decimals will result in errors between each speed shift, i.e. pulse number for each shift may differ due to this operation. For ensuring the required output pulse number, PLC will fill in pulses as need automatically in order to correct the deviation.

15. There is no limitation on the times of using this instruction in the program. However, only 4 instructions can be executed at the same scan time. When several pulse output instructions (PLSY, PWM, PLSR) use Y1 as the output device in the same scan cycle, PLC will execute pulse output according to the driven order of these instructions.

16. Set value falls out of the available range of operands will be automatically corrected with the min. or max available value.
17. When M1334 or M1335 is enabled, execute API59 PLSR/DPLSR instructions on Y0 or Y2 to ramp-down when the conditional contacts are closed. When the conditional contacts are closed, the deceleration will stop and the flags M1334/M1335 will be cleared. After the conditional contacts are closed, if you need to use the flags M1334/M1335 to stop the deceleration, you need to enable the flags M1334/M1335 again.

Series	**ES2/ EX2**	**ES2-C**	**ES2-E**	**12SA2/ SX2**	**SS2**	**12SE**	**26SE**	**28SA2**
Firmware version	V3.42	V3.48	V1.00	V2.86	V3.28	V2.02	V1.0	V2.90

(For EC5 Series PLC CPU)

1. PLSR instruction performs a frequency ramp up/down process when positioning. Speed ramp up process is activated between static status to the target speed. Pulse output persists in target speed before getting close to target position. When target position is near, speed ramp down process executes, and pulse output stops when target position is achieved.
2. Set range of **S_1** pulse output frequency:

Output frequency range of EC5 Series PLC CPU		
range	Output	Y0, Y2, Y4, Y6
	16-bit instruction	6~32,767 Hz
	32-bit instruction	6~50,000 Hz
If frequency equals or smaller than 6Hz is specified, pulse output will be disabled. If frequency bigger than max frequency is specified, PLC will output with max frequency.		

3. When output device is specified with Y0, Y2, Y4, Y6, the start/end frequency of Y0 is set by D1340, of Y2 by D1352, of Y4 by D1379, and of Y6 by D1380.
4. When D1220/D1221/D1341/D1342= K0, K1,or K2, the positive/negative sign of **S_2** denotes pulse output direction (Positive/negative).
5. Four pulse output modes:

Mode / Output	D1220		D1221		D1341		D1342	
	K0	K1	K0	K1	K0	K1	K0	K1
Y0	Pulse	Pulse						
Y1		Dir						
Y2			Pulse	Pulse				
Y3				Dir				
Y4					Pulse	Pulse		
Y5						Dir		
Y6							Pulse	Pulse
Y7								Dir

PLSR instruction does NOT support the value of pulse=0, e.g. infinite pulse output.

6. When assigning Y0, Y2, Y4, Y6 output mode as Pulse, i.e. D1220 = K0, D1221 = K0, D1341 = K0, D1342 = K0), the available range for **S₂** is 1~32,767 (16-bit instruction) and 1~2,147,483,647 (32-bit instruction).
7. When assigning Y0 and Y2 output mode as Pulse/Dir, i.e. D1220 = K1, D1221 = K1, D1341 = K0, D1342 = K0, the available range for **S_2** is 1~32,767 or -1~-32,768 (16-bit instruction) and 1~2,147,483,647 or -1~-2,147,483,648 (32-bit instruction)
8. **S_3**: Ramp up/down time (unit: ms; minimum: 20ms).
   When assigning output device as Y0, Y2, Y4 and Y6 and if:
   - M1534 = OFF (Y0), M1535 = OFF (Y2), M1536 = OFF (Y4), M1537 = OFF (Y6), the ramp up and ramp down time should be the same.
   - M1534 = ON, M1535 = ON, M1536 = ON, M1537 = ON, then **S_3** specifies ramp up time only. The ramp down time is specified by value set in D1348 (Y0), D1349 (Y2), D1350 (Y4), and D1351 (Y6).
9. When M1257 = OFF, ramp up/down curve of Y0 and Y2 is straight line. When M1257 = ON, ramp up/down curve will be S curve. The ramp up/down curve of Y1 and Y3 is fixed as straight line
10. The output will not be affected if **S_1**, **S_2** or **S_3** are changed when PLSR instruction is being executed. PLSR instruction has to be stopped if changing values in **S_1**, **S_2** or **S_3** is required.
11. Flags for indicating pulse output status:

Output device	Y0	Y2	Y4	Y6
Completed Flag	M1029	M1102	M1321	M1322
Immediately pause	M1078	M1104	M1310	M1311

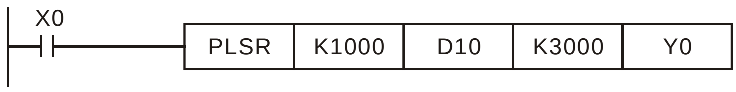
   a) M1029 = ON after Y0/Y1 (D1220=K1, pulse/Dir) output is completed.
      M1102 = ON after Y2/Y3 (D1221=K1, pulse/Dir) output is completed.
      M1321 = ON after the Y4/Y5 (D1341 = K1, pulse/Dir) output is completed.
      M1322 = ON after the Y6/Y7 (D1342 = K1, pulse/Dir) output is completed.
   b) When PLSY / DPLSY instruction is OFF, the pulse output completed flags will all be reset.
12. During the ramp up process, the pulse numbers (frequency x time) of each speed shift may not all be integer values, but PLC will operate integer value only. In this case, the omitted decimals will result in errors between each speed shift, i.e. pulse number for each shift may differ due to this operation. For ensuring the required output pulse number, PLC will fill in pulses as need automatically in order to correct the deviation.
13. There is no limitation on the times of using this instruction in the program. However, only 4 instructions can be executed at the same scan time. When several pulse output instructions (PLSY, PWM, PLSR) use Y1 as the output device in the same scan cycle, PLC will execute pulse output according to the driven order of these instructions.

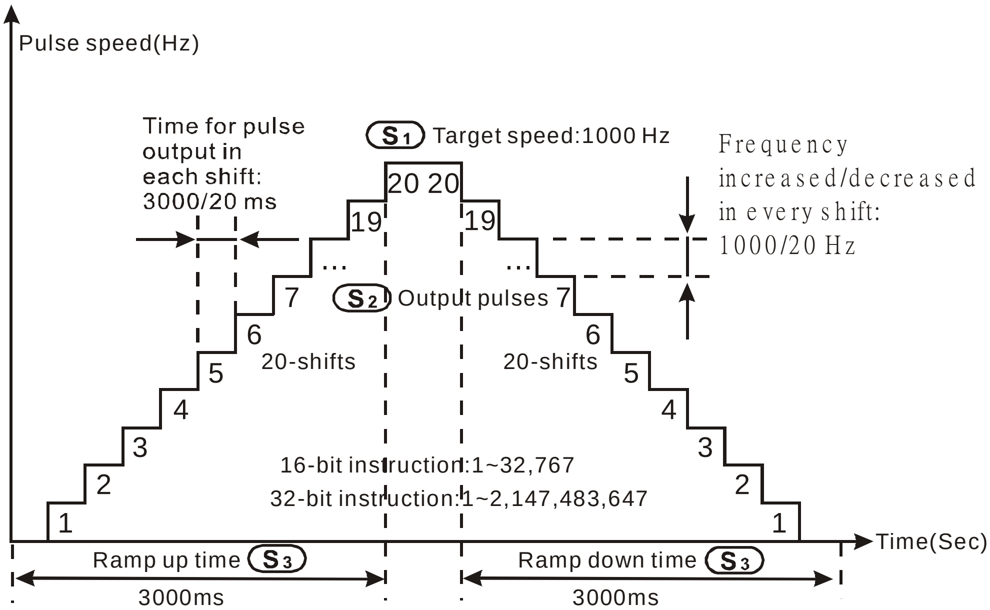
14. If the set value falls out of the available range of operands, it will be automatically corrected with the minimum or maximum available value.
15. When the flags, M1334, M1335, M1520, or M1521 is enabled, execute API59 PLSR/DPLSR instructions on Y0, Y2, Y4, or Y6 to ramp-down when the conditional contacts are closed. When the conditional contacts are closed, the deceleration will stop and the flags M1334, M1335, M1520, or M1521 will be cleared. After the conditional contacts are closed, if you need to use the flags M1334, M1335, M1520 or M1521 to stop the deceleration, you need to enable the flags M1334, M1335, M1520 or M1521 again.

Output device	Y0	Y2	Y4	Y6
Ramp-down when the conditional contacts are closed	M1334	M1335	M1520	M1521

Program Example:

1. When X0 = ON, PLSR performs pulse output on Y0 with a target speed of 1000Hz, output pulse number D10 and ramp up/down time of 3000ms. Ramp up process begins to increase 1000/20 Hz in every shift and every shift outputs D10/40 pulses for 3000/20 ms.
2. When X0 = OFF, the output stops immediately and starts from the count value in D1030, D1031 when PLSR is executed again.
3. Ramp up/down shifts for Y0, Y2: 20. Ramp up/down shifts for Y1, Y3: 10





Explanations on associated flags and registers:

1. Description on associated flags:

   For M1029, M1030, M1102, M1103, M1078, M1079, M1104, M1105, M1538, M1539, M1540, M1541, M1347, M1348, M1524, M1525, please refer to PLSY instruction.

   M1108: Y0 pulse output pause (ramp down). ON = pause, OFF = resume

   M1109: Y1 pulse output pause (ramp down). ON = pause, OFF = resume (Not available for EC5)

   M1110: Y2 pulse output pause (ramp down). ON = pause, OFF = resume

   M1111: Y3 pulse output pause (ramp down). ON = pause, OFF = resume (Not available for EC5)

   M1133: Y4 pulse output pause (ramp down). ON = pause, OFF = resume (Applicable to EC5)

   M1134: Y6 pulse output pause (ramp down). ON = pause, OFF = resume (Applicable to EC5)

   M1156: Enabling the mask and alignment mark function on I400/I401(X4) corresponding to Y0.

   M1257: Set the ramp up/down of Y0, Y2 to be “S curve.” ON = S curve.

   M1158: Enabling the mask and alignment mark function on I600/I601(X6) corresponding to Y2.

   M1534: Enable ramp-down time setting on Y0. Has to be used with D1348.

   M1535: Enable ramp-down time setting on Y2. Has to be used with D1349.

   M1536: Enable ramp-down time setting on Y4. Has to be used with D1350. (Applicable to EC5)

   M1537: Enable ramp-down time setting on Y6. Has to be used with D1351. (Applicable to EC5)

2. Description on associated special registers:

   For D1030~D1033, D1336~D1339, D1220, D1221, please refer to PLSY instruction

   D1026: M1156 = ON, D1026 stores pulse number for masking Y0 (Low word).

   D1027: M1156 = ON, D1026 stores pulse number for masking Y0 (High word).

   D1135: M1158 = ON, D1135 stores pulse number for masking Y2 (Low word).

   D1136: M1158 = ON, D1135 stores pulse number for masking Y2 (High word).

   D1232: Output pulse number for ramp-down stop when Y0 mark sensor receives signals. (Low word).

   D1233: Output pulse number for ramp-down stop when Y0 mark sensor receives signals. (High word).

   D1234: Output pulse number for ramp-down stop when Y2 mark sensor receives signals (Low word).

D1235: Output pulse number for ramp-down stop when Y2 mark sensor receives signals (High word).

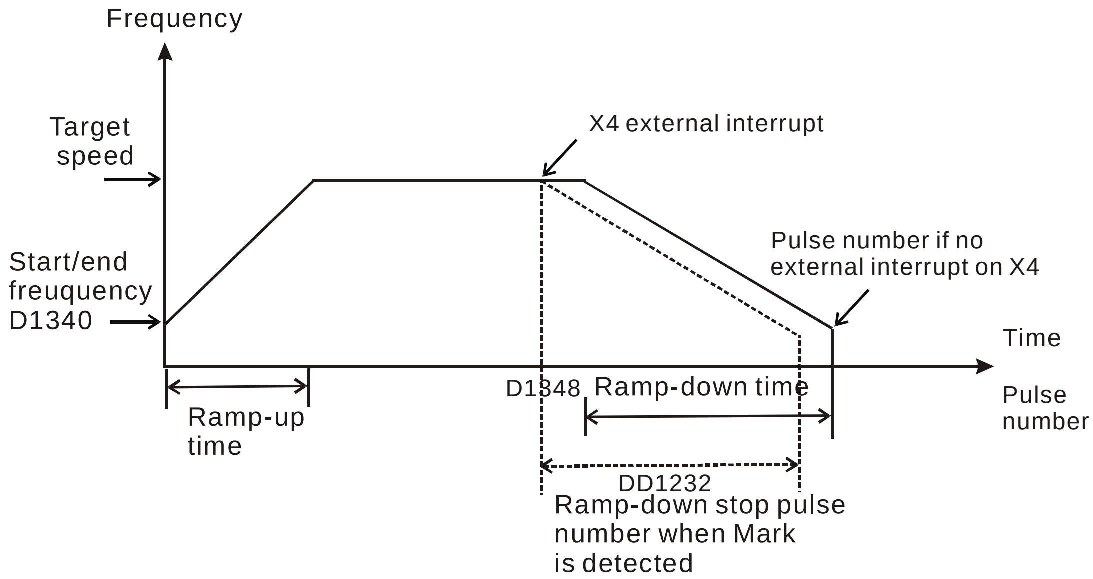
D1348: M1534 = ON, D1348 stores the ramp-down time of CH0(Y0, Y1) pulse output.

D1349: M1535 = ON, D1349 stores the ramp-down time of CH1(Y2, Y3) pulse output.

D1340 Start/end frequency of the pulse output CH0 (Y0, Y1)

D1352 Start/end frequency of the pulse output CH1 (Y2, Y3)

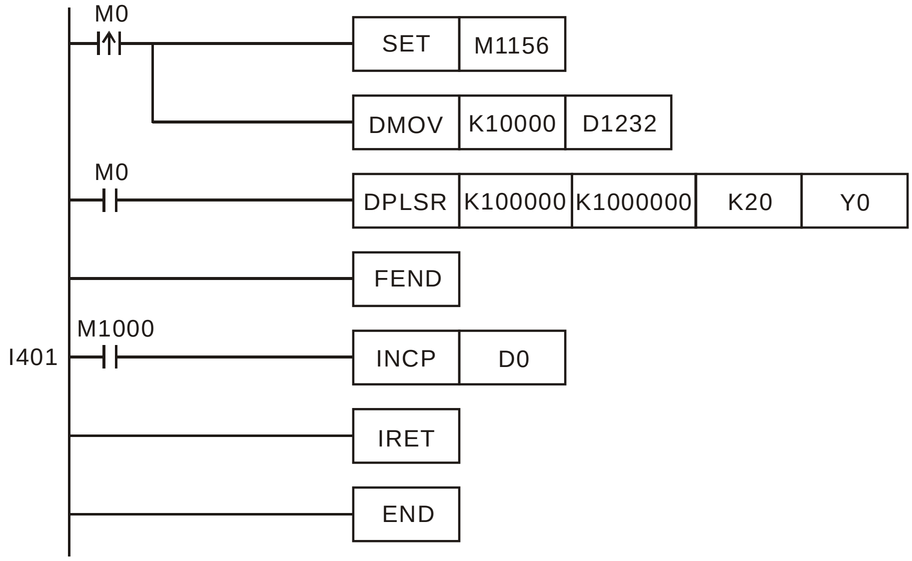
3. Operation of Mark function on Y0:



- When M1156/M1158 = ON, enable ramp-down pause (Mark function) on Y0/Y2 when X4/X6 receives interrupt signals.
- When Mark function is enabled, ramp down time is independent of the ramp up time. Users can set ramp up time in **S**3 and ramp down time in D1348/D1349. (Range: 20ms~32767ms)
- When Mark function is executed and the ramp-down stop pulses (DD1232/DD1234) are specified, PLC will execute ramp-down stop with specified pulses after Mark is detected. However, if DD1232/DD1234 are less than the specified ramp-down time (D1348 / D1349), PLC will fill DD1232/DD1234 with the value of ramp-down time. In addition, if DD1232/DD1234 is more than the half of total output pulses, PLC will modify DD1232/DD1234 to be less than half of the total output pulses.
- Ramp-down stop pulses (DD1232/DD1234) are 32-bit value. Set value K0 will disable the Mark function.
- Y0,Y2 relative parameters for Mask and Alignment Mark function:

Parameter / Output	Mark flag	Input points	Ramp down time	Pulse number for masking output	Pulse number for ramp-down of Mark function	Output pause (ramp down)	Pause status
Y0	M1156	X4	D1348	D1026, D1027	D1232, D1233	M1108	M1538
Y2	M1158	X6	D1349	D1135, D1136	D1234, D1235	M1110	M1540

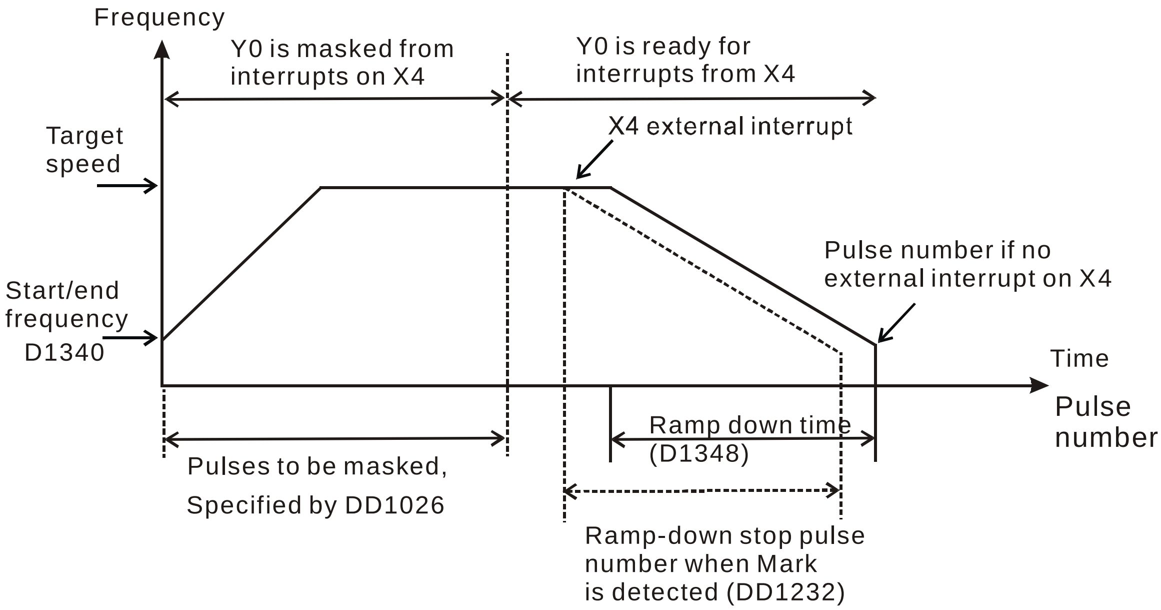
Program example 1:



Explanations:

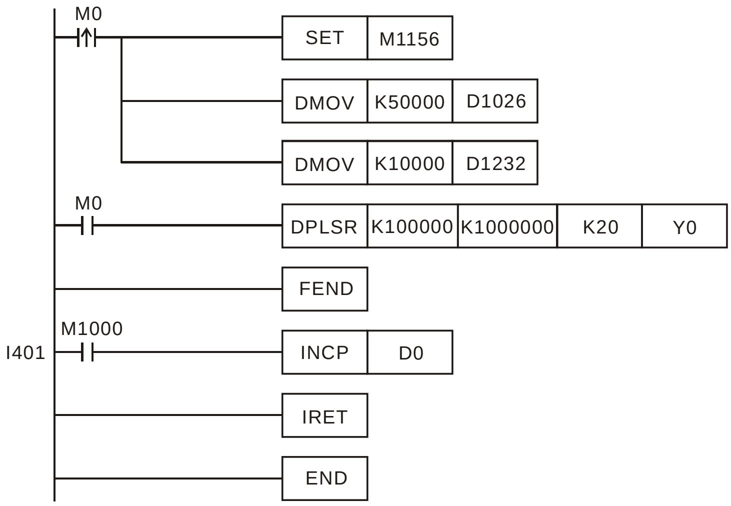
- When M0 is triggered, Y0 executes pulse output. If external interrupt is detected on X4, pulse output will perform ramp down process for 10,000 pulses and then stop. M1108 will be ON to indicate the pause status (ramp down). If no interrupt is detected, Y0 pulse output will stop after 1,000,000 pulses are completed.
- When pulse output ramps down and stops after Mark is detected, M1538 will be ON to indicate the pause status. If users need to complete the remaining pulses, set OFF the flag M1108 and pulse output will resume.

4. Operation of Mask function on Y0:



- Mask function on Y0 will be enabled when D1026 and D1027 are specified with values other than 0. Mask function is disabled when D1026 and D1027 are specified with 0. If pulse output process can not reach the target speed, PLC will clear DD1026 to disable the Mask function. If the Mask range is set to be within the ramp-up section, PLC will automatically modify DD1026 to be longer than the ramp-up section. On the other hand, if DD1026 is set between ramp- down section, PLC will modify DD1026 to be the range before the beginning of ramp-down process. Mask function setting method on Y2 is the same as Y0.

Program example 2:



Explanations:

- When M0 is triggered, Y0 executes pulse output. When external interrupt is detected on X4 after 50,000 pulses, pulse output will perform ramp down process for 10,000 pulses and then stop. M1108 will be ON. If no interrupt is detected on X4, Y0 pulse output will stop after 1,000,000 pulses are completed.
- Interrupt triggered between 0 ~ 50,000 pulses will be invalid, i.e. no ramp-down process will be performed before 50,000 pulses are achieved.

Points to note:

- When Mark function is executed with Mask function, PLC will check the validity of Mask range first, then ramp-down stop pulses of Mark function. If the above set values exceed the proper range, PLC will automatically modify the set values after the instruction is executed.
- When PLSR or positioning instructions with ramp-up/down section are enabled, the user can check the pulses of ramp-up section in DD1127 and pulses of ramp-down section in DD1133.
- Users can perform single speed positioning when ramp-up/down time setting is not specified.

5. Adding mask and alignment mark function for CH0 and CH1

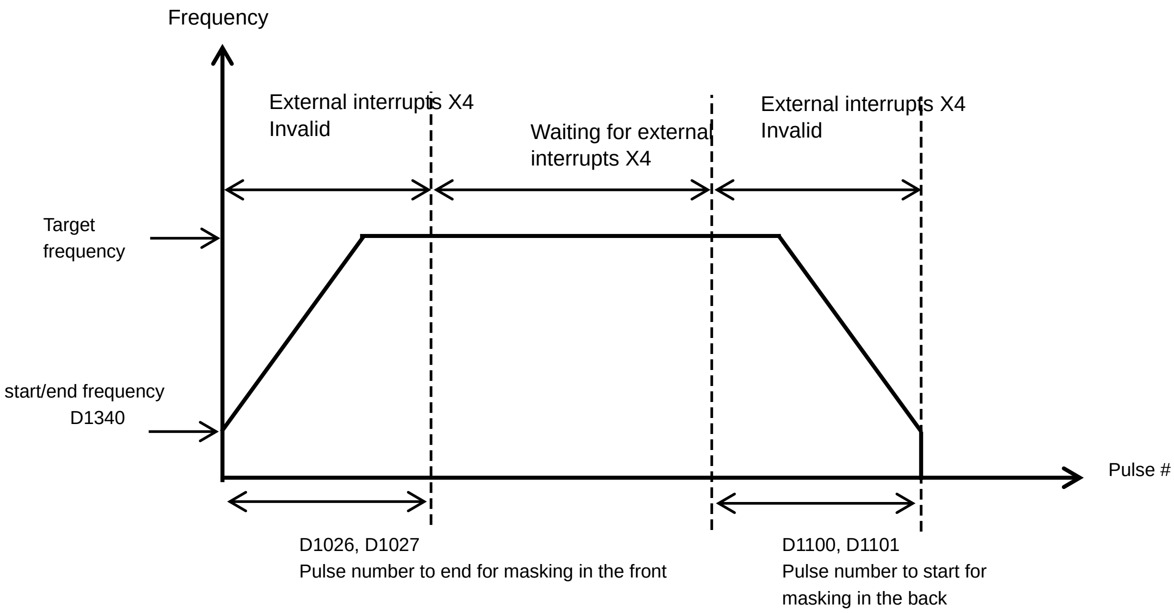
- Available for the followings

Series	ES2/ EX2	ES2-C	ES2-E	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware version	V3.28	V3.28	V1.00	V2.82	V3.28	V2.02	V2.0	V2.9

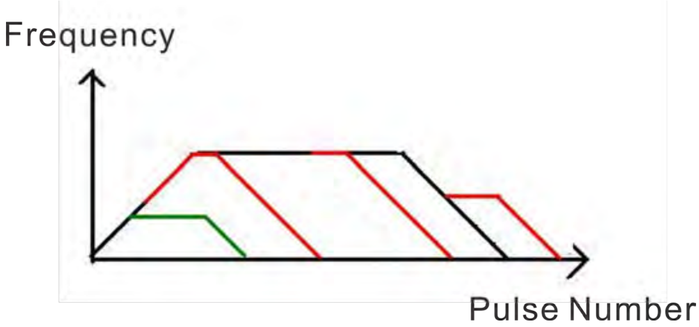
- CH0 and CH1 relative parameters for Mask and Alignment Mark function:

Output number	Marking deceleration flag	External input point	Ramp-up time	Ramp-down time	Starting/ Stopping frequency	Number of ramp-down pulses after marking	Front masking	Back masking
CH0 (Y0/Y1)	M1156	X4	D1343	D1348	D1340	D1232/D1233	D1026 D1027	D1100 D1101
CH1 (Y2/Y3)	M1158	X6	D1353	D1349	D1352	D1234/D1235	D1135 D1136	D1102 D1103

- Execution of the mask function (use Y0 as an example)



- Alignment mark function can be done in the sections of ramp-up, rump-down and speed.



6. Adding fixed slope function for CH0 and CH1
   - Available for the followings

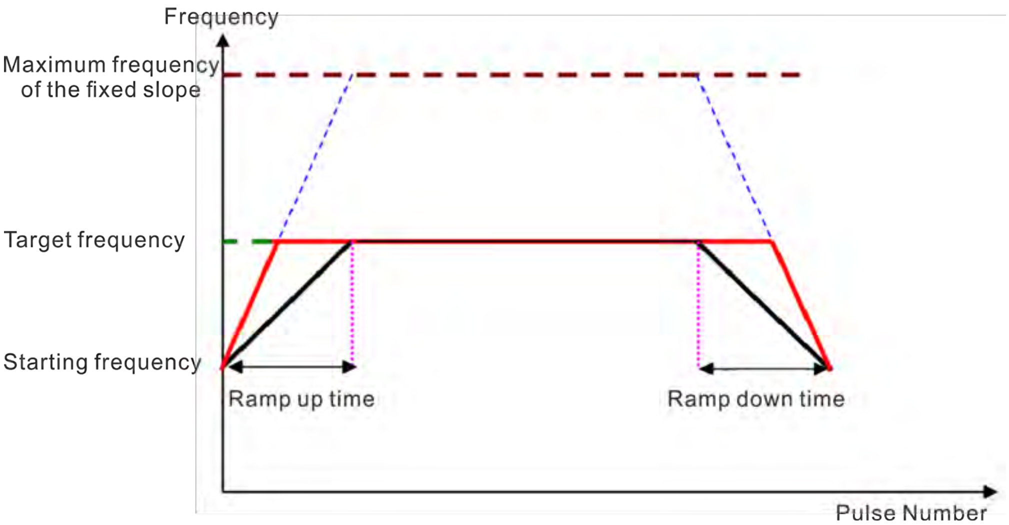
Series	ES2/EX2	ES2-C	ES2-E	12SA2/SX2	SS2	12SE	26SE	28SA2
Firmware version	V3.28	V3.28	V1.00	V2.82	V3.24	V2.02	V2.0	V2.9

3

- Y0 and Y2 relative parameters for fixed slope function:

Output	Flag for fixed slope	Special device for the maximum frequency
Y0	M1604	D1410, D1411
Y2	M1605	D1412, D1413

- The frequency for the normal slope is defined by the frequencies of starting, ending and the target as well as the time of ramp-up and down. See the black line for reference. The frequency for the fixed slope is defined by the frequencies of starting, ending and the maximum as well as the time of ramp-up and down. See the red line for reference.



7. Add new functions such as adding alignment marks to the ramping down, the frequency of the fixed slope and selected masking for the output points Y1 and Y3. The actions are the same as aforementioned 5 and 6. And the relative parameters are listed below.

- Available for the followings

Series	ES2/ EX2	ES2-C	ES2-E	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware version	V3.42	V3.48	V1.40	V2.86	--	V2.02	V1.2	V2.9

- **Special D/M Devices Corresponding to the Marking and Masking Function**

Output number	Marking deceleration flag	External input point	Ramp-up time	Ramp-down time	Starting/ Stopping frequency	Number of ramp-down pulses after marking	Front masking	Back masking
Y0	M1156	X4	D1343	D1348	D1340	D1232/D1233	D1026/ D1027	D1100/ D1101
Y1	M1157	X5	NA	NA	NA	D1236/D1237	D1154/ D1155	D1156/ D1157
Y2	M1158	X6	D1353	D1349	D1352	D1234/D1235	D1135/ D1136	D1102/ D1103
Y3	M1159	X7	NA	NA	NA	D1238/D1239	D1158/ D1159	D1160/ D1161

It does not support separating the ramp up and ramp down nor does it support setting up the start/stop frequency.

- Y1 and Y3 relative parameters for fixed slope function:

Output	Flag for fixed slope	Special device for the maximum frequency
Y1	M1606	D1988, D1989
Y3	M1607	D1990, D1991

- For EC5 Series:

Output	Flag for fixed slope	Special device for the maximum frequency
Y0	M1604	D1410, D1411
Y2	M1605	D1412, D1413
Y4	M1606	D1988, D1989
Y6	M1607	D1990, D1991

Note: If the values in the device where stores pulse number for masking in the front for Y0-Y3 is zero or less than -4 or equals to -4, it indicates the marking and masking functions in the front are disabled. On the other hand, if the values is greater than 0 or between -1 to -3, it indicates the marking and masking functions in the front are enabled. If the values in the device where stores pulse number for masking in the back for Y0-Y3 is less than 0 or equals to 0, it indicates the marking and masking functions in the back are disabled. On the other hand, if the values is greater than 0 or if the values in the device where stores pulse number for masking in the front is less than -3, it indicates the marking and masking functions in the back are enabled.

8. PLSR/DPLSR Instructions

- Added new marking behaviors A-C for PLSR/DPLSR instructions and behavior B (-3) for DCLLM instruction.

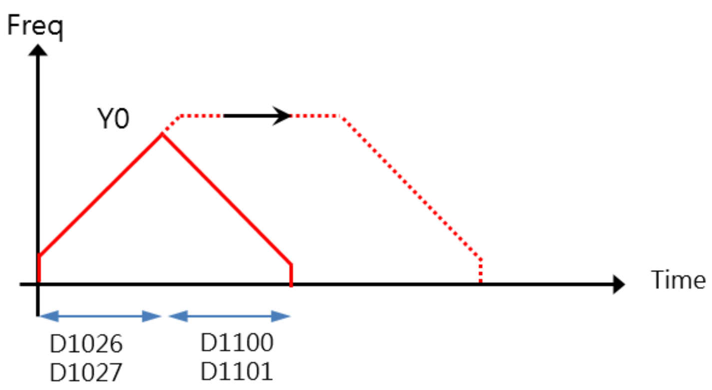
Applicable Models and Starting Versions

Series	ES2/EX2/ES2-C	ES2-E	12SA2/SX2	SS2	12SE	26SE	28SA2
Firmware	V3.60	V1.40	V3.00	--	V2.02	V2.02	V2.9

Descriptions of behaviors A-C

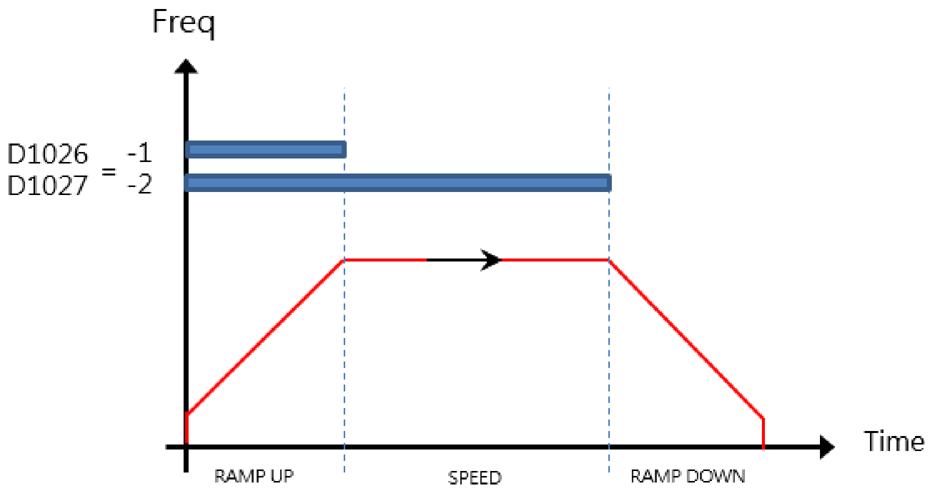
A. When the number of pulses is not sufficient to complete acceleration/deceleration, marking and masking are added in the area.

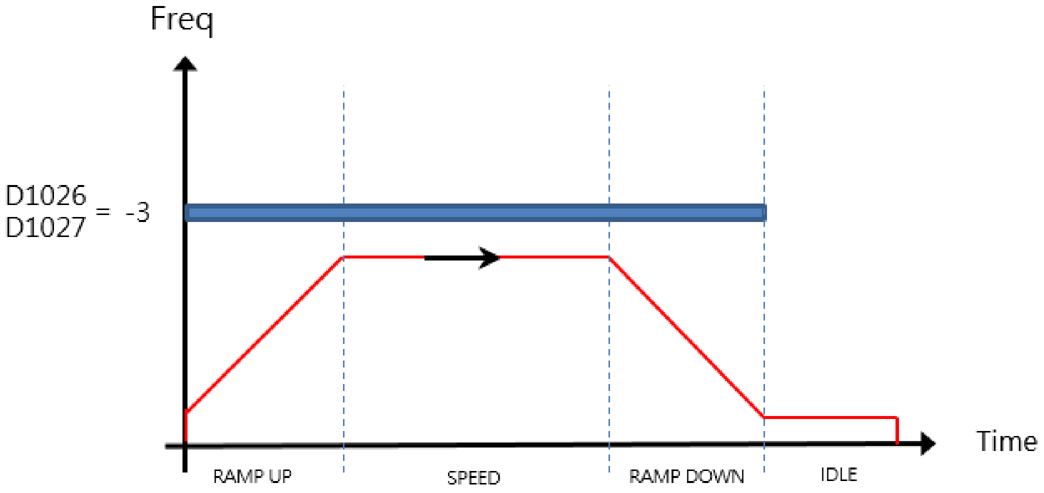
See the Y0 example below. The masking in D1026/1027 of the front masking area and D1100/1101 of the back masking area are effective.



B. When the masking number is -1 in the front masking area, it indicates masking occurs in the acceleration area; -2 in the front masking area means masking occurs in the areas of acceleration and full-speed; -3 (only available for DCLLM instruction) in the front masking area means masking occurs in the areas of acceleration, full-speed and deceleration.

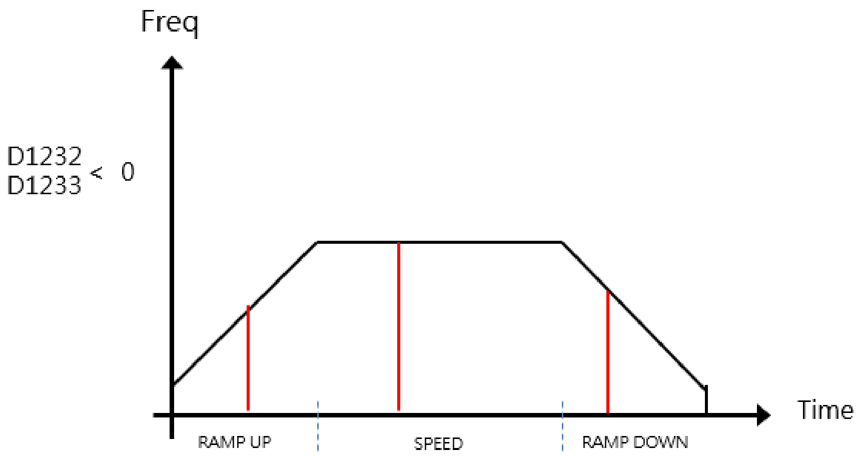
See the Y0 example below. The values of D1026/1027 in front masking area are set among -1 to -3. The masking can be done accordingly, you do not need to calculate the number of pulses in each area.





C. You can set number of deceleration pulses after marking to less than 0 (<0) and when marking is done, the output stopped immediately.

See the Y0 example below. If you set the number of deceleration pulses after marking to less than 0 in D1232/1233, the output stopped immediately after it received the signal, whether it's in the area of acceleration, full-speed or deceleration.



3.6.7 Handy Instructions

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
60	IST	-	-	Initial state	✓	✓	✓	✓	7	-
61	SER	DSER	✓	Search a data stack	✓	✓	✓	✓	9	17
62	ABSD	DABSD	-	Absolute drum sequencer	✓	✓	✓	✓	9	17
63	INCD	-	-	Incremental drum sequencer	✓	✓	✓	✓	9	-
64	TTMR	-	-	Teaching timer	✓	✓	✓	✓	5	-
65	STMR	-	-	Special timer	✓	✓	✓	✓	7	-
66	ALT	-	✓	Alternate state	✓	✓	✓	✓	3	-
67	RAMP	DRAMP	-	Ramp variable value	✓	✓	✓	✓	9	17
68	DTM	-	✓	Data transform and move	✓	✓	✓	✓	9	-
69	SORT	DSORT	-	Data sort	✓	✓	✓	✓	11	21
315	XCMP	-	-	Setting up to compare the inputs of multiple work stations	ES2 EX2	-	✓	✓	11	-
316	YOUT	-	-	Comparing the outputs of multiple work stations	ES2 EX2	-	✓	✓	9	-

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
60	IST	S D1 D2	Initial State				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	IST: 7 steps
S	*	*	*													
D_1				*												
D_2				*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device for assigning pre-defined operation modes (8 consecutive devices). **D_1** The smallest No. of step points in auto mode. **D_2**: The greatest No. of step points in auto mode.

Explanations:

1. The IST is a handy instruction specifically for the initial state of the step ladder operation modes.
2. The range of **D_1** and **D_2** : S20~S911, **D_1** < **D_2**.
3. IST instruction can only be used one time in a program.

Program Example 1:

```
M1000
──┤├──────[ IST | X20 | S20 | S60 ]
```

S:

X20: Individual operation (Manual operation)
X21: Zero return
X22: Step operation
X23: One cycle operation
X24: Continuous operation
X25: Zero return start switch
X26: Start switch
X27: Stop switch

1. When IST instruction is executed, the following special auxiliary relays will be assigned automatically.

   M1040: Movement inhibited
   M1041: Movement start
   M1042: Status pulse
   M1047: STL monitor enable
   S0: Manual operation/initial state step point
   S1: Zero point return/initial state step point
   S2: Auto operation/initial state step point

2. When IST instruction is used, S10~S19 are occupied for zero point return operation and cannot be used as a general step point. In addition, when S0~S9 are in use, S0 initiates “manual operation mode”, S1 initiates “zero return mode” and S2 initiates “auto mode”. Thus, the three step points of initial state have to be programmed in first priority.
3. When S1 (zero return mode) is initialized, i.e. selected, zero return will NOT be executed if any of the state S10~S19 is ON.
4. When S2 (auto mode) is initialized, i.e. selected, auto mode will NOT be executed if M1043 = ON or any of the state between **D_1** to **D_2** is ON.

Program Example 2:

Robot arm control (by IST instruction):

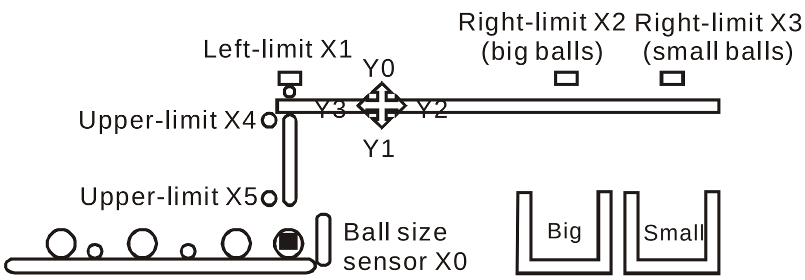
1. Control purpose:

   Select the big balls and small balls and move them to corresponding boxes. Configure the control panel for each operation.

2. Motion of the Robot arm:

   lower robot arm, clip balls, raise robot arm, shift to right, lower robot arm, release balls, raise robot arm, shift to left to finish the operation cycle.

3. I/O Devices



4. Operation mode:

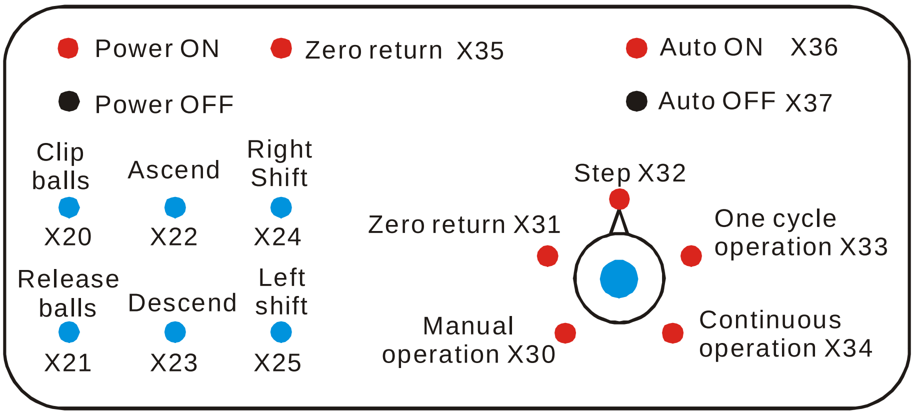
   Single step: Press single button for single step to control the ON/OFF of external load.

   Zero return: Press zero return button to perform homing on the machine.

   Auto (Single step / One cycle operation / Continuous operation):

   - Single step: the operation proceeds with one step every time when Auto ON is pressed.
   - One cycle operation: press Auto ON at zero position, the operation performs one full cycle operation and stops at zero point. If Auto OFF is pressed during the cycle, the operation will pause. If Auto ON is pressed again, the operation will resume the cycle and stop at zero point.
   - Continuous operation: press Auto ON at zero position, the operation will perform continuous operation cycles. If Auto OFF is pressed, the operation will stop at the end of the current cycle.

5. Control panel

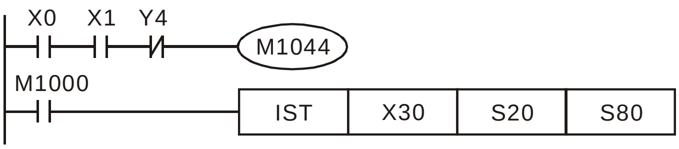


a) X0: ball size sensor.

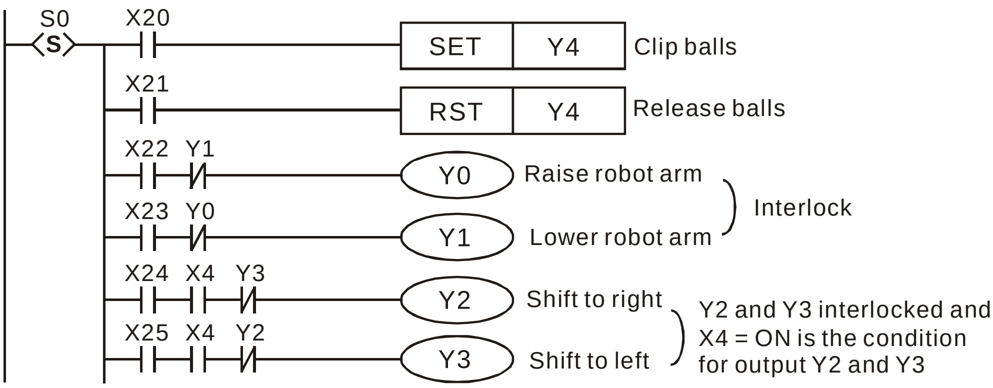
b) X1: left-limit of robot arm, X2: right-limit (big balls), X3: right-limit (small balls), X4: upper-limit of clamp, X5: lower-limit of clamp.

c) Y0: raise robot arm, Y1: lower robot arm, Y2: shift to right, Y3: shift to left, Y4: clip balls.

6. START circuit:

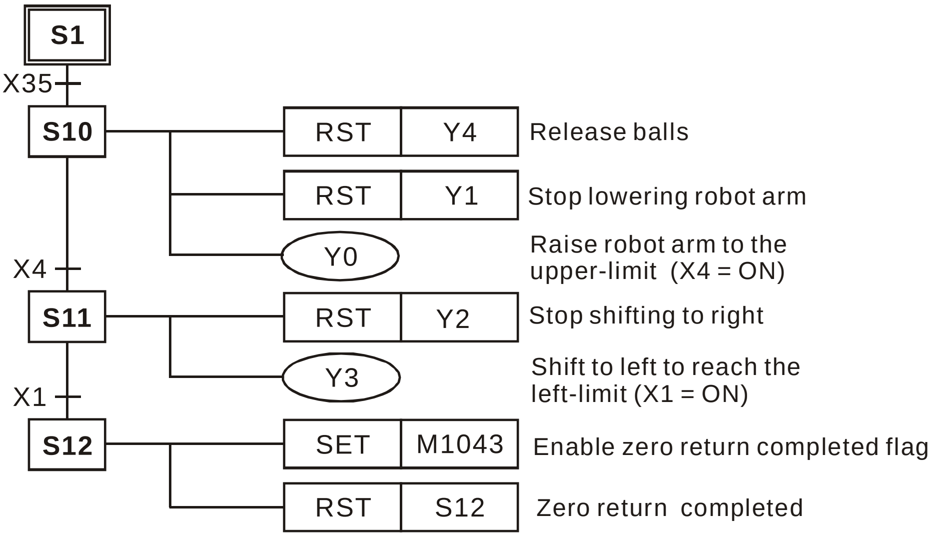


7. Manual mode:

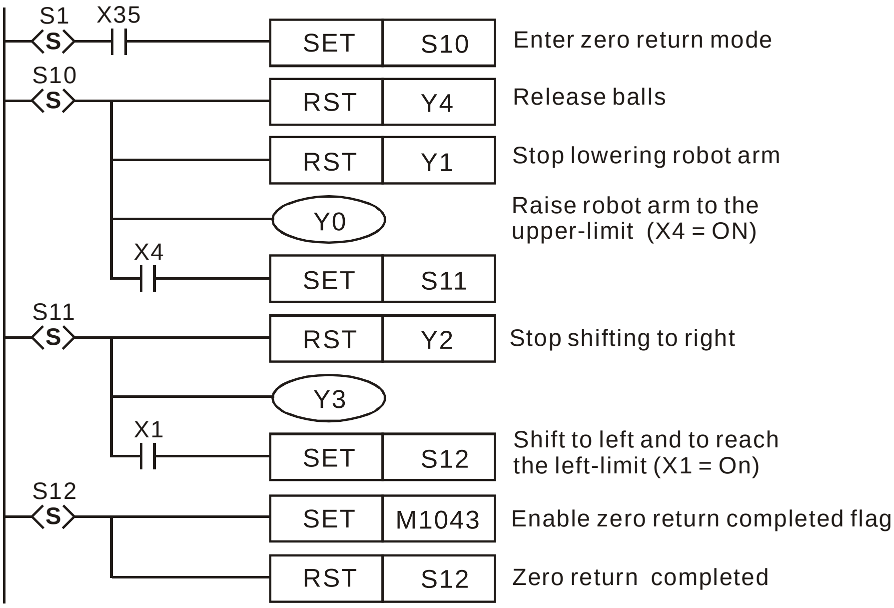


8. Zero return mode:

a) SFC:

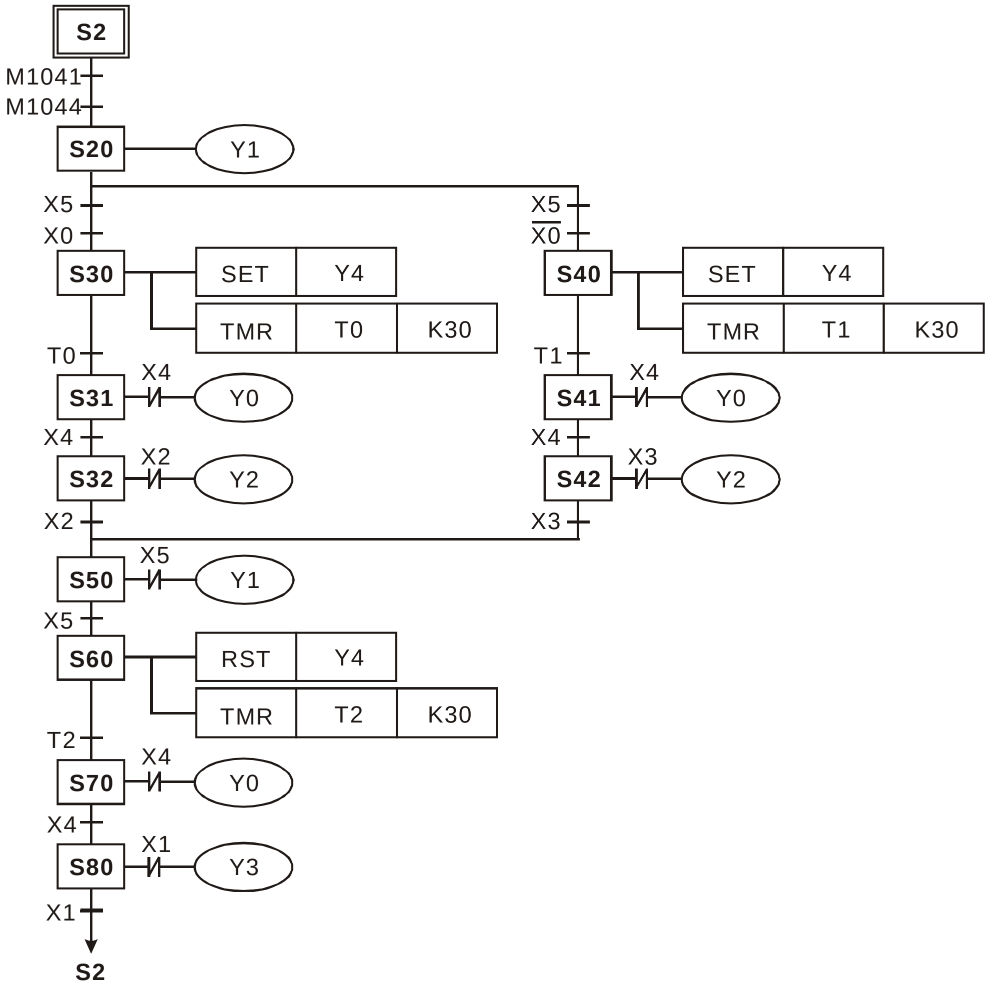


b) Ladder Diagram:

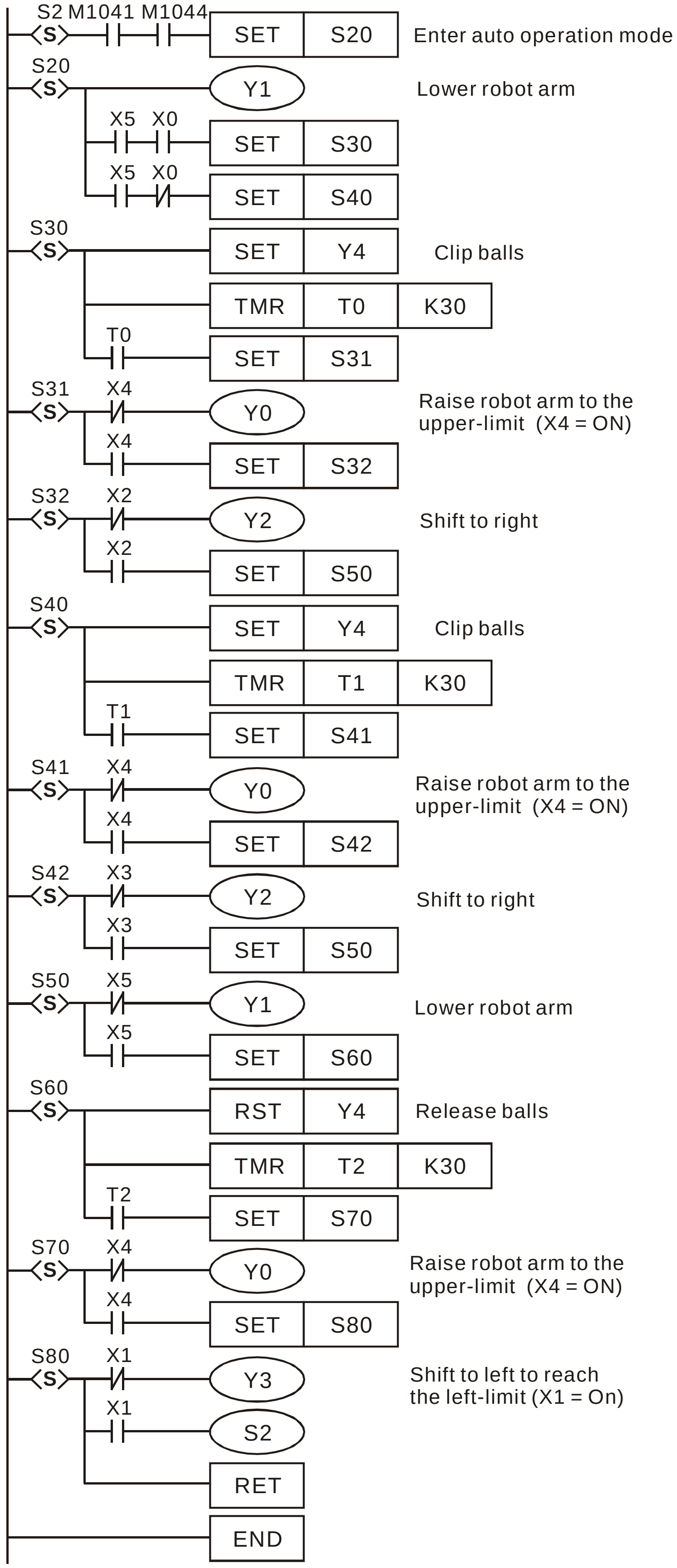


9. Auto operation (Single step / One-cycle operation / continuous operation):

 a) SFC:



b) Ladder Diagram:



Flag explanation:

M1040:

Disable step transition. When M1040 = ON, all motion of step points are disabled.

1. **Manual operation mode:** M1040 remains ON in manual mode.
2. **Zero return mode/one cycle operation mode:** M1040 remains ON in the interval after Auto Stop and before Auto Start is pressed.
3. **Step operation mode:** M1040 remians ON until Auto Start is pressed.
4. **Continuous operation mode:** When PLC goes from STOP→RUN, M1040 = ON. When Auto Start is pressed, M1040 turns OFF.

M1041:

Step transition starts. This special M indicates the transition from step point S2 to the next step point.

1. **Manual operation mode/Zero return mode:** M1041 remians OFF.
2. **Step operation mode/One cycle operation mode:** M1041 = ON when Auto Start is pressed.
3. **Continuous operation mode:** M1041 stays ON when Auto Start is pressed and turns OFF when Auto Stop is pressed.

3

M1042:

Enable pulse operation: When Auto Start is pressed, PLC sents out pulse once for operation. .

M1043:

Zero return completed: M1043 = ON indicates that zero return is completed.

M1044:

Zero point condition: In continuous operation mode, M1044 has to be ON as a condition for enabling step transition from S2 to the next step point.

M1045:

Disable "all output reset" function.

- If the machine (not at the zero point) goes,
  - from manual (S0) to zero return (S1)
  - from auto (S2) to manual (S0)
  - from auto (S2) to zero return (S1)

And

M1045 = OFF, any of the S among $\mathbf{D_1}$ ~ $\mathbf{D_2}$ in action will be reset as well as the output Y.

M1045 = ON, output Y will be retained but the step in action will be reset.

- If the machine (at the zero point) goes from zero return (S1) to manual (S0), no matter M1045 is ON or OFF, Y output will be retained but the step in action will be reset.

M1046:

Indicates STL(Step Ladder) status. When STL operation is activate, M1046 = ON if any of the step point S is ON. If M1047 = ON, M1046 also activates to indicate ON status of step points. In addition, D1040 ~ D1047 records 8 step numbers from the current ON step to the previous 7 ON steps.

M1047:

Enable STL monitoring. When IST instruction executes, M1047 will be forced ON, i.e. M1047 remains ON in every scan cycle as long as IST instruction is executing. This flag is used to monitor all step points (S).

D1040~D1047:

Records 8 step numbers from the current ON step to the previous 7 ON steps.

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
61	D	SER	P	(S₁) (S₂) (D) (n)	Search a Data Stack				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1							*	*	*	*	*	*	*			SER, SERP: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DSER, DSERP: 17 steps
D								*	*	*	*	*	*			
N					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Start device of data stack **S_2**: Device to be searched **D**: Start device for storing search result (occupies 5 consecutive devices) **n**: Stack length

Explanations:

1. SER instruction searches for the value stored in **S_2** from the data stack starting with **S_1**, with a stack length **n**. The search results are stored in the 5 registers starting from **D**
2. **D** stores the total of the matched results; **D+1** stores the No. of device storing the first matched result; **D+2** stores the No. of device storing the last matched result; **D+3** stores the No. of device storing the smallest value; **D+4** stores the No. of device storing the biggest value.
3. If operand **S_2** uses index F, only 16-bit instruction is available
4. If the instruction applied 32-bit instruction, operands **S_1**, **S_2**, **D**, **n** will specify 32-bit registers.
5. The range of operand **n**: **n** = 1~256 (16-bit instruction), **n** = 1~128 (32-bit instruction)

Program Example:

1. When X0 = ON, the data stack D10~D19 are compared with D0 and the result is stored in D50~D54. If there is no matched result, the content of D50~D52 will all be 0.
2. D53 and D54 store the location of the smallest and biggest value. When there are more than one smallest and biggest values, the devices with bigger No. will be recorded.

```
  X0
--| |------[ SER | D10 | D0 | D50 | K10 ]
```

S_1	Content	Data to be compared	Data No.	Result
D10	88		0	
D11	100	**S_2**	1	Equal
D12	110		2	
D13	150		3	
D14	100		4	Equal
D15	300	D0=K100	5	
D16	100		6	Equal
D17	5		7	Smallest
D18	100		8	Equal
D19	500		9	Largest

D	Content	Explanation
D50	4	The total data numbers of equal value
D51	1	The number of the first equal value
D52	8	The number of the last equal value
D53	7	The number of the smallest value
D54	9	The number of the largest value

API	Mnemonic		Operands	Function	Controllers			
62	D	ABSD	S1 S2 D n	Absolute Drum Sequencer	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1							*	*	*	*	*	*	*			ABSD: 9 steps DABSD: 17 steps
S_2											*	*	*			
D		*	*	*												
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

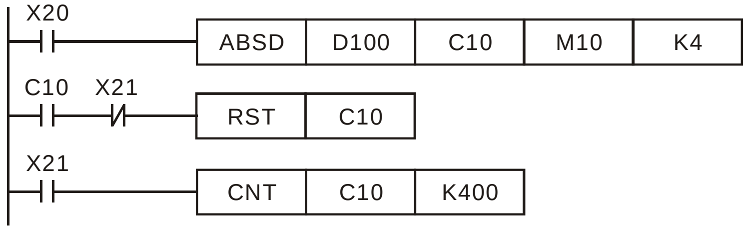
S₁: Start device of the data table **S₂**: No. of counter **D**: Start device for indicating comparison result **n**: Groups of data to be compared (**n**: 1~64)

Explanations:

1. ABSD instruction creates various output wave forms according to the current value of the counter designated by **S_2**. Usually, the instruction is applied for absolute cam control.
2. **S_2** of DABSD instruction can designate high speed counters. However, when the present value in the high speed counter is compared with the target value, the result cannot output immediately owing to the scan time. If an immediate output is required, please use DHSZ instruction that is exclusively for high speed counters.
3. When operand **S_1** uses KnX, KnY, KnM, KnS patterns, Kn should be K4 for 16-bit instruction and K8 for 32-bit instruction.

Program Example:

1. Before the execution of ABSD instruction, use MOV instruction to write all the set values into D100 ~ D107 in advance. The even-number D is for lower bound value and the odd-number D is for upper bound value.
2. When X10 = ON, the present value in counter C10 will be compared with the four groups of lower and upper bound values in D100 ~ D107. The comparison results will be stored in M10 ~ M13.
3. When X10 = OFF, the original ON/OFF status of M10 ~ M13 will be retained.

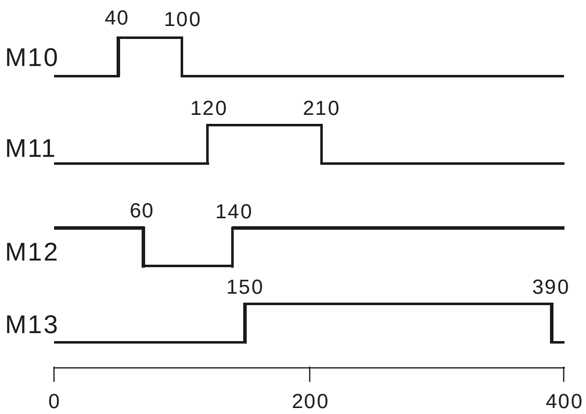


4. M10~ M13 = ON when the current value of C10 falls between lower and upper bounds.

Lower-bound value	Upper- bound value	Current value of C10	Output
D100= 40	D101 = 100	40≦C10≦100	M10 = ON
D102 = 120	D103 = 210	120≦C10≦210	M11 = ON
D104 = 140	D105 = 170	140≦C10≦170	M12 = ON
D106 = 150	D107 = 390	150≦C10≦390	M13 = ON

5. If the lower bound value is bigger than upper bound value, when C10<60 or C10 > 140, M12 = ON.

Lower- bound value	Upper- bound value	Current value of C10	Output
D100 = 40	D101 = 100	40≦C10≦100	M10 = ON
D102 = 120	D103 = 210	120≦C10≦210	M11 = ON
D104 = 140	D105 = 60	60≦C10≦140	M12 = OFF
D106 = 150	D107 = 390	150≦C10≦390	M13 = ON



API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
63	INCD	S1 S2 D n	Incremental drum sequencer				

Type OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1							*	*	*	*	*	*	*			INCD: 9 steps
S_2												*				
D		*	*	*												
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Start device of the data table **S_2**: No. of counter **D**: Start device for indicating comparison result **n**: Number of data to be compared (**n**: 1~64)

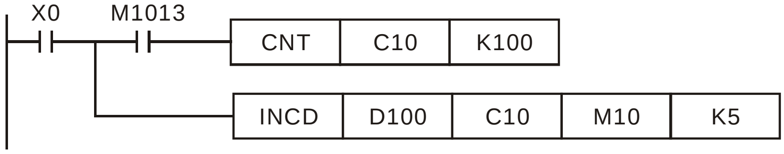
Explanations:

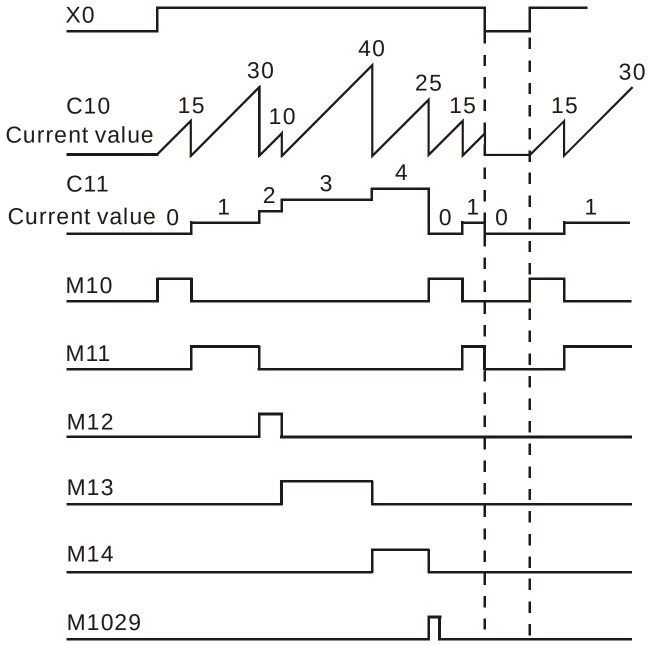
1. INCD instruction creates various output wave forms according to the current value of the counter designated by **S_2**. and **S_2+1**. Usually, the instruction is applied for relative cam control
2. The current value in **S_2** is compared with the set points specified by **S_1** (**n** consecutive devices) When value in **S_2** reaches the first set point, **S_2+1** counts once for indicating the number of present section, associated **D** turns ON, and **S_2** is reset then counts up from 0 again. When the drive contact of INCD instruction is OFF, the content in **S_2**. and **S_2+1** will be cleared.
3. When operand **S_1** uses KnX, KnY, KnM, KnS patterns, Kn should be K4 for 16-bit instruction.
4. Operand **S_2** should be C0~C198 and occupies 2 consecutive counters.
5. When the comparison of **n** data has been completed, the execution completed flag M1029 = ON for one scan cycle.

Program Example:

1. Before the execution of INCD instruction, use MOV instruction to write all the set values into D100 ~ D104 in advance. D100 = 15, D101 = 30, D102 = 10, D103 = 40, D104 = 25.
2. The current value of counter C10 is compared against the set-point value of D100~D104. Once the current value is equal to the set-point value, C10 will be reset and count up from 0 again. Meanwhile C11 counts once for indicating the number of present section
3. When the content of C11 increase 1, M10~M14 will be ON sequentially. Please refer to the following timing diagram.
4. When the comparison of 5 data has been completed, the execution completed flag M1029 = ON for one scan cycle and C11 is reset for next comparison cycle.

5. When X0 turns from ON →OFF, C10 and C11 will all be reset to 0 and M10~M14 = OFF. When X0 turns ON again, this instruction will be executed again from the beginning.





API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
64	TTMR	D n	Teaching Timer				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
D													*			TTMR: 5 steps
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

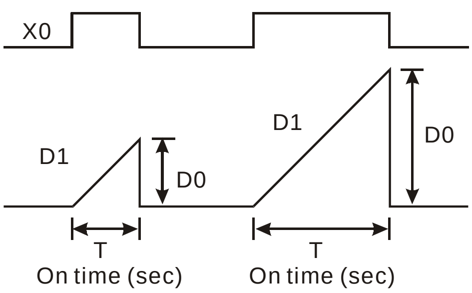
D: Device No. for storing the ON time of the input **n**: setting of multiple (**n**: K0~K2)

Explanations:

1. The ON time of the external button switch is measured and stored in **D + 1**(unit: 100ms). Value in **D + 1** is multiplied with a multiple specified by **n** and stored in **D** (unit: sec).
2. When **n** = K0, the value in **D + 1**(unit: 100ms) is multiplied with 1 and converted to **D** (unit: sec). When **n** = K1, the value in **D + 1**(unit: 100ms) is multiplied with 10 and converted to **D** (unit: sec). When **n** = K2, the value in **D + 1**(unit: 100ms) is multiplied with 100 and converted to **D** (unit: sec).
3. TTMR instruction can be used max 8 times in a program.

Program Example 1:

1. The duration that input X0 is pressed (ON duration of X0) will be stored in D1. The value in D1, multiplied by a multiple specified by **n**, is then moved to D0. In this case, the button switch can be used to adjust the set value of a timer.
2. When X0 = OFF, the content of D1 will be reset but the content of D0 remains.

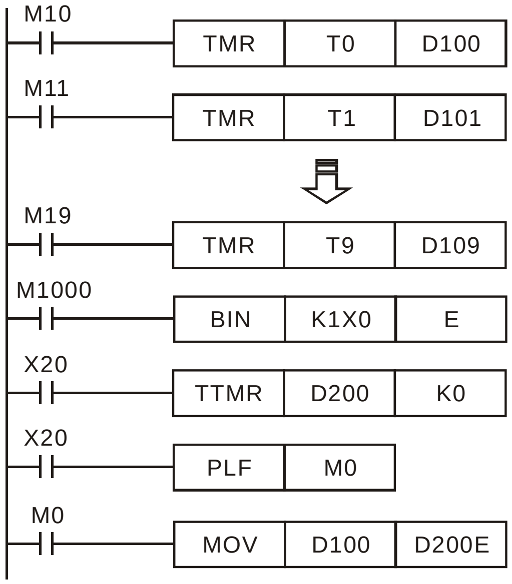


3. If ON duration of X0 is T sec, the relation between D0, D1 and **n** are shown as the table below.

n	D0 (unit: sec)	D1 (unit: 100 ms)
K0	T (sec) ×1	D1 = D0×10
K1	T (sec) ×10	D1 = D0
K2	T (sec) ×100	D1 = D0/10

Program Example 2:

1. Use TMR instruction to write in 10 groups of set time.
2. Write the set values into D100 ~ D109 in advance
3. The timer resolution is 0.1 sec for timers T0 ~ T9 and 1 sec for the teaching timer.
4. Connect the 1-bit DIP switch to X0 ~ X3 and use BIN instruction to convert the set value of the switch into a bin value and store it in E.
5. The ON duration (in sec) of X20 is stored in D200.
6. M0 is a pulse for one scan cycle generated when the teaching timer button X20 is released.
7. Use the set number of the DIP switch as the index pointer and send the content in D200 to D100E (D100 ~ D109).



Note:

The TTMR instruction can only be used 8 times in a program. If TTMR is used in a CALL subroutine or interrupt subroutine, it only can be use once.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
65	STMR	(S) (m) (D)	Special Timer				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S											*					STMR: 7 steps
m					*	*										
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: No. of timer (T0~T183) **m**: Set value in timer (**m** = 1~32,767, unit: 100ms)

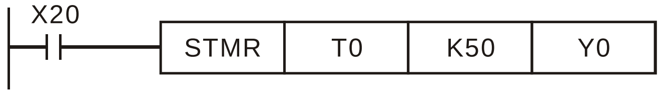
D: Start No. of output devices (occupies 4 consecutive devices)

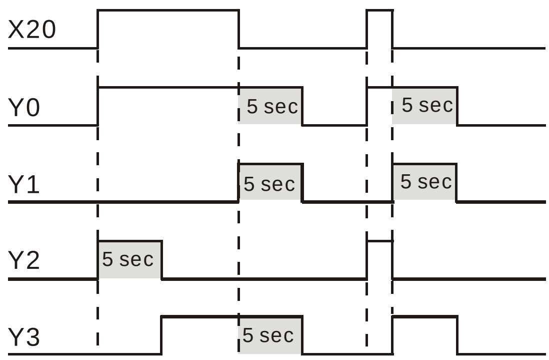
Explanations:

1. STMR instruction is specifically used for delay-OFF, ON/OFF triggered timer and flashing circuit.
2. The timer number (**S**) specified by STMR instruction can be used only once

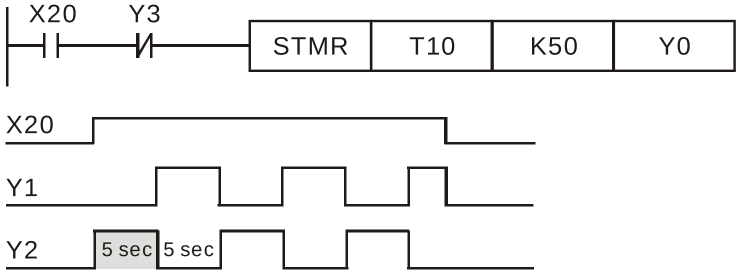
Program Example:

1. When X20 = ON, STMR sets T0 as the 5 sec special timer.
2. Y0 is the delay-OFF contact. When X20 is triggered, Y0 = ON; When X20 is OFF, Y0 = OFF after a 5 sec delay.
3. When X20 goes from ON to OFF, Y1 = ON for 5 seconds.
4. When X20 goes from OFF to ON, Y2 = ON for 5 seconds.
5. When X20 goes from OFF to ON, Y3 = ON after a 5 second delay. When X20 turns from ON to OFF, Y3 = OFF after a 5 second delay.





6. Apply a NC contact Y3 after the drive contact X20, and Y1, Y2 will form a flashing circuit output. When X20 turns OFF, Y0, Y1 and Y3 = OFF and the content of T10 will be reset.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
66		ALT	P	D	Alternate State				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ALT, ALTP: 3 steps
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

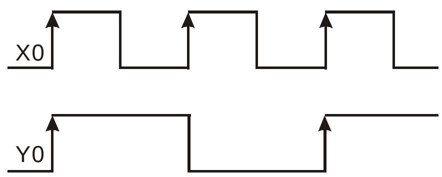
D: Destination device

Explanations:

1. The status of **D** is alternated every time when the ALT instruction is executed.
2. When ALT instruction is executed, ON/OFF state of **D** will be switched which is usually applied on switching two operation modes, e.g. Start/Stop
3. This instruction is generally used in pulse execution mode (ALTP).

Program Example 1:

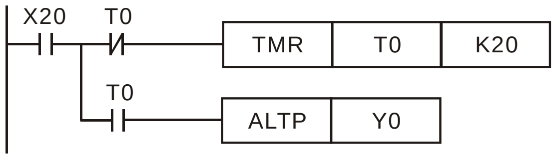
When X0 goes from OFF to ON, Y0 will be ON. When X0 goes from OFF to ON for the second time, Y0 will be OFF.



Program Example 2:

Creating a flashing circuit by applying ALTP with a timer

When X20 = ON, T0 will generate a pulse every two seconds and output Y0 will be switched between ON and OFF by the pulses from T0.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
67	D	RAMP	S_1 S_2 D n	Ramp variable Value				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	RAMP: 9 steps
S_1													*			DRAMP: 17 steps
S_2													*			
D													*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

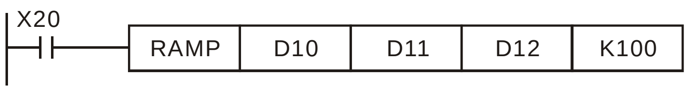
S_1: Start of ramp signal **S_2**: End of ramp signal **D**: Current value of ramp signal (occupies 2 consecutive devices) **n**: Times for scan (**n**: 1~32,767)

Explanations:

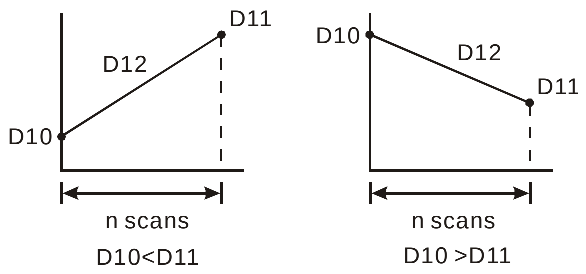
1. This instruction creates a ramp output. A ramp output linearity depends on a consistent scan time. Therefore, scan time has to be fixed before executing RAMP instruction.
2. When RAMP instruction is executed, the ramp signal will vary from **S_1** to **S_2**. Current value of ramp signal is stored in **D** and **D+1** stores the current number of accumulated scans. When ramp signal reaches **S_2**, or when the drive contact of RAMP instruction turns OFF, the content in **D** varies according to the setting of M1026 which is explained later in **Points to note**.
3. When **n** specifies a D register, the value in D cannot be modified during the execution of the instruction. Please modify the content of D when the instruction is stopped.
4. When this instruction is applied with analog output function, Ramp start and Ramp stop function can be achieved.

Program example:

1. Before executing the instruction, first drive M1039 = ON to fix the scan time. Use MOV instruction to write the fixed scan time to the special data register D1039. Assume the scan time is 30ms and take the below program for example, n = K100, the time for D10 to increase to D11 will be 3 seconds (30ms × 100).
2. When X20 goes OFF, the instruction will stop its execution. When X10 goes ON again, the content in D12 will be reset to 0 for recalculation
3. When M1026 = OFF, M1029 will be ON to indicate the completion of ramp process and the content in D12 will be reset to the set value in D10.
4. Set the Start and End of ramp signal in D10 and D11. When X20 = ON, D10 increases towards D11, the current value of the variation is stored in D12 and the number of current scans is stored in D13.



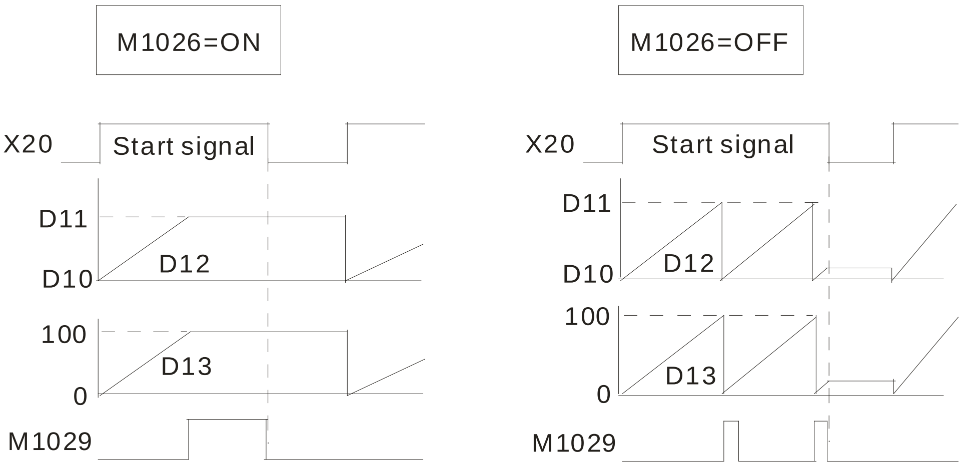
If X20 = ON,



The scan times is stored in D13

Points to note:

The variation of the content in D12 according to ON/OFF state of M1026 (Ramp mode selection):



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
68	DTM	P	S D m n	Data Transform and Move				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S													*			DTM: 9 steps
D													*			
m					*	*							*			
n					*	*							*			

(Bit Devices: X, Y, M, S; Word devices: K, H, KnX, KnY, KnM, KnS, T, C, D, E, F)

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Start device of the source data stack **D**: Start device of the destination data stack

m: Transformation mode **n**: Length of source data stack

Explanations:

1. For parameter settings of operand **m**, please refer to the following description. K, H, D devices can be specified by operand **m.** If the set value is not in the available range, no transformation or move operation will be executed and no error will be detected.
2. K, H, D devices can be specified by operand **n,** which indicates the length of the source data stack. The available range for **n** is 1~256. If the set value falls out of available range, PLC will take the max value (256) or the min value (1) as the set value automatically.
3. The parameter settings and series to support the m operand are listed below:

Parameters	Descriptions
K0	Transform 8-bit data into 16-bit data (Hi-byte, Lo-byte)
K1	Transform 8-bit data into 16-bit data (Hi-byte, Lo-byte)
K2	Transform 16-bit data into 8-bit data (Hi-byte, Lo-byte)
K3	Transform 16-bit data into 8-bit data (Hi-byte, Lo-byte)
K4	Transform 8-bit HEX data into ASCII data (higher 4 bits, lower 4 bits)
K5	Transform 8-bit HEX data into ASCII data (higher 4 bits, lower 4 bits)
K6	Transform 8-bit ASCII data into HEX data (higher 4 bits, lower 4 bits)
K7	Transform 8-bit ASCII data into HEX data (higher 4 bits, lower 4 bits)
K8	Transform 8-bit GPS data into 32-bit floating point data
K9	Calculate the optimal frequency function Available for ES2/EX2 V1.2, SS2/SA2/SX2/SE V1.0
K11	Conversion from local time to local sidereal time (longitude) Available for SA2 V1.0, SX2 V1.2, ES2/EX2 V2.0, SS2/SE V1.0

K12	Proportional value calculation function of multi-point areas (16-bit values) Available for SEV1.0, ES2/EX2 V2.4, SA2/SX2 V2.0, SS2 V2.2
K13	Proportional value calculation function of multi-point areas (32-bit values) Available for SEV1.0, ES2/EX2 V2.4, SA2/SX2 V2.0, SS2 V2.2
K14	Proportional value calculation function of multi-point areas (floating-point values); Available for SEV1.0, ES2/EX2 V2.4, SA2/SX2 V2.0, SS2 V2.2
K15	Calculate the local time for sunrise and sunset Available for ES2/EX2 V3.60, 12SA2/SX2 V3.00, ES2-E V1.4, SS2 V3.50, 12SE V1.92, 26SE V1.00, 28SA2 V2.90
K16	String combination function Available for SA2 /SE V1.0, SX2 V1.2, ES2/EX2/SS2 V2.0
K17	String capture function Available for SA2 /SE V1.0, SX2 V1.2, ES2/EX2/SS2 V2.0
K18	Convert data string to floating point value Available for S SA2 /SE V1.0, SX2 V1.2, ES2/EX2/SS2 V2.0
K19	Convert floating point value to data string Available for SA2 /SE V1.0, SX2 V1.2, ES2/EX2/SS2 V2.0
K30	Exchange the 16-bit data Available for ES2/EX2 V3.42, ES2-C V3.48, 28SA2 V1.0, 12SA2/SX2 V3.00
K31	Copy word type data to the consecutive registers of the PLC Available for ES2/EX2 V3.46, ES2-C V3.48, SA2/SX2 V2.86, SS2 V3.40
K32	Read the first written register to the D device (target value) and move the second written register to the position of the first written register and so on. (first in first out) Available for ES2/EX2 V3.46, ES2-C V3.48, SA2/SX2 V2.86, SS2 V3.40
K33	Read the last written register (last in first out) Available for ES2/EX2 V3.46, ES2-C V3.48, SA2/SX2 V2.86, SS2 V3.40
K34	Copy BIT type data to the consecutive registers of the PLC Available for ES2/EX2 V3.46, ES2-C V3.48, SA2/SX2 V2.86, SS2 V3.40
K35	Read the first written BIT data and move the second written BIT data to the position of the first written BIT data and so on. (first in first out) Available for ES2/EX2 V3.46, ES2-C V3.48, SA2/SX2 V2.86, SS2 V3.40
K36	Read the last written BIT data (last in first out) Available for ES2/EX2 V3.46, ES2-C V3.48, SA2/SX2 V2.86, SS2 V3.40
K39	Read PLC product serial number Available for: ES2/EX2/ES2-C V3.48, 12SA2/SX2 V3.00, ES2-E V1.00, SS2 V3.48, 12SE V2.00, 26SE V1.00, 28SA2 V2.88

4. Explanations on parameter settings of **m** operand:

K0: With n = 4, transform 8-bit data into 16-bit data (Hi-byte, Lo-byte) in the following rule:

Hi-byte	Lo-byte
	①
	②
	③
	④

⇨

Hi-byte	Lo-byte
①	②
③	④

K1: With n = 4, transform 8-bit data into 16-bit data (Lo-byte, Hi-byte) in the following rule:

Hi-byte	Lo-byte
	①
	②
	③
	④

⇨

Hi-byte	Lo-byte
②	①
④	③

K2: With n = 2, transform 16-bit data (Hi-byte, Lo-byte) into 8-bit data in the following rule:

K2 can work with K4, refer to example of K4 for more information.

Hi-byte	Lo-byte
①	②
③	④

⇨

Hi-byte	Lo-byte
	①
	②
	③
	④

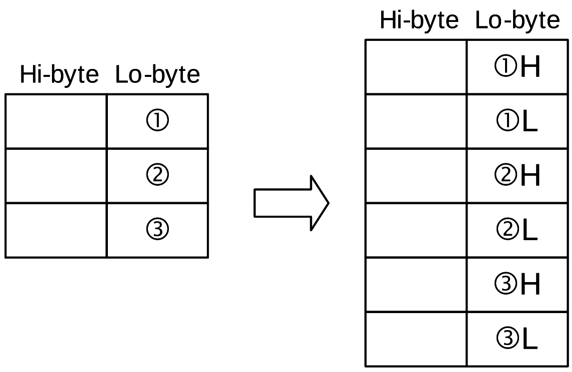
K3: With n = 2, transform 16-bit data (Lo-byte, Hi-byte) into 8-bit data in the following rule:

Hi-byte	Lo-byte
①	②
③	④

⇨

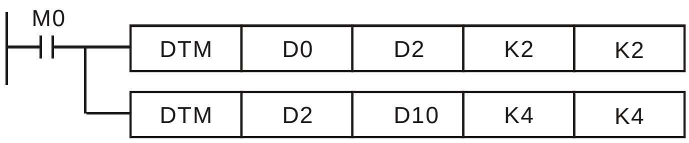
Hi-byte	Lo-byte
	②
	①
	④
	③

K4: With n = 3, transform 8-bit HEX data into ASCII data (higher 4 bits, lower 4 bits) in the following rule:



Example: Use both K2 and K4 at a time

1. When M0 = ON, transform 16-bit data in D0, D1 into ASCII data in the following order: H byte - L byte - H byte - Low byte, and store the results in D10.
2. Move the 16-bit data to where the data of the L-byte are.
3. Transform 8-bit HEX data into ASCII data



- Value of source devices D0, D1:

Register	D0	D1
Value	H1234	H5678

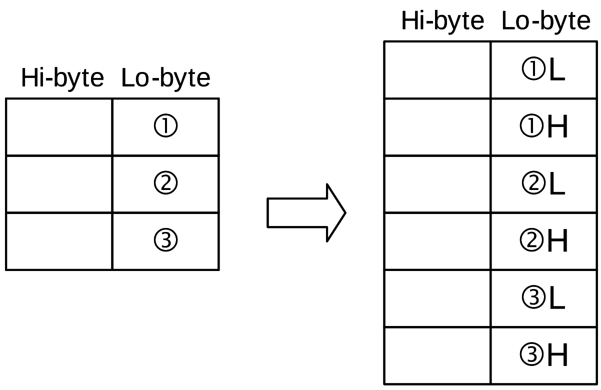
- When the 1st DTM instruction executes (m=K2), ELC transforms the 16-bit data (Hi-byte, Lo-byte) into 8-bit data and move to registers D2~D5.

Register	D2	D3	D4	D5
Value	H12	H34	H56	H78

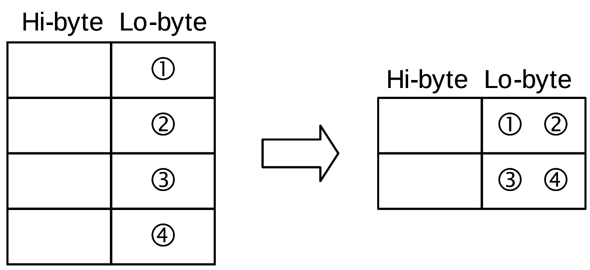
- When the 2nd DTM instruction executes (m=K4), ELC transforms the 8-bit HEX data into ASCII data and move to registers D10~D17.

Register	D10	D11	D12	D13	D14	D15	D16	D17
Value	H0031	H0032	H0033	H0034	H0035	H0036	H0037	H0038

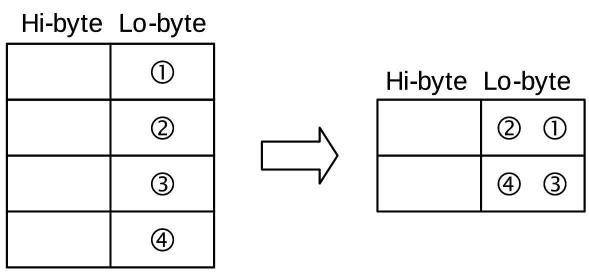
K5: With n = 3, transform 8-bit HEX data into ASCII data (lower 4 bits, higher 4 bits) in the following rule:



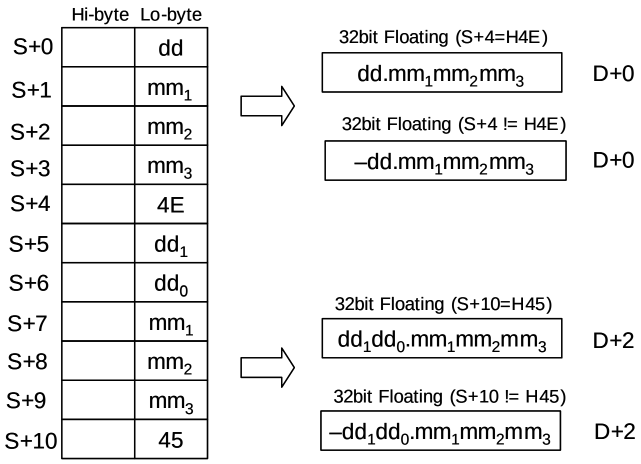
K6: When n = 4, transform 8-bit ASCII data (higher 4 bits, lower 4 bits) into HEX data in the following rule: (ASCII value to be transformed includes 0 ~ 9 (0x30~0x39), A ~ F (0x41~0x46), and a ~ f (0x61~0x66).)



K7: When n = 4, transform 8-bit ASCII data (lower 4 bits, higher 4 bits) into HEX data in the following rule:



K8: Transform 8-bit GPS data into 32-bit floating point data in the following rule:



K9: Calculate the optimal frequency for positioning instructions with ramp up/ down function.

- Users only need to set up the total number of pulses for positioning and the total time for positioning first, DTM instruction will automatically calculate the optimal max output frequency as well as the optimal start frequency for positioning instructions with ramp-up/down function such as PLSR, DDRVI and DCLLM.

Points to note:

1. When the calculation results exceed the max frequency of PLC, the output frequency will be set as 0.
2. When the total of ramp-up and ramp-down time exceeds the total time for operation, PLC will change the total time for operation (**S**+2) into "ramp-up time (**S**+3) + ramp-down time (**S**+4) + 1" automatically.

Explanation on operands:

S+0, **S**+1: Total number of pulses for operation (32-bit)

S+2: Total time for operation (unit: ms)

S+3: Ramp-up time (unit: ms)

S+4: Ramp-down time (unit: ms)

D+0, **D**+1: Optimal max output frequency (unit: Hz) (32-bit)

D+2: Optimal start frequency (Unit: Hz)

n: Reserved

Example: K9

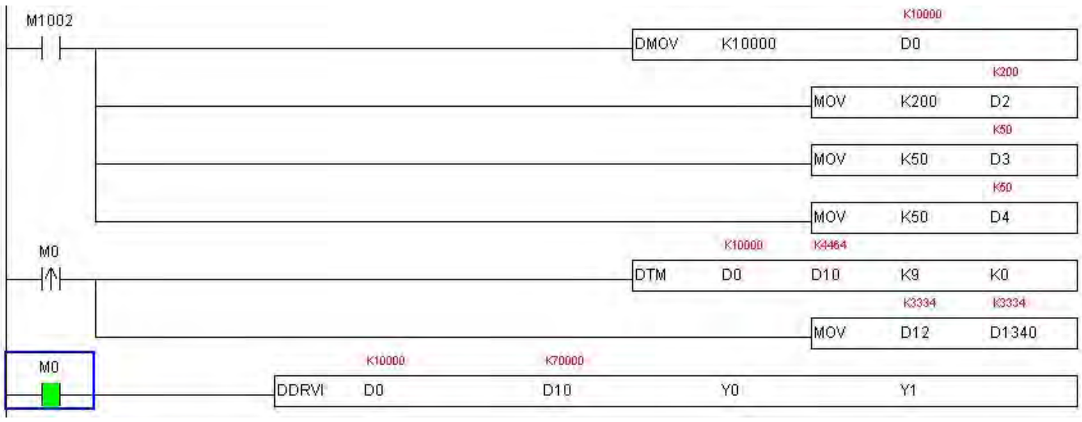
1. Set up total number of pulses, total time, ramp-up time and ramp-down time in source device starting with D0. Execute DTM instruction and the optimal max frequency as well as optimal start frequency can be obtained and executed by positioning instructions.

2. Assume the data of source device is set up as below:

Total Pulses	Total Time	Ramp-up Time	Ramp-down Time
D0, D1	D2	D3	D4
K10000	K200	K50	K50

3. The optimal positioning results can be obtained as below:

Optimal max frequency	Optimal start frequency
D10, D11	D12
K70000	K3334



K11: Conversion from Local Time to Local Sidereal Time

Unlike the common local time defined by time zones, local sidereal time is calculated based on actual longitude. The conversion helps the user obtain the more accurate time difference of each location within the same time zone.

Explanation on operands:

S+0, **S**+1: Longitude (32-bit floating point value; East: positive, West: negative)

S+2: Time zone (16-bit integer; unit: hour)

S+3~ **S**+8: Year, Month, Day, Hour, Minute, Second of local time (16-bit integer)

D+0~**D**+5: Year, Month, Day, Hour, Minute, Second of the converted local sidereal time (16-bit integer)

n: Reserved

Example:

Input: Longitude F121.55, Time zone: +8, Local time: AM 8:00:00, Jan/6/2011

Conversion results: AM 8:06:12, Jan/6/2011

K12: Proportional Value Calculation Function of Multi-point Areas (16-bit values)

Explanation on operands (16-bit values):

S: input value

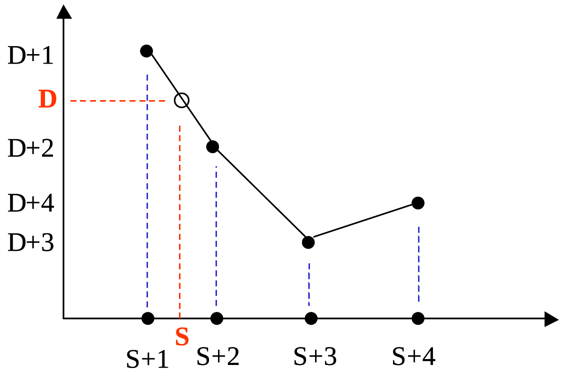
S+1, **S**+2….. **S**+n: set values of multi-point areas. **S**+1 must be the minimum value, **S**+2 must be larger than **S**+1 and so on. Therefore, **S**+n must be the maximum value.

D: output value gotten from the proportional value calculation

D+1, **D** +2 … **D**+n: the range of values gotten from the proportional value calculation

n: set values of multi-point areas. The range of set values is K2~K50. When the set value exceeds the range, it will not be executed.

The sample curve: (n is set to be K4)



The explanation of the sample:

1. When input value S is larger than S+1 (S_1 for short) and smaller than S+2 (S_2 for short), D+1 (D_1 for short) and D+2 (D_2 for short), D= $((S - S_1) \times (D_2 - D_1) / (S_2 - S_1)) + D_1$.
2. When input value S is smaller than S+1, D= D+1; when input value S is larger than S+n, D= D+n.
3. The operation of instructions uses floating-point values. After the decimal value of the output values is omitted, the value will be output in the 16-bit form.

K13: Proportional Value Calculation Function of Multi-point Areas (32-bit values)

The explanations of source and destination devices are illustrated as the explanation of K12, but devices S and D are indicated by 32-bit values.

K14: Proportional Value Calculation Function of Multi-point Areas (floating-point values)

The explanations of source and destination devices are illustrated as the explanation of K12, but devices S and D are indicated by 32-bit floating-point values.

K15: to Calculate the local time for sunrise and sunset

Explanation on operands **S, D, m, n**:

S:

S +0, **S**+1: the local longitude (floating-point format)

S+2, **S**+3: the local latitude (floating-point format)

S+4: the local time zone (integer format)

S+5: to see if the daylight saving is enabled or not (integer format); 0: disabled

S+6, **S**+7, **S**+8: the present year, month and date (integer format); you need to enter the last two digits of the year and the first two digits will be filled by PLC.

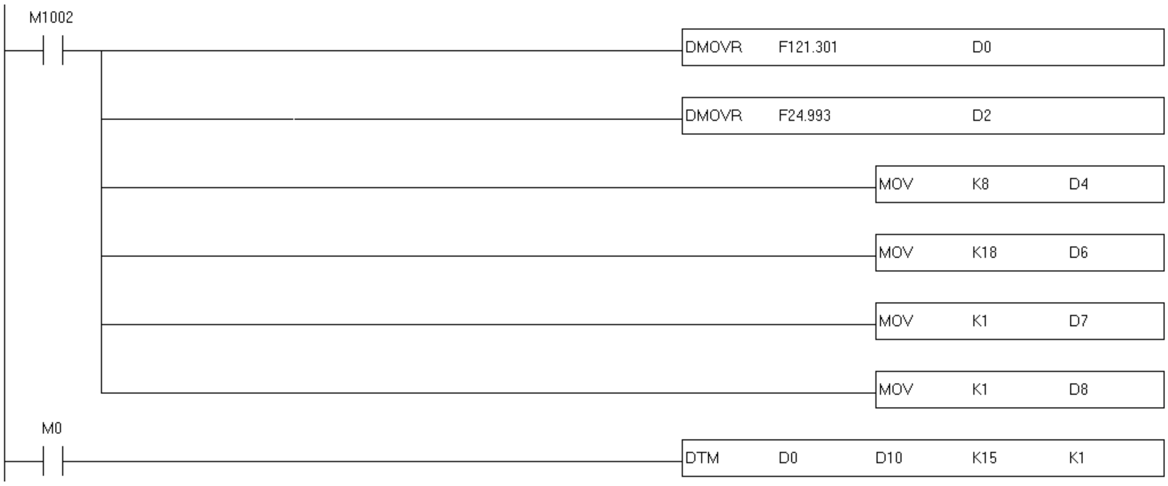
D:

D+0, **D**+1, **D**+2: calculated sunrise time (24 hour format); hr : min : sec (integer format)

D+3, **D**+4, **D**+5: calculated sunset time (24 hour format); hr : min : sec (integer format)

m: K15

n: Reserved



Calculate the time to sunrise and sunset in Taoyuan, Taiwan on January 1st, 2018. The official time zone in Taiwan is defined by an UTC offset of +08:00. And daily saving time is NOT implemented in Taiwan. After calculation, the time to sunrise is at 06:39:47 and the time to sunset is at 17:16:42. See the example program below. The allowable range is from +3 to -3.

Enter values for the local longitude and latitude in numbers. For example, the longitude and latitude of Taoyuan, Taiwan is 121.30098 and 24.99363. Latitudes north of the Equator are denoted by a positive sign. Latitudes south of the Equator are given negative values.

Register	Inputted value	Register	Calculated value
D0 (longitude)	F121.301	D10 (sunrise: hour)	6
D2 (latitude)	F24.993	D11 (sunrise: minute)	39
D4 (time zone)	K8	D12 (sunrise: second)	47
D6 (year)	K18	D13 (sunset: hour)	17
D7 (month)	K1	D14 (sunset: minute)	16
D8 (date)	K1	D15 (sunset: second)	42

K16: String combination

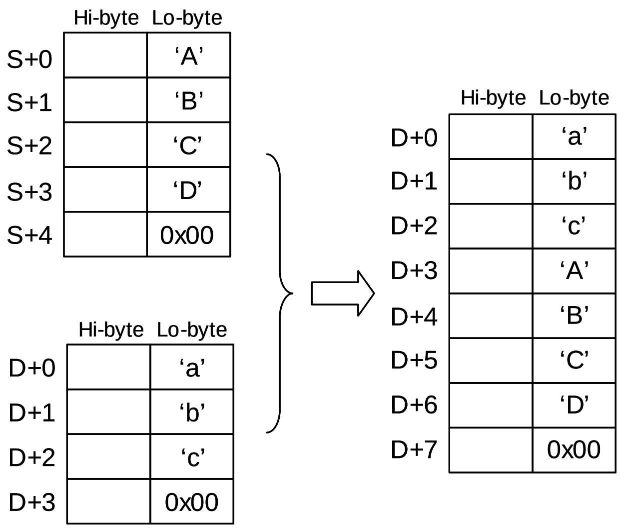
Explanation:

The system searches for the location of ETX (value 0x00) of the destination data string (lower 8 bits), then copies the data string starting of the source register (lower 8 bits) to the end of the destination data string. The source data string will be copied in byte order until the ETX (value 0x00) is reached.

Points to note:

The operand n sets the max data length after the string combination (max 256). If the ETX is not reached after the combination, the location indicated by n will be the ETX and filled with 0x00.

The combination will be performed in the following rule:



K17: String capture

Explanations:

The system copies the source data string (lower 8 bits) with the data length specified by operand n to the destination registers, where the n+1 register will be filled with 0x00. If value 0x00 is reached before the specified capture length n is completed, the capture will also be ended.

The capture will be performed by the following rule:

	Hi-byte	Lo-byte
S+0		‘a’
S+1		‘b’
S+2		‘c’
S+3		‘A’
S+4		‘B’
S+5		‘C’
S+6		‘D’
S+7		0x00

n = k3 ⇨

	Hi-byte	Lo-byte
D+0		‘a’
D+1		‘b’
D+2		‘c’
D+3		0x00

K18: Convert data string to floating point value

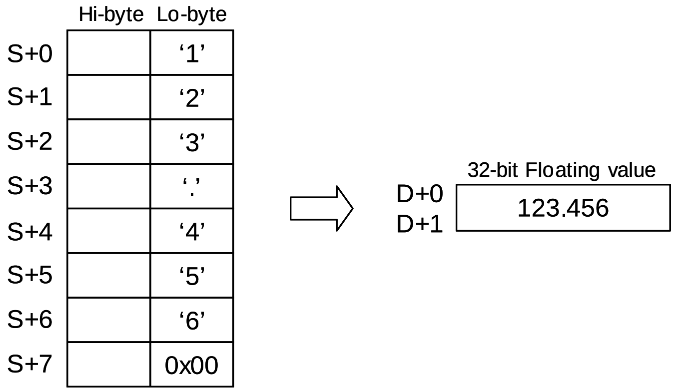
Explanations:

The system converts n words (lower 8 bits) of the source data string (decimal point is not included) to floating point value and stores the converted value in the destination device D. The ending character 0x00 does not work in K18.

Points to note:

1. Operand **n** sets the number of total digits for the converted floating value. Max 8 digits are applicable and the value over **n** digit will be omitted. For example, **n** = K6, data string “123.45678” will be converted to “123.456”.
2. When there are characters other than numbers 0~9 or the decimal point in the source data string, the character before the decimal point will be regarded as 0, and the value after the decimal point will be regarded as the ETX. Thus if you put + or – before the number, it will also be seen as 0.
3. If the source data string contains no decimal point, the converted value will be displayed by a **n**-digit floating point value automatically.

The conversion will be performed in the following rule:



K19: Convert floating point value to data string

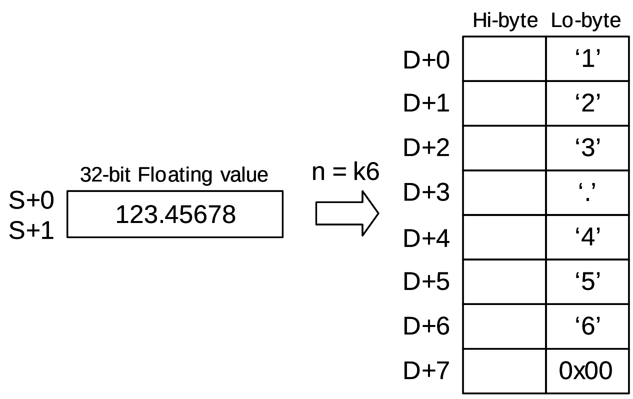
Explanations:

The system converts the floating point value in the source device S to data string with specified length **n** (decimal point is not included).

Points to note:

1. Operand **n** sets the number of total digits for the floating point value to be converted. Max 8 digits are applicable and the value over **n** digit will be omitted. For example, **n** = K6, floating value F123.45678 will be converted to data string “123.456”.
2. When the digits of source value are more than the specified **n** digits, only the **n** digits from the left will be converted. For example, source value F123456.78 with **n**=K4 will be converted as data string "1234”.
3. If the source value is a decimal value without integers, e.g. 0.1234, the converted data string will be “.1234” where the first digit is the decimal point.
4. Only supports conversion of positive floating point numbers, and only supports floating point numbers less than 100000000.0.

The conversion will be performed in the following rule:



K30: Swap 16-bit data

Swap the Bit data stored in S+0~S+(N-1) to D+0~D+(N-1).

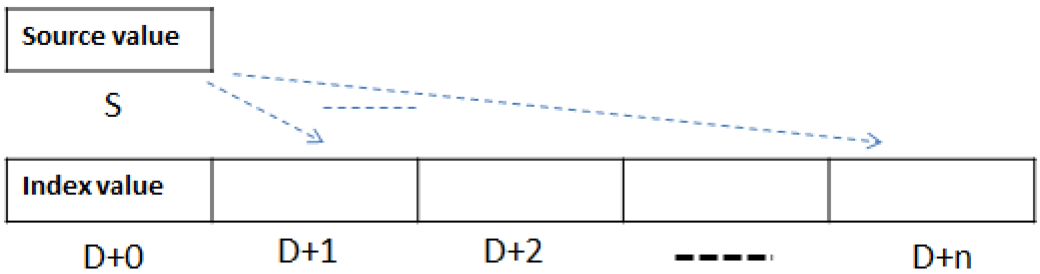
The movement of BIT SWAP: BIT15⇔BIT0, BIT14⇔BIT1, BIT13⇔BIT2 and so on.

Example: DTM D0 D10 K30 K8

D0 = 0x0001	D10 = 0x8000
D1 = 0x0002	D11 = 0x4000
D2 = 0x0004	D12 = 0x2000
D3 = 0x0008	D13 = 0x1000
D4 = 0x0010	D14 = 0x0800
D5 = 0x0020	D15 = 0x0400
D6 = 0x0040	D16 = 0x0200
D7 = 0x0080	D17 = 0x0100

K31: Copy word type data to the consecutive registers of the PLC

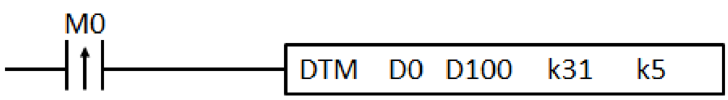
Copy the source value stored in S to the target device as the index value indicated and then accumulate 1 to the index value.



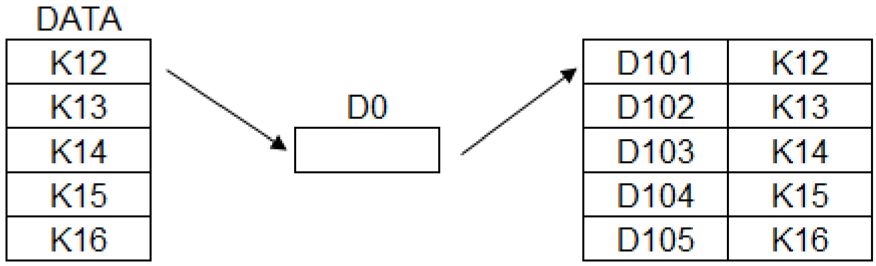
Note1: when the index value (D+0) is less than 1, it will be treated as 1 and the actions of data copy and accumulation begin. When the index value (D+0) is bigger than n (default: n+1), the action of data copy will not begin.

Note 2: D1000~D1999 cannot be used as D devices.

Example:



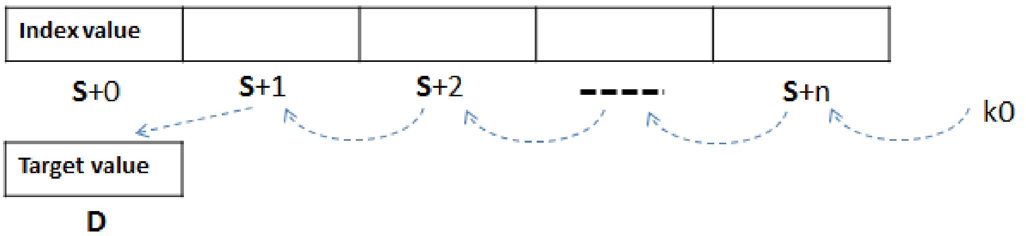
1. If M0 switches OFF→ON for 5 times in a row, execute the instruction DTM will copy the values stored in D0 to D101~105, as the image shown below:



2. Adding one to the value stored in D100, after the execution of the DTM instruciton is complete.

K32: Read the first written register to the D device (target value) and move the succeeding registers forward, for example, move the second register to where the first written register was and so on. (first in first out)

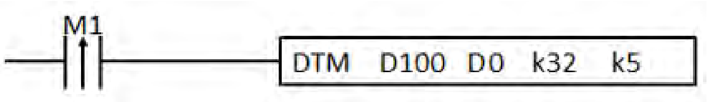
Read and store the data stored from S+1 to the D device and move the value in S+2 forward to S+1. Put k0 to the last and then diminish the index value (S+0) by 1.



Note:

When the value in S+0 is less than 2, it means there is no data to be read/moved and no action will be taken. When the value in S+0 is bigger than n+1, it means the data is full and no action will be taken. “No action will be taken” means no error will be displayed nor the index value (S+0) will have any change.

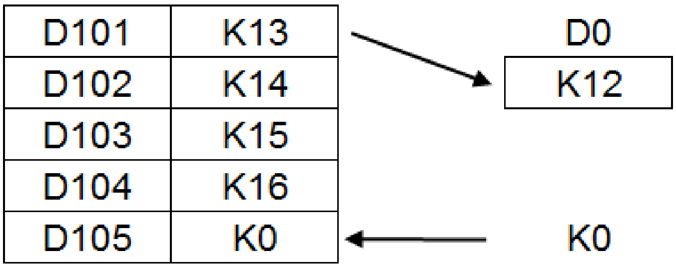
Example:



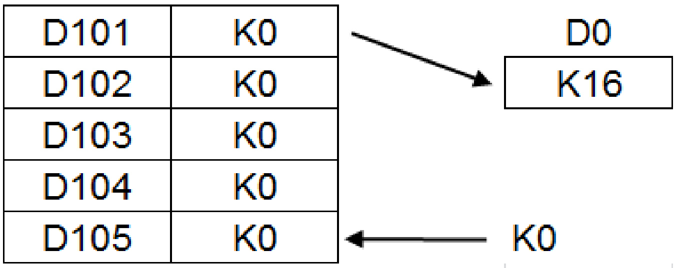
1. If D100=6.

2. If M0 switches OFF→ON, execute the instruction DTM will copy the values stored in D101 to D0.

3. As the image shown below, execute the instruciton DTM to copy the value K12 stored in D101 to D0 and put K0 to D105.

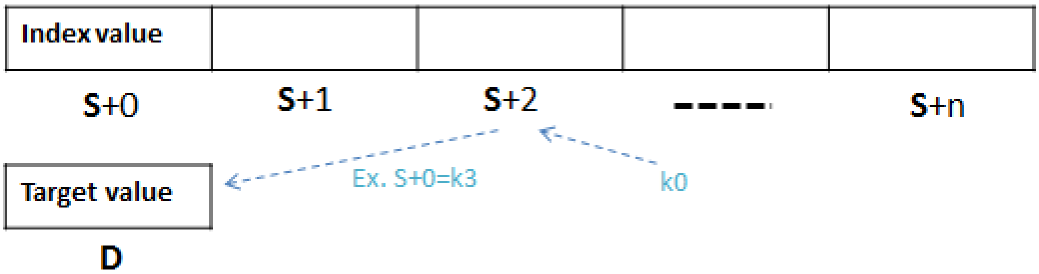


4. Execute the instruciton DTM for 5 times to have the results: D0=K16 and values in D101~105 are K0.



K33: Read the last written register to the D device (target value) (last in first out).

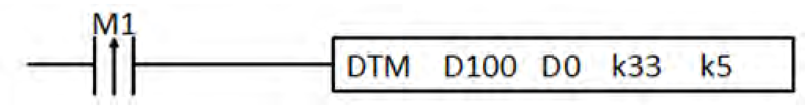
Diminish the index value (S+0) by 1 and then read and store the data stored from S+[**S**+0] to the D device and put K0 to the source value S+[**S**+0].



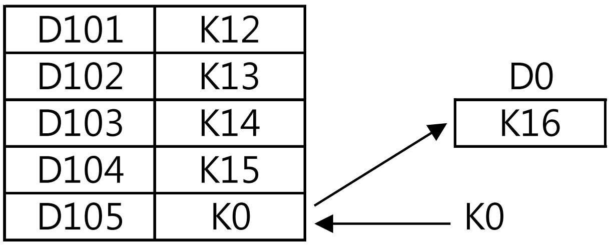
Note:

When the value in S+0 is less than 2, it means there is no data to be read/moved and no action will be taken. When the value in S+0 is bigger than n+1, it means the data is full and no action will be taken. “No action will be taken” means no error will be displayed nor will the index value (S+0) have any change.

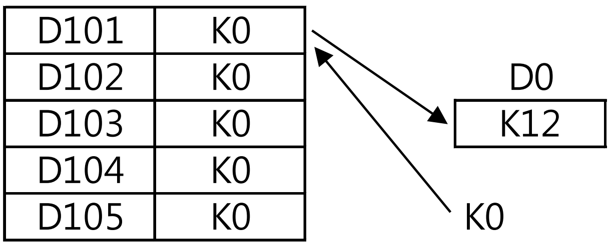
Example:



1. If D100=6.
2. If M0 switches OFF→ON, execute the instruction DTM will copy the values stored in D105 to D0.
3. As the image shown below, execute the instruciton DTM to copy the value K16 stored in D105 to D0 and put K0 to D105.



4. Execute the instruciton DTM for 5 times to have the results: D0=K12 and values in D101~105 are K0.

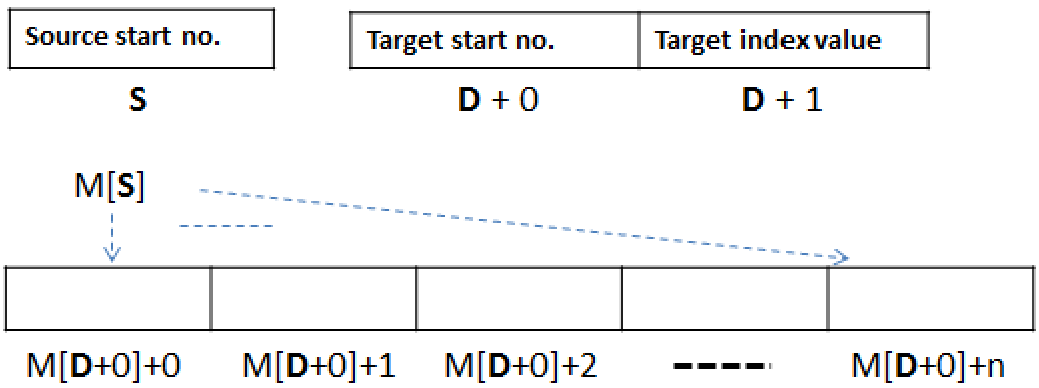


K34: Copy BIT type data to the consecutive registers of the PLC

S: the source start number of the M device

D: the target start number of the M device

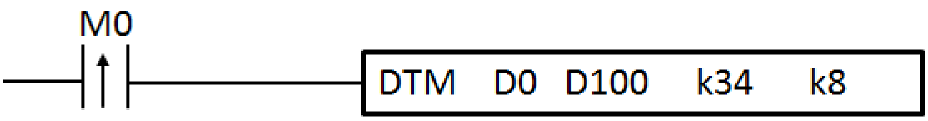
Copy the M state from source value stored in S to the M[**D**+0] (target device) as the target index value indicated and then accumulate 1 to D+1 (target index value).



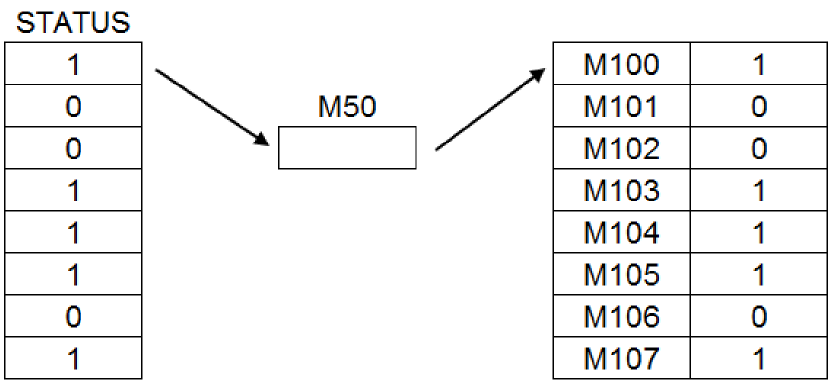
Note1: when the target index value (D+1) is less than 0, change D+1 to 0 and the actions of data copy and accumulation begin. When the target index value (D+1) is bigger than n-1, change D+1 to n; the action of data copy will not begin.

Note 2: M1000~M1999 cannot be used as M devices.

Example:



1. Set D0=K50 and D100=K100, and execute the instruction DTM will copy the values stored in M50 to M100~107. After the execution of DTM is complete, add one to the value stored in D101.
2. Execute the instruciton DTM for 8 times to have the results as shown below:

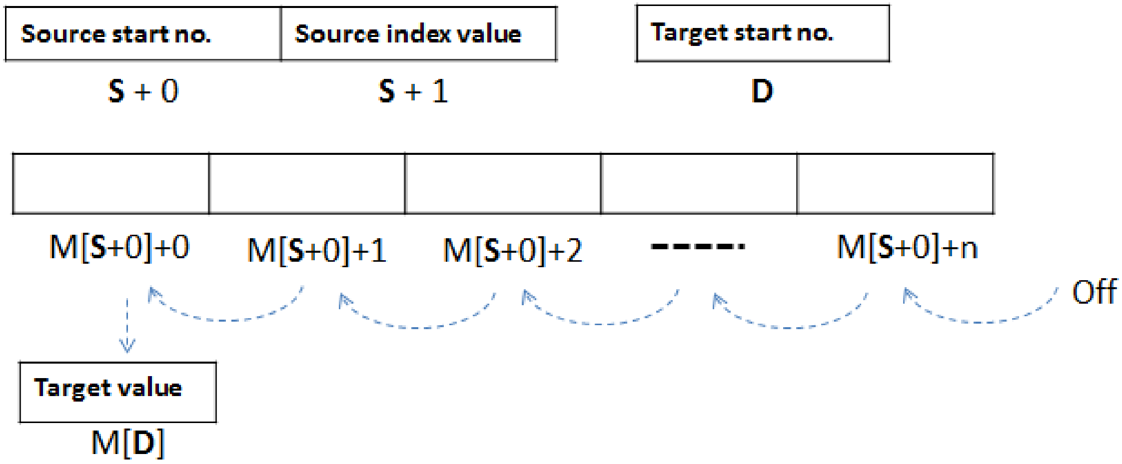


K35: Read the first written BIT data to the BIT device (target value) and move the succeeding BIT data forward, for example, move the second BIT data to where the first written BIT data was and so on. (first in first out)

S: the source start number of the M device

D: the target start number of the M device

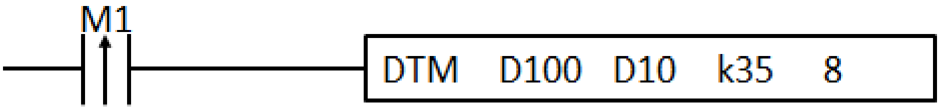
Read and store the state stored from M[**S**+0]+0 to the M[**D**] (target value) and move the state of the succeeding forward and change the state of the last to OFF and then diminish the index value (S+1) by 1.



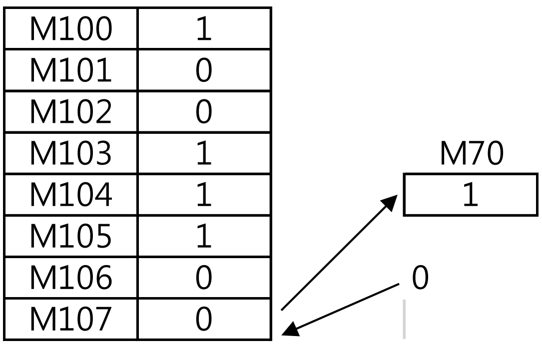
Note 1: When the value in S+1 is less than 1, it means there is no data to be read/moved and no action will be taken. When the value in S+1 is bigger than n, it means the data is full and no action will be taken. “No action will be taken” means no error will be displayed nor will the index value (S+1) have any change.

Note 2: M1000~M1999 cannot be used as M devices.

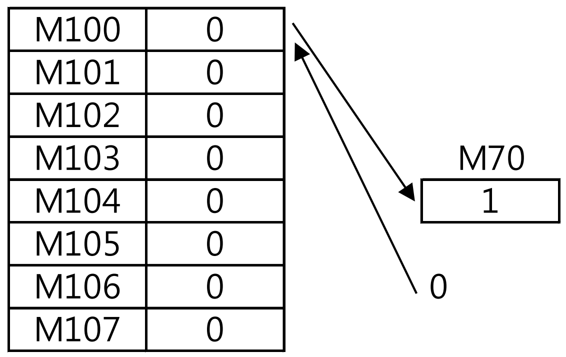
Example:



1. Set D100=K100 and D10=K70, and execute the instruction DTM will copy the states in M100~107 to M70. After the execution of DTM is complete, add one to the value stored in D101.
2. Execute the instruciton DTM to move the stae 1 in M107 to M70 and put 0 in M107 as shown below:

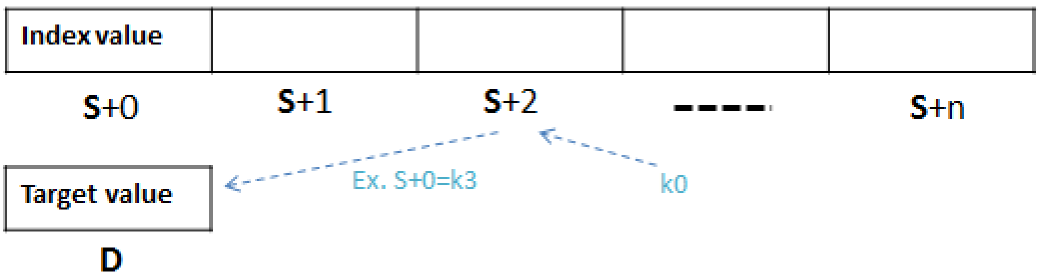


3. Execute the instruciton DTM for 8 times to have the results: M70=1 and the states in M100~M107 are 0.



K36: Read the last written register to the D device (target value) (last in first out).

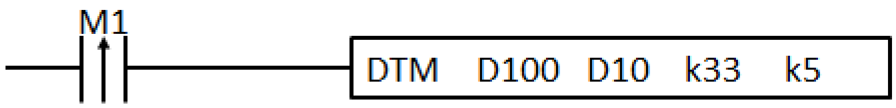
Diminish the index value (S+0) by 1 and then read and store the data stored from S+[**S**+0] to the D device and put K0 to the source value S+[**S**+0].



Note:

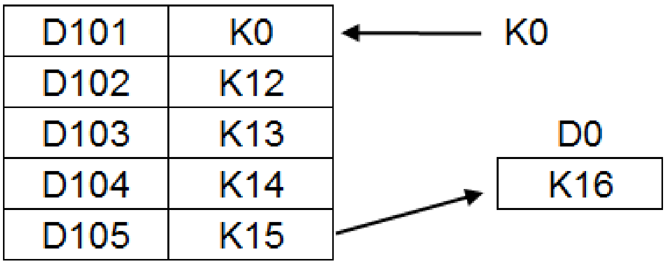
When the value in S+0 is less than 2, it means there is no data to be read/moved and no action will be taken. When the value in S+0 is bigger than n+1, it means the data is full and no action will be taken. “No action will be taken” means no error will be displayed nor the index value (S+0) will have any change.

Example:

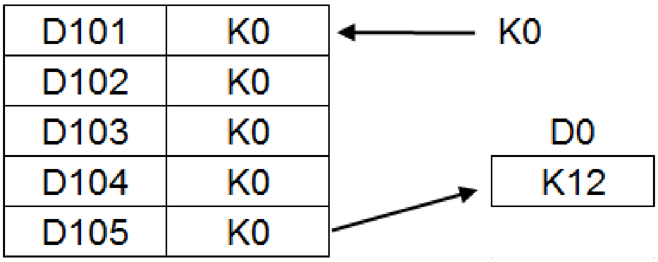


4. If M0 switches OFF→ON, execute the instruction DTM will copy the values stored in D105 to D0.

5. As the image shown below, execute the instruciton DTM to copy the value K16 stored in D105 to D0 and put K0 to D101.



6. Execute the instruciton DTM for 5 times to have the results: D0=K12 and values in D101~105 are K0.



K39: Read the PLC product serial number

S and n: no use; if you put any number here, there will be no action.

D: for storing the length of the product serial number

D1, D2, D3, and so forth: for storing the product serial number

Use K39 to read the length of PLC product serial number and the PLC product serial number.

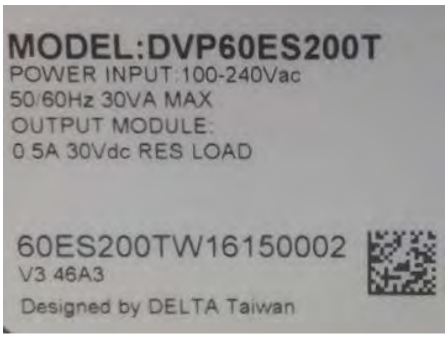
The length of the PLC product serial number is the actual length. You cannot change the value here.

The value (the PLC product serial number) in the registers D1, D2 and so forth should be in WORD format, if not, use H00 in the last byte to form a WORD.

Example:

Read the PLC product serial number and store the information in D0 (definable).

D0 is used to store the length of the PLC product serial number. D1, D2 and so forth is used in ASCII code, high byte and low byte to store the PLC product serial number. Up to 9 data registers can be used to store the PLC product serial number.



Use the instruction to read the PLC product serial number. (suggested to use single trigger to activate the instruction)

The followings can be read.

Device Name	Present Value (16 bits)	Format
D0	K17	Signed Decimal
D1	60	ASCII
D2	ES	ASCII
D3	20	ASCII
D4	0T	ASCII
D5	W1	ASCII
D6	61	ASCII
D7	50	ASCII
D8	00	ASCII
D9	2*	ASCII

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
69	D	SORT	S m1 m2 D n	Data sort				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SORT: 11 steps
S													*			DSORT: 21 steps
m_1					*	*										
m_2					*	*										
D													*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

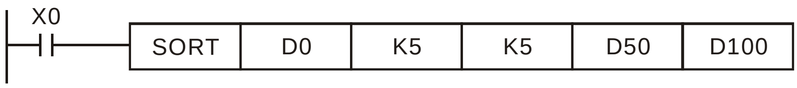
S: Start device for the source data **m_1**: Groups of data to be sorted (**m_1** =1~32) **m_2**: Number of columns in the table (**m_2** =1~6) **D**: Start device for the sorted data **n**: The No. of column to be sorted. (**n**=1~ **m2**)

Explanations:

1. The sorted data is stored in the **m_1** × **m_2** registers starting from the device designated in **D**. Therefore, if **S** and **D** designate the same register, the sorted results will be the same.
2. It is better that the rightmost number of the device number of the register specified by **S** is 0.
3. SORT instruction is completed after **m_1** times of scan. Once the SORT instruction is completed, the Flag M1029 (Execution completed flag) = ON.
4. There is no limitation on the times of using this instruction in the program. However, only one instruction can be executed at a time
5. The function of sorting one-dimensional data is added. If **m_1** is 1, and **m_2** is 1, the function will be enabled, and the operand **n** represents the number of data (**n**=1~32). The data in **n** devices starting from the operand **S** are sorted. The sort result is stored in the devices starting from the operand **D**. It takes one scan cycle for the data to be sorted. After the data is sorted, M1029 will be On. This function supports SS2 V3.0/SA2 V2.6/SX2 V2.4/ES2/EX2/ES2-C V3.2.

Program Example:

When X0 = ON, the sorting process starts. When the sorting is completed, M1029 will be ON. DO NOT change the data to be sorted during the execution of the instruction. If the sorting needs to be executed again, turn X0 from OFF to ON again.



Example table of data sort

Columns of data: **m_2**

Groups of data: **m_1**

	Data Column				
Column / Row	1	2	3	4	5
	Students No.	English	Math.	Physics	Chemistry
1	(D0) 1	(D5) 90	(D10) 75	(D15) 66	(D20) 79
2	(D1) 2	(D6) 55	(D11) 65	(D16) 54	(D21) 63
3	(D2) 3	(D7) 80	(D12) 98	(D17) 89	(D22) 90
4	(D3) 4	(D8) 70	(D13) 60	(D18) 99	(D23) 50
5	(D4) 5	(D9) 95	(D14) 79	(D19) 75	(D24) 69

Sort data table when D100 = K3

Columns of data: **m_2**

Groups of data: **m_1**

	Data Column				
Column / Row	1	2	3	4	5
	Students No.	English	Math.	Physics	Chemistry
1	(D50) 4	(D55) 70	(D60) 60	(D65) 99	(D70) 50
2	(D51) 2	(D56) 55	(D61) 65	(D66) 54	(D71) 63
3	(D52) 1	(D57) 90	(D62) 75	(D67) 66	(D72) 79
4	(D53) 5	(D58) 95	(D63) 79	(D68) 75	(D73) 69
5	(D54) 3	(D59) 80	(D64) 98	(D69) 89	(D74) 90

Sort data table when D100 = K5

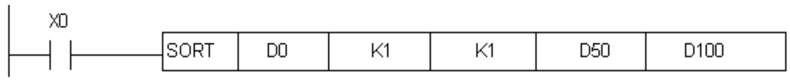
Columns of data: **m_2**

Groups of data: **m_1**

	Data Column				
Column / Row	1	2	3	4	5
	Students No.	English	Math.	Physics	Chemistry
1	(D50) 4	(D55) 70	(D60) 60	(D65) 99	(D70) 50
2	(D51) 2	(D56) 55	(D61) 65	(D66) 54	(D71) 63
3	(D52) 5	(D57) 95	(D62) 79	(D67) 75	(D72) 69
4	(D53) 1	(D58) 90	(D63) 75	(D68) 66	(D73) 79
5	(D54) 3	(D59) 80	(D64) 98	(D69) 89	(D74) 90

Program Example 1: (Sorting one-dimensional data)

If X0 is On, the data specified will be sorted. After the data is sorted, M1029 will be On.



If **m₁** is K1, and **m₂** is K1, one-dimensional data will be sorted. The value in D100 is K5. The values in D0~D4 are shown below.

1. The values in D0~D4 are listed below.

Data source (**S**)	D0	D1	D2	D3	D4
Data	75	65	98	60	79

2. The sort result is stored in D50~D54.

Sort result (**D**)	D50	D51	D52	D53	D54
Data	60	65	75	79	98

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2	SS2	SE	SA2/ SX2
315	XCMP	**S**1, **S**2, **S**3, **S**4, **D**	Setting up to compare the inputs of multiple work stations				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	XCMP: 11 steps
S1	*															
S2												*				
S3													*			
S4													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SE	SA2/ SX2	ES2/EX2	SS2	SE	SA2/ SX2	ES2/EX2	SS2	SE	SA2/ SX2

Operands:

S1 : Trigger input point

S2 : High-speed counter number

S3 : Setting for the numbers for work station and objects

S4 : Reference value for comparison and the observational error

D : First corresponding device for the comparison result in the stack area

Explanation

1. Use **S1** for setting the trigger input points; for ES2 series, use built-in inputs X4 and X6 for immediate trigger input points and other inputs from X0 to X17 for general trigger input points. Executing the instruction enables the external interrupts for the inputs. Therefore it is suggested that you not use the inputs with interrupt tasks; otherwise, when the instruction is executed, the interrupts are disabled and resumed only after the instruction completes. The general type inputs are affected by the scan time though they are suitable for the environments where the inputs are not as stable.

2. **S2** works with 32-bit counters (C200–C255) and is limited to accumulated count. When the inputs are the high-speed trigger input type, it is suggested that you implement the hardware high-speed counter such as C251 or C253 and use the DCNT instruction to enable the counter. When you need high-speed output, you can use the DMOV instruction to copy the output current position; for example copying the current output position D1030 to C200.

3. **S3** occupies seven consecutive 16-bit devices. **S3**+0 is n (the work station number) and **S3**+1 is m (the maximum object number). **S3**+2 is the result of the object being filtered. **S3**+3 (Low word) and **S3**+4 (High word) are the result of rising-edge triggered number (32-bit). **S3**+5 (Low word)

and **S_3**+6 (High word) are the result of rising-edge triggered number (32-bit). The range for n and m is between 1–32. The range for n and m is between 1–32. When this value is out of range, the value used is the maximum (32) or the minimum (1). The range for **S_3**+2 (the number of filter) is between 0–32767. Zero is used for any value less than 0 ; and a value of 0 disables the filtering function. It is suggested that you declare an array of 3 words or 3 consecutive word type variables.

4. It is suggested that you set the maximum number for **S_3**+1 (m). If m < n, note the objects and make sure they are sufficient on the production line.

5. For **S_3**, the supported range in **n**, and **m** is 1 to 64.

Module	ES2/ EX2	ES2-C	ES2-E	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware version	V3.68	V3.68	V1.48	V3.02	N/A	V2.08	V2.08	V2.90

6. **S_4** occupies 3xn consecutive 32-bit devices (6xn 16-bit devices). If the required space exceeds the range of device D, the instruction is not executed. The value of n is the work station number set in the operand **S_3**. The following table lists the functions for each device and the corresponding number for **S_4**. It is suggested that you declare an array of 3n double words or 3 consecutive double word type variables for **S**.

Function	Work station 1	Work station 2	···	Work station n
Reference value for comparison (32-bit)	**S_4**+0	**S_4**+2	···	**S_4**+(n-1)x2
Observational error when entering (32-bit)	**S_4**+2xn	**S_4**+2xn+2	···	**S_4**+(2xn-1)x2
Observational error when leaving (32-bit)	**S_4**+4xn	**S_4**+4xn+2	···	**S_4**+(3xn-1)x2

7. When you set the reference value to 0 for a specific work station, the specific work station stops working. You can use this technique to manage work stations.

8. **D** is the first corresponding device for the comparison result in the stack area. **D** occupies 2xn consecutive 16-bit devices and 2xmxn consecutive 32-bit devices (or 4xmxn consecutive 16-bit devices). If the required space exceeds the range of device D, the instruction is not executed. The following table lists the functions for each device and the corresponding number for **D**.

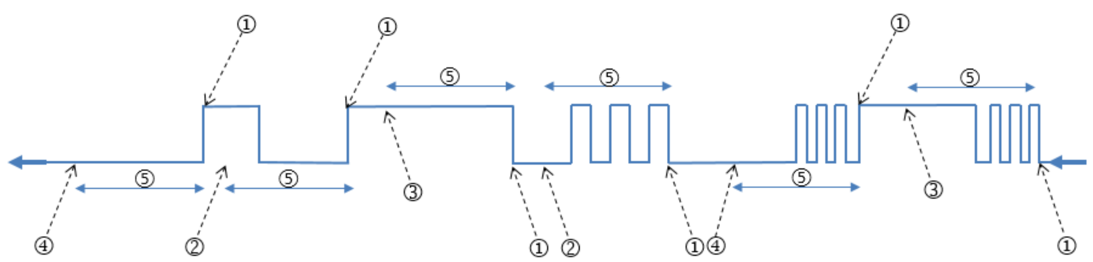
Function	Work station 1	Work station 2	···	Work station n
Value of the head index (16-bit)	**D**+0	**D**+1	···	**D**+(n-1)
Value of the tail index (16-bit)	**D**+n	**D**+(n+1)	···	**D**+(2xn-1)
Compared counter result 1 of the object when entering (32-bit)	**D**+2xn	**D**+2xn+2	···	**D**+2xn+2(n-1)
Compared counter result 1 of the object when leaving (32-bit)	**D**+4xn	**D**+4xn+2	···	**D**+4xn+2(n-1)
:	:	:	:	:
Compared counter result m of the object when entering (32-bit)	**D**+4xmxn-2xn		···	**D**+4xmxn-2
Compared counter result m of the object when leaving (32-bit)	**D**+4xmxn		···	**D**+4xmxn+2 (n-1)

9. **D** tends to occupy more space in the stack area. If the required space exceeds the range of device **D**, the PLC only executes what is valid in the storage and does not show a no warning. It is suggested that you declare an array of 2xn+4xmxn words for **D**.

10. There is no limit on the number of times you can execute the instruction but only one execution can be done at a time.

11. It is suggested to use this instruction with the YOUT instruction (API 0710) and use the same first corresponding device for the comparison result in the stack area (**D**).

12. The following timing diagram shows executing the high-speed counter and filter (reading from right to left).



①. PLC reads the current counter value.

②. Drop the counter value: the number of filters read is less than the number of filters set.

③. Record the counter value: the signal is high (ON time) and records the counter value to the comparing stack area for entering.

④. Record the counter value: the signal is low (OFF time) and records the counter value to the comparing stack area for leaving.

⑤. Unsettled pulse section

13. When the signal is rising- or falling-edge triggered, and the PLC completes processing the filters, the PLC reads the high-speed counter value and adds one in the value of the head index. The PLC then records the entering and leaving counter results for each work station. The compared counter result is the current counter value + reference value + observational error. For either rising- or falling-edge triggered, the value of the head index is incremented. The maximum value for the head index mx2 (the maximum number of objects).

14. The value of the head index is cyclically incremented, when the signal is rising- or falling-edge triggered and completes processing the number of filters (the default for trigger input is OFF). The maximum value for the head index is mx2 (the maximum number of objects). For example, if you set the number of objects to 10, the value of the head index (default: 0) is incremented to 1, 2, 3 to 20 and then 1, 2, 3 to 20 repeatedly. When the value of the head index is 0, it means no object has entered after executing the instruction. The PLC adds one to the value of the head index, and then checks the value of the tail index. If the value (after adding one) in the value of the head index equals the value of the tail index, the PLC cancels the addition and records the counter result.

15. When the instruction is executed and the state of the initial input is **OFF**, the **rising-edge trigger** corresponds to the **odd numbers** of the head index value, and the ***falling-edge trigger*** corresponds to the ***even numbers*** of the head index value.

16. When the PLC executes the instruction and the state of the initial input is **ON**, the **falling-edge trigger** corresponds to the **odd numbers** of the head index value, and the ***rising-edge trigger*** corresponds to the ***even numbers*** of the head index value.

17. When the PLC executes the instruction, it does not clear the values in the accumulated area and the index areas. If the data is in a latched area and needs to be enabled again, use the ZRST instruction to clear the values in the head and tail indexes.

18. The following models and firmware versions that support the XCMP and YOUT instructions.

Model	ES2/ EX2	ES2-C	ES2-E	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware Version	V3.60	V3.60	V1.4	V3.00	N/A	V2.02	V2.02	V2.90

Example

Refer to the example in the YOUT instruction (API 316) for more information.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2	SS2	SE	SA2/ SX2
316	YOUT	**S**$_1$, **S**$_2$, **S**$_3$, **D**	Comparing the outputs of multiple work stations				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	YOUT: 9 steps
S$_1$	*											*				
S$_2$													*			
S$_3$													*			
D		*	*													

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SE	SA2/ SX2	ES2/EX2	SS2	SE	SA2/ SX2	ES2/EX2	SS2	SE	SA2/ SX2

Operands:

S$_1$: High-speed counter number

S$_2$: Setting for the number for work stations and objects

S$_3$: First corresponding device for the comparison result in the stack area

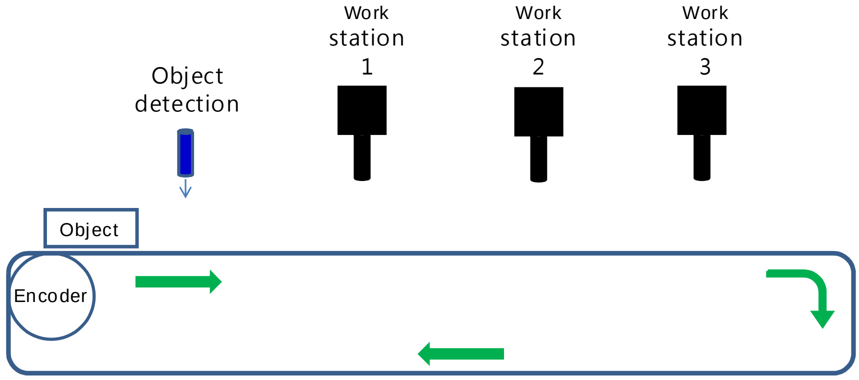
D : First corresponding device for the output work station

Explanation

1. **S**$_1$ is for the setting of the high-speed counter. Use the same settings for the high-speed counter as for the high-speed counter for the XCMP (API 315) instruction.
2. **S**$_2$ occupies two consecutive 16-bit devices. **S**$_2$+0 is n (the work station number) and **S**$_2$+1 is m (the maximum number of objects). The range for n and m is between 1–32. When the value is out of range, the value used is the maximum (32) or the minimum (1). The settings for the operands should be the same as for the XCMP instruction.
3. **S**$_3$ is first corresponding device for the comparison result in the stack area. **S**$_3$ occupies 2xn consecutive 16-bit devices and 2xmxn consecutive 32-bit devices (or 4xmxn consecutive 16-bit devices). For information on the functions of each device and the corresponding number for **D**, refer to the XCMP instruction (API 315). It is suggested that you use the same variable as you use for the XCMP instruction.
4. There is no limit on the number of times you can execute the instruction but only one execution can be done at a time.
5. It is suggested that you use with the XCMP instruction (API 315), and use the same first corresponding device for the comparison result in the stack area (**S**$_3$).
6. **D** is only for the outputs of Y and M devices; Y and M should be the BOOL data type. It occupies a consecutive number of work stations Xn. When used as the output point of Y or the M device, the instruction refreshes the output states.

7. The odd numbered head index values (for example 1, 3, 5,…) are the compared counter results for the object when entering. The even numbered head index values (for example 2, 4, 6,…) are the compared counter result of the object when leaving.
8. When the compared counter result for entering and leaving in the stack area are 0, the actions in this area are not executed and the state of the corresponding output work station is OFF. Add 2 to the value of the tail index and the added value in the tail index should not exceed the value of the head index.
9. When the YOUT instruction is executed, each work station checks the compared value for entering and leaving in the tail index. When the counter value is larger or the same as the compared value for entering, the corresponding output point is ON and adds 1 to the value of the tail index. When the counter value is larger or the same as the compared value for leaving, the corresponding output is OFF and adds 1 to the value of the tail index; but the value of the tail index (after adding 1) does not exceed the value of the head index.

Example: three work stations and up to four objects



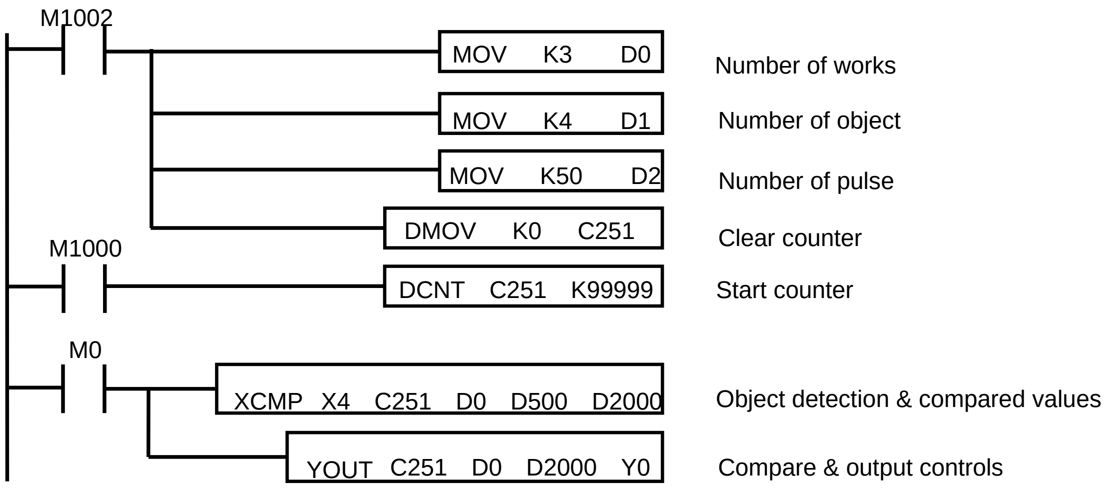
Step 1: use the input point X4 as the object detection interrupt, C251 as the high-speed counter for the encoder and output point Y0 as the first output point for the work station.

Step 2: edit the register to set up the reference values, and the observational error when entering and leaving.

Device D	D500	D502	D504
Reference value for comparison (32-bit)	K2000	K3000	K4000
Device D	D506	D508	D510
Observational error when entering (32-bit)	K100	K120	K130
Device D	D512	D514	D516
Observational error when leaving (32-bit)	K50	K-20	K20
Device D	D2000	D2001	D2002
Value of the head index (16-bit)	K0	K0	K0

Device D	D2003	D2004	D2005
Value of the tail index (16-bit)	K0	K0	K0

Step 3: set up the initial values and write the programs.



After the contact M0 is activated, the system sets the object detection, the compared values, the compared counter result of the object entering and leaving, and the output controls for each work station. For example, the system detects two objects have entered and then four triggers to read the compared counter results: 3000, 3500, 4500, and 5000 in C251 (C251=K5060). The result of the last rising-edge / falling-edge of X4 from C251 for the values K4500 and K5000 are stored in (D3, D4) and (D5, D6) in 32-bit. The following table shows the compared value and the head/tail index in the stack area.

Device D	D2000	D2001	D2002
Value of the head index (16-bit)	K4	K4	K4
Device D number	D2003	D2004	D2005
Value of the tail index (16-bit)	K1	K1	K1
Device D number	D2006	D2008	D2010
Compared counter result 1 of the object when entering (32-bit)	K5100	K6120	K7130
Device D number	D2012	D2014	D2016
Compared counter result 1 of the object when leaving (32-bit)	K5550	K6480	K7520
Device D number	D2018	D2020	D2022

Compared counter result 2 of the object when entering (32-bit)	K6600	K7620	K8630
Device D number	D2024	D2026	D2028
Compared counter result 2 of the object when leaving (32-bit)	K7050	K7980	K9020
Device D number	D2030	D2032	D2034
Compared counter result 3 of the object when entering (32-bit)	K0	K0	K0
Device D number	D2036	D2038	D2040
Compared counter result 3 of the object when leaving (32-bit)	K0	K0	K0

3

The following table shows the state of the output point Y when the high-speed counter C251 reaches 5200.

Output point Y number	Y0	Y1	Y2
16-bit value	ON	OFF	OFF
Device D number	D2000	D2001	D2002
Value of the head index (16-bit)	K4	K4	K4
Device D number	D2003	D2004	D2005
Value of the tail index (16-bit)	K2	K1	K1

The following table shows the state of the output point Y when the high-speed counter C251 reaching 6200.

Output point Y number	Y0	Y1	Y2
16-bit value	OFF	ON	OFF
Device D number	D2000	D2001	D2002
Value of the head index (16-bit)	K4	K4	K4
Device D number	D2003	D2004	D2005
Value of the tail index (16-bit)	K3	K2	K1

The following table shows the state of the output point Y when the high-speed counter C251 reaching 6800.

Output point Y number	Y0	Y1	Y2
16-bit value	ON	OFF	OFF
Device D number	D2000	D2001	D2002
Value of the head index (16-bit)	K4	K4	K4
Device D number	D2003	D2004	D2005
Value of the tail index (16-bit)	K4	K3	K1

The following table shows the state of the output point Y when the high-speed counter C251 reaching 7300.

Output point Y number	Y0	Y1	Y2
16-bit value	OFF	OFF	ON
Device D number	D2000	D2001	D2002
Value of the head index (16-bit)	K4	K4	K4
Device D number	D2003	D2004	D2005
Value of the tail index (16-bit)	K4	K3	K2

The following table shows the state of the output point Y when the high-speed counter C251 reaching 7700.

Output point Y number	Y0	Y1	Y2
16-bit value	OFF	ON	OFF
Device D number	D2000	D2001	D2002
Value of the head index (16-bit)	K4	K4	K4
Device D number	D2003	D2004	D2005
Value of the tail index (16-bit)	K4	K4	K3

The following table shows the state of the output point Y when the high-speed counter C251 reaching 8000.

Output point Y number	Y0	Y1	Y2
Output state	OFF	OFF	OFF
Device D number	D2000	D2001	D2002
Value of the head index (16-bit)	K4	K4	K4
Device D number	D2003	D2004	D2005
Value of the tail index (16-bit)	K4	K4	K3

The following table shows the state of the output point Y when the high-speed counter C251 reaching 8700.

Output point Y number	Y0	Y1	Y2
Output state	OFF	OFF	ON
Device D number	D2000	D2001	D2002
Value of the head index (16-bit)	K4	K4	K4
Device D number	D2003	D2004	D2005
Value of the tail index (16-bit)	K4	K4	K4

3.6.8 External I/O Display

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
70	TKY	DTKY	-	10-key input	✓	✓	✓	✓	7	13
71	HKY	DHKY	-	Hexadecimal key input	✓	✓	✓	✓	9	17
72	DSW	-	-	DIP Switch	✓	✓	✓	✓	9	-
73	SEGD	-	✓	7-segment decoder	✓	✓	✓	✓	5	-
74	SEGL	-	-	7-segment with latch	✓	✓	✓	✓	7	-
75	ARWS	-	-	Arrow switch	✓	✓	✓	✓	9	-
76	ASC	-	-	ASCII code conversion	✓	✓	✓	✓	11	-
77	PR	-	-	Print (ASCII code output)	✓	✓	✓	✓	5	-

API	Mnemonic		Operands	Function	Controllers			
70	D	TKY	(S) (D1) (D2)	Ten key input	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S	*	*	*	*												TKY: 7 steps
D_1								*	*	*	*	*	*	*	*	DTKY: 13 steps
D_2		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

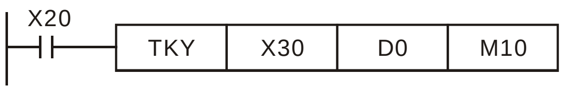
S: Start device for key input (occupies 10 consecutive devices) **D_1**: Device for storing keyed-in value **D_2**: Output signal (occupies 11 consecutive devices)

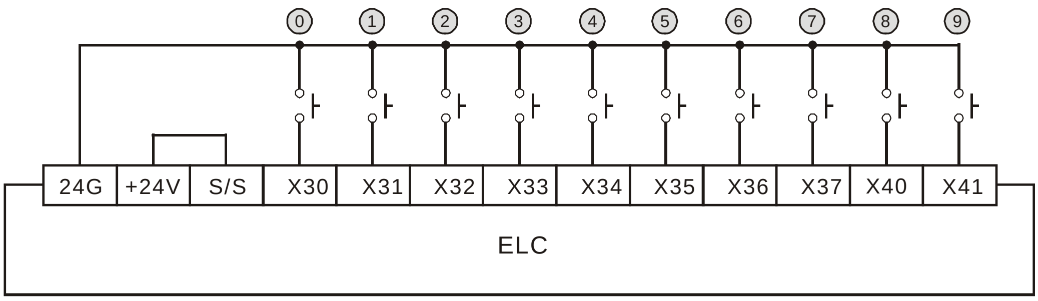
Explanations:

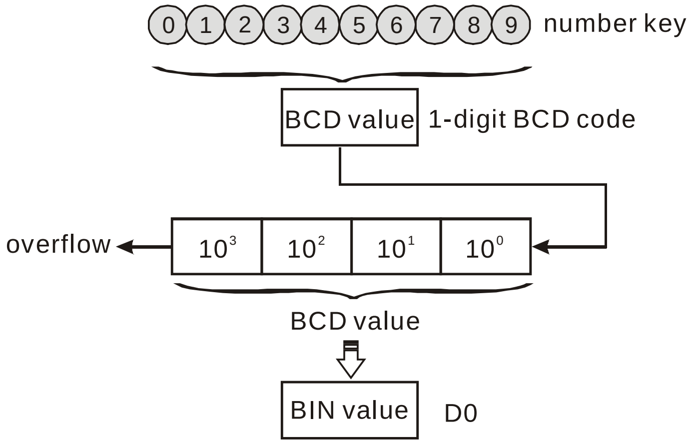
1. This instruction designates 10 external input points (corresponding to decimal numbers 0 ~ 9) starting from **S**, connecting to 10 keys respectively. Input point started from **S** triggers associated device in **D_2** and **D_2** maps to a decimal value, a 4-digit decimal value 0~9,999 (16-bit instruction) or an 8-digit value 0~99,999,999 (32-bit instruction). The decimal value is stored in **D_1**.
2. There is no limitation on the times of using this instruction in the program, however only one instruction is allowed to be executed at the same time.

Program Example:

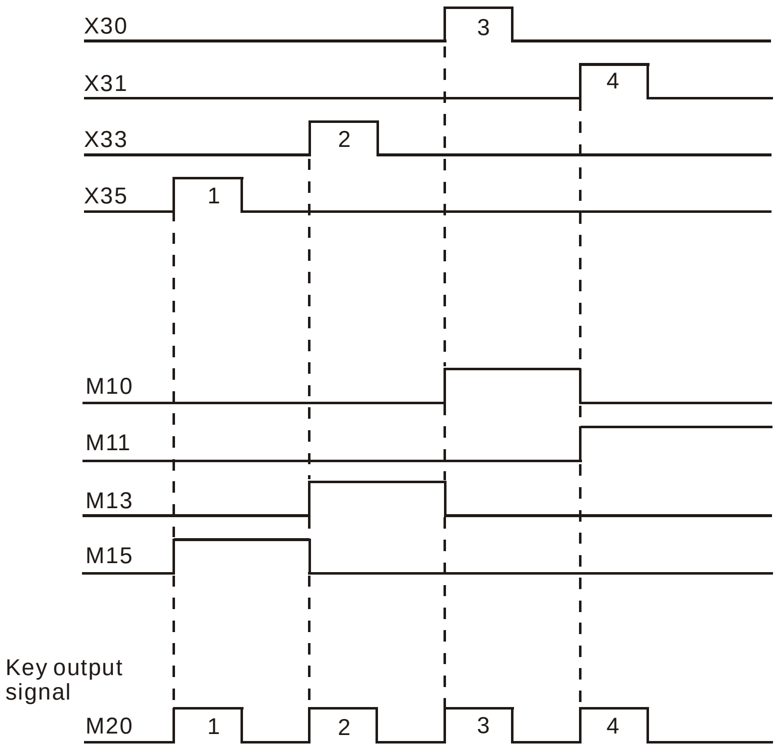
1. Connect the 10 input points starting from X30 to the 10 keys (0 ~ 9). When X20 = ON, the instruction will be executed and the key-in values will be stored in D0 in BIN form. The key status will be stored in M10 ~ M19.







2. As shown in the timing diagram below, four keys connected with X35, X33, X31 and X30 are pressed in order. Therefore, the number 5,301 is generated and stored in D0. 9,999 is the maximum value allowed for D0. If the entered number exceeds the available range, the highest digit performs overflow.
3. When X35 is pressed, M15 remains ON until another key is pressed and the rule applies to other inputs.
4. M20 = ON when any of the keys is pressed.
5. When X20 is OFF, the value in D0 remains unchanged but M10~M20 will be OFF.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
71	D	HKY	S D1 D2 D3	Hexadecimal key input				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	HKY: 9 steps DHKY: 17 steps
S	*															
D_1		*														
D_2											*	*	*	*	*	
D_3		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

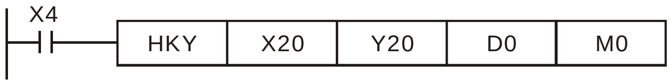
S: The start of input devices (occupies 4 consecutive devices) **D_1**: The start of output devices (occupies 4 consecutive devices) **D_2**: Device for storing key input value **D_3**: Key input status (occupies 8 consecutive devices)

Explanations:

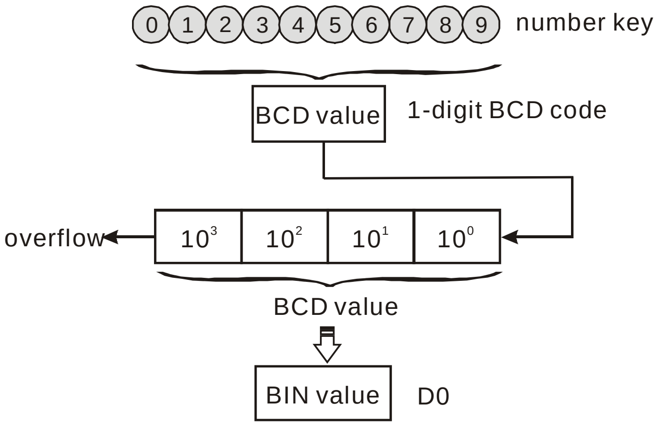
1. This instruction creates a 16-key keyboard by a multiplex of 4 consecutive external input devices from **S** and 4 consecutive external output devices from **D_1**. By matrix scan, the key input value will be stored in **D_2**. **D_3** stores the condition of keys A~F and indicates the key input status of both 0~9 and A~F..
2. M1029 = ON for a scan cycle every time when a key is pressed.
3. If several keys are pressed, only the first pressed key is valid.
4. **D_2** maps to a decimal value, a 4-digit decimal value 0~9,999 (16-bit instruction) or an 8-digit value 0~99,999,999 (32-bit instruction). If the entered number exceeds the available range, i.e. 4 digit in 16-bit and 8 digits in 32-bit instruction, the highest digit performs overflow
5. There is no limitation on the times of using this instruction in the program, but only one instruction is allowed to be executed in the same scan time.

Program Example:

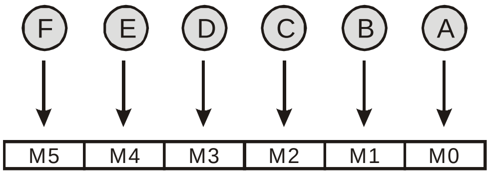
1. Designate 4 input points X20 ~ X23 and the other 4 output points Y20 ~ Y23 to construct a 16-key keyboard. When X4 = ON, the instruction will be executed and the keyed-in value will be stored in D0 in BIN form. The key status will be stored in M10 ~ M19.



2. Input keys 0~9:

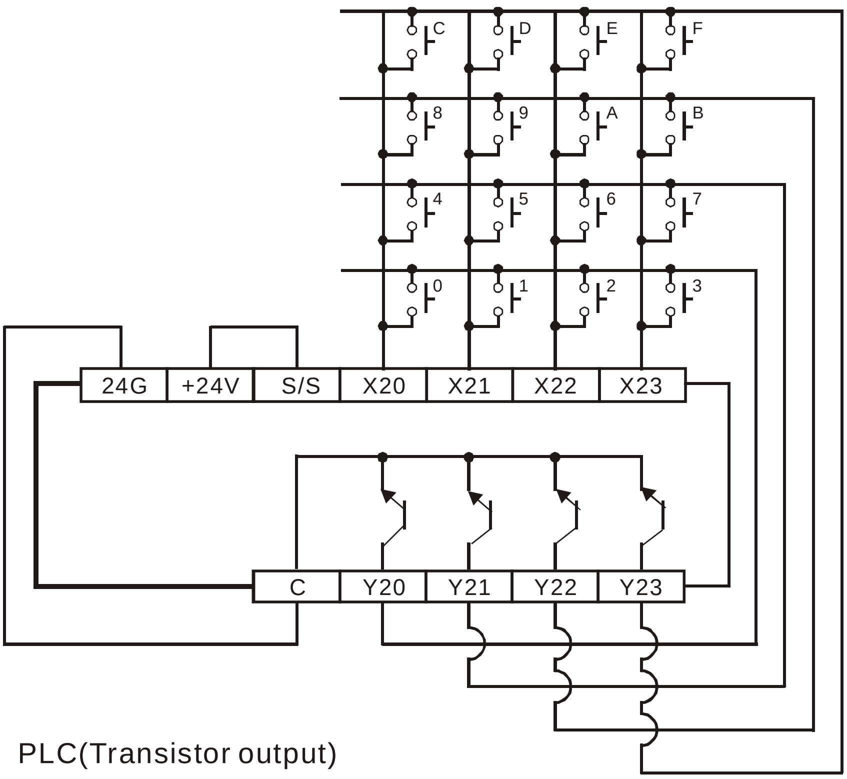


3. Input keys A~F:
   a) When A is pressed, M0 will be ON and retained. When D is pressed next, M0 will be OFF, M3 will be ON and retained..
   b) If two or more keys are pressed at the same time, only the key activated first is effective.



4. Key input status:
   a) When any key of A ~ F is pressed, M6 = ON for one scan time.
   b) When any key of 0 ~ 9 is pressed, M7 = ON for one scan time.
5. When the drive contact X4 = OFF, the value d in D0 remains unchanged but M0~M7 = OFF.

6. External wiring:



Points to note:

1. When HKY instruction is executed, 8 scan cycles (matrix scan) are required for reading the input value successfully. A scan cycle that is too long or too short may cause the input to be read incorrectly. In this case we suggest the following solutions:

   If the scan cycle is too short, I/O may not be able to respond in time, resulting in incorrect input values. To solve this problem please fix the scan time.

   If the scan period is too long, the key may respond slowly. In this case, write this instruction into the time-interrupt subroutine to fix the execution time for this instruction.

2. The function of flag M1167:

   When M1167 = ON, HKY instruction can input hexadecimal value consists of 0~F.

   When M1167 = OFF, A~F of HKY instruction are used as function keys.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
72	DSW	S D1 D2 n	DIP Switch				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											DSW: 9 steps
S	*															
D_1		*														
D_2											*	*	*			
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

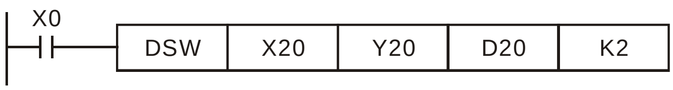
S: The Start of input devices **D_1**: The Start of output devices **D_2**: Device for storing switch input value **n**: Groups of switches (**n** = 1~2)

Explanations:

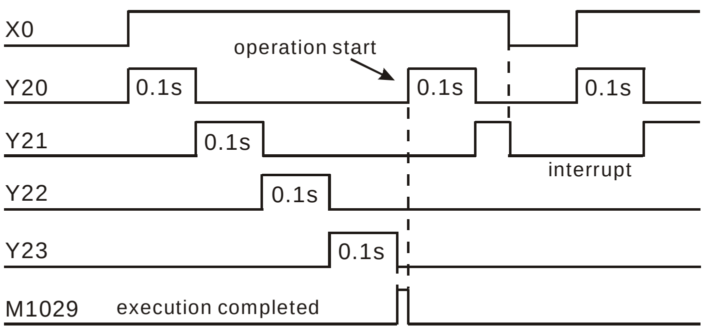
1. This instruction creates 1(2) group of 4-digit DIP switch by the combination of 4(8) consecutive input points starting from **S** and 4 consecutive output points starting from **D_1**. The set value will be read in **D_2** and the value in **n** specifies the number of groups (1~2) of the DIP switch.
2. **n** = K1, **D_2** occupies 1 register. **n** = K2, **D_2** occupies 2 consecutive registers.
3. There is no limitation on the times of using this instruction in the program, however only one instruction is allowed to be executed at the same scan time.

Program Example:

1. The first group of DIP switches consists of X20 ~ X23 and Y20 ~ Y23. The second group of switches consists of X24 ~ X27 and Y20 ~ Y23. When X10 = ON, the instruction will be executed and the set value of the first switch will be read and converted into BIN value then stored in D20. BIN value of 2nd switch will be stored in D21.

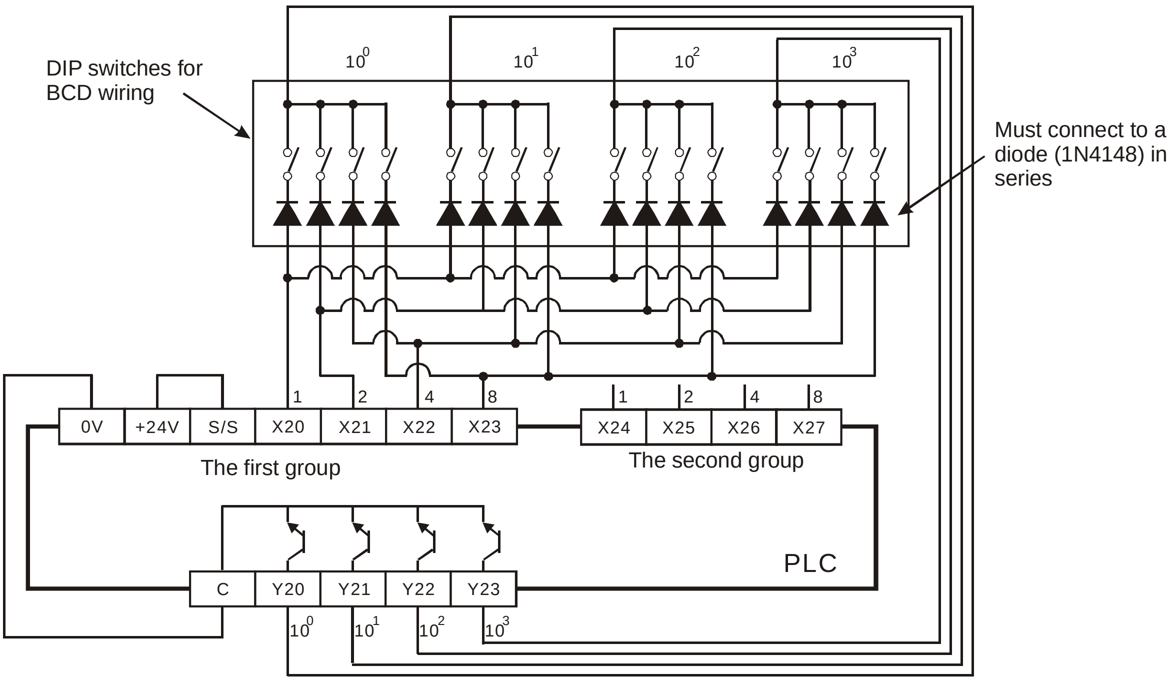


2. When X0 = ON, Y20~Y23 are scanned repeatedly. M1029 = ON for a scan time when a scan cycle from Y20 to Y23 is completed.



3. Please use transistor output for Y20 ~ Y23. Every pin 1, 2, 4, 8 shall be connected to a diode (0.1A/50V) in series before connecting to the input terminals on PLC.

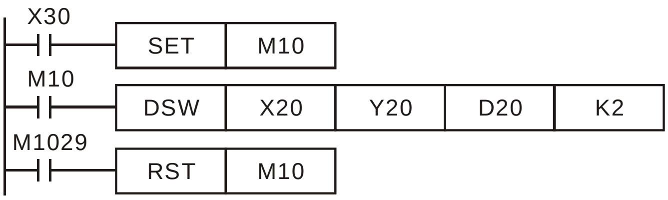
Wiring diagram of DIP switch:



Points to note:

When the terminals to be scanned are relay outputs, the following program methods can be applied:

1. When X30 = ON, DSW instruction will be executed. When X30 goes OFF, M10 remains ON until the current scan cycle of output terminals is completed..
2. If the drive contact X30 uses button switch, M10 turns off only when the current scan cycle on outputs is completed, so that a correct value from DIP switch can be read. In addition, the continuous scan cycle on outputs will be performed only when the drive contact is pressed and held. Applying this method can reduce the driving frequency of relay outputs so as to extend to life-span of relays.



API	Mnemonic		Operands	Function	Controllers
73	SEGD	P	(S) (D)	7-segment decoder	ES2/EX2/EC5, SS2, SA2 SE, SX2

Type / OP	Bit Devices X	Y	M	S	Word devices K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S					*	*	*	*	*	*	*	*	*	*	*	SEGD, SEGDP: 5 steps
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

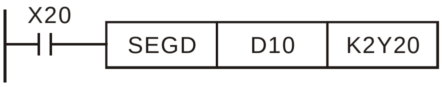
S: Source device for decoding **D**: Output device after decoding

Explanations:

The instruction decodes the lower 4 bits (Hex data: 0 to 9, A to F) of source device **S** and stores the decoded data in lower 8 bits of **D** so as to form a 7-segment display.

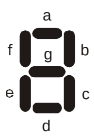
3

Program Example:



When X20 = ON, the content of the lower 4 bits (b0~b3) of D10 will be decoded into the 7-segment display. . The decoded results will be stored in Y20~Y27. If the source data exceeds 4bits, still only lower 4 bits will be decoded.

Decoding table of the 7-segment display:

Hex	Bit combi-nation	Composition of the 7-segment display	Status of each segment B0(a)	B1(b)	B2(c)	B3(d)	B4(e)	B5(f)	B6(g)	Data displayed
0	0000		ON	ON	ON	ON	ON	ON	OFF	
1	0001		OFF	ON	ON	OFF	OFF	OFF	OFF	
2	0010		ON	ON	OFF	ON	ON	OFF	ON	
3	0011		ON	ON	ON	ON	OFF	OFF	ON	
4	0100		OFF	ON	ON	OFF	OFF	ON	ON	
5	0101		ON	OFF	ON	ON	OFF	ON	ON	
6	0110		ON	OFF	ON	ON	ON	ON	ON	
7	0111		ON	ON	ON	OFF	OFF	ON	OFF	
8	1000		ON	ON	ON	ON	ON	ON	ON	
9	1001		ON	ON	ON	ON	OFF	ON	ON	
A	1010		ON	ON	ON	OFF	ON	ON	ON	
B	1011		OFF	OFF	ON	ON	ON	ON	ON	
C	1100		ON	OFF	OFF	ON	ON	ON	OFF	
D	1101		OFF	ON	ON	ON	ON	OFF	ON	
E	1110		ON	OFF	OFF	ON	ON	ON	ON	
F	1111		ON	OFF	OFF	OFF	ON	ON	ON	

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
74	SEGL	S D n	7-segment with Latch				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SEGL: 7 steps
S					*	*	*	*	*	*	*	*	*	*	*	
D		*														
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device storing the value to be displayed in 7-segment display **D**: Output device for 7-segment display

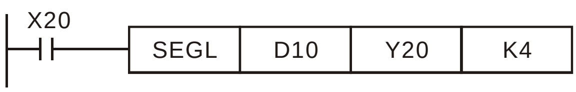
n: Configuration setting of output signal (**n** = 0~7)

Explanations:

1. This instruction occupies 8 or 12 consecutive external output points starting from **D** for displaying the data of 1 or 2 sets of 4-digit 7-segment display. Every digit of the 7-segment display carries a “Drive” which converts the BCD codes into 7-segment display signal. The drive also carries latch control signals to retain the display data of 7-segment display.
2. **n** specifies the number of sets of 7-segment display (1 set or 2 sets), and designates the positive / negative output of PLC and the 7-segment display.
3. When there is 1 set of 4-digit output, 8 output points will be occupied. When there are 2 sets of 4-digit output, 12 output points will be occupied
4. When the instruction is executed, the output terminals will be scanned circularly. When the drive contact goes from OFF to ON again during the execution of instruction, the scan will restart from the beginning of the output terminals.
5. Flag: When SEGL is completed, M1029 = ON for one scan cycle.
6. There is no limitation on the times of using this instruction in the program, however only one instruction is allowed to be executed at a time.

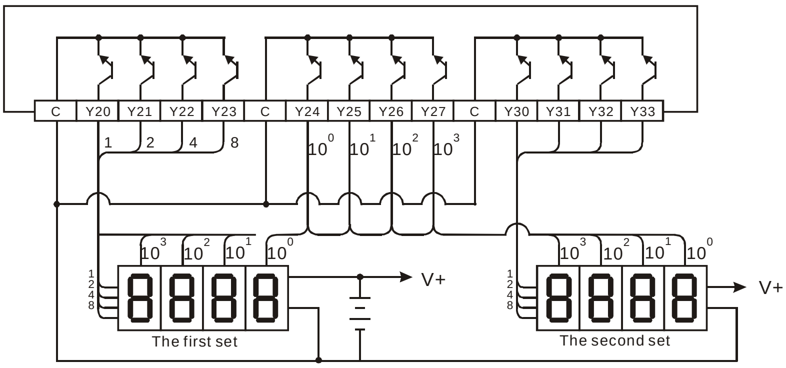
Program Example:

1. When X20 = ON, SEGL instruction executes and Y24~Y27 forms an output scan loop for 7-segment display. The value of D10 will be mapped to Y20~Y23, converted to BCD code and sent to the 1st set of 7-segment display. The value of D11 will be mapped to Y30~Y33, converted to BCD code and sent to the 2nd set of 7-segment display. If the values in D10 and D11 exceed 9,999, operational error will occur.



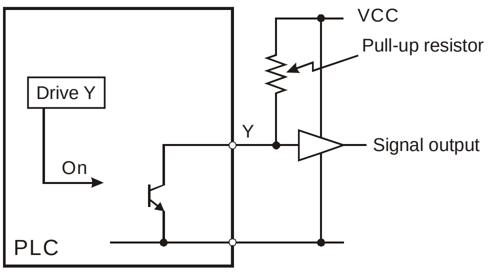
2. When X20 = ON, Y24~Y27 will be scanned in circles automatically. Each circle requires 12 scan cycles. M1029 = ON for a scan cycle whenever a circle is completed.
3. When there is 1 set of 4-digit 7-segment display, **n** = 0 ~ 3
   a) Connect the 7-segment display terminals 1, 2, 4, 8 in parallel then connect them to Y20 ~ Y23 on PLC. After this, connect the latch terminals of each digit to Y24 ~ Y27 on PLC.
   b) When X20 = ON, the content of D10 will be decoded through Y20 ~ Y23 and sent to 7-segment display in sequence by the circulation of Y24 ~ Y27
4. When there are 2 sets of 4-digit 7-segment display, **n** = 4 ~ 7
   a) Connect the 7-segment display terminals 1, 2, 4, 8 in parallel then connect them to Y30 ~ Y33 on PLC. After this, connect the latch terminals of each digit to Y24 ~ Y27 on PLC.
   b) The content in D10 is sent to the 1st set of 7-segment display. The content in D11 is sent to the 2nd set of 7-segment display. If D10 = K1234 and D11 = K4321, the 1st set will display 1 2 3 4, and the 2nd set will display 4 3 2 1.

Wiring of the 7-segment display scan output:



Points to note:

1. For executing this instruction, scan time must be longer than 10ms. If scan time is shorter than 10ms, please fix the scan time at 10ms.
2. If the output points of PLC is transistor output, please apply proper 7-segment display.
3. Operand **n** is used for setting up the polarity of the transistor output and the number of sets of the 4-digit 7-segment display.
4. The output point must be a transistor module of NPN output type with open collector outputs. The output has to connect to a pull-up resistor to VCC (less than 30VDC). When wiring, output should connect a pull-high resistor to VCC (less than 30VDC). Therefore, when output point Y is ON, the output signal will be LOW.



5. Positive logic (negative polarity) output of BCD code

BCD value				Y output (BCD code)				Signal output			
b_3	b_2	b_1	b_0	8	4	2	1	A	B	C	D
0	0	0	0	0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	1	1	1	1	0
0	0	1	0	0	0	1	0	1	1	0	1
0	0	1	1	0	0	1	1	1	1	0	0
0	1	0	0	0	1	0	0	1	0	1	1
0	1	0	1	0	1	0	1	1	0	1	0
0	1	1	0	0	1	1	0	1	0	0	1
0	1	1	1	0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	0	0	1	1	1
1	0	0	1	1	0	0	1	0	1	1	0

6. Negative logic (Positive polarity) output of BCD code

BCD value				Y output (BCD code)				Signal output			
b_3	b_2	b_1	b_0	8	4	2	1	A	B	C	D
0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	0	0	0	0	1
0	0	1	0	1	1	0	1	0	0	1	0
0	0	1	1	1	1	0	0	0	0	1	1
0	1	0	0	1	0	1	1	0	1	0	0
0	1	0	1	1	0	1	0	0	1	0	1
0	1	1	0	1	0	0	1	0	1	1	0
0	1	1	1	1	0	0	0	0	1	1	1
1	0	0	0	0	1	1	1	1	0	0	0
1	0	0	1	0	1	1	0	1	0	0	1

7. Operation logic of output signal

Positive logic (negative polarity)		Negative logic (positive polarity)	
Drive signal (latch)	Data control signal	Drive signal (latch)	Data control signal
1	0	0	1

8. Parameter **n** settings:

Sets of 7-segment display	1 set				2 sets			
BCD code data control signal	＋		－		＋		－	
Drive (latch) signal	＋	－	＋	－	＋	－	＋	－
n	0	1	2	3	4	5	6	7

'＋': Positive logic (Negative polarity) output

'－': Negative logic (Positive polarity) output

9. The polarity of PLC transistor output and the polarity of the 7-segment display input can be designated by the setting of **n**.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
75	ARWS	(S) (D1) (D2) (n)	Arrow switch				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S	*	*	*	*												ARWS: 9 steps
D_1											*	*	*	*	*	
D_2		*														
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

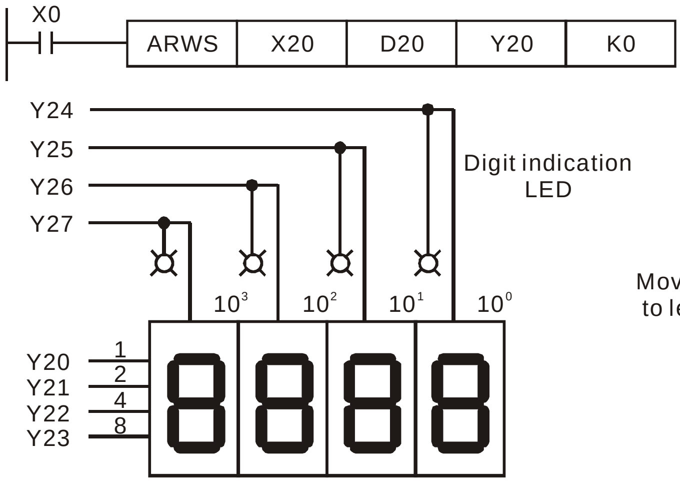
S: Start device for key input (occupies 4 consecutive devices) **D_1**: Device storing the value to be displayed in 7-segment display **D_2**: Output device for 7-segment display **n**: Configuration setting of output signal (**n** = 0~3). Please refer to explanations of SEGL instruction for the **n** usage.

Explanations:

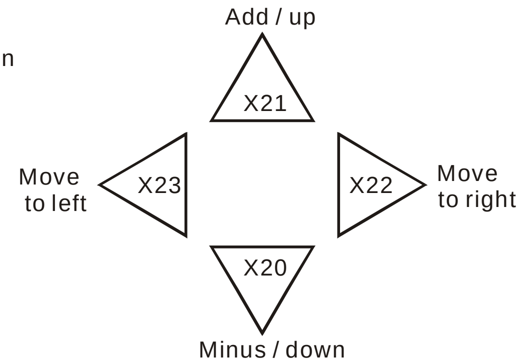
1. ARWS instruction displays the value set in device **D_1** on a set of 4-digit 7 segment display. PLC automatically converts the decimal value in **D_1** to BCD format for displaying on the 7 segment display. Each digit of the display can be modified by changing the value in **D_1** through the operation of the arrow switch.
2. Number of **D_2** only can be specified as a multiple of 10, e.g. Y0, Y10, Y20…etc.
3. Output points designated by this instruction should be transistor output.
4. When using this instruction, please fix the scan time, or place this instruction in the timer interruption subroutine (I610/I699, I710/I799).
5. There is no limitation on the times of using this instruction in the program, but only one instruction is allowed to be executed at a time.

Program Example:

1. When the instruction is executed, X20 is defined as the Minus key, X21 is defined as the Add key, X22 is defined as the Right key and X23 is defined as the Left key. The keys are used to modify the set values (range: 0 ~ 9,999) stored in D20..
2. When X0 = ON, digit 103 will be the valid digit for setup. When Left key is pressed, the valid digit will shift as the following sequence: 103→100→101→102→103→100.
3. When Right key is pressed, the valid digit will shift as the following sequence: 103→102→101→100→103→102. Besides, the digit indicators (LED, Y24 to Y27) will be ON for indicating the position of the valid digit during shift operation.
4. When Add key is pressed, the content in the valid digit will change as 0 → 1 → 2 … → 8 → 9 → 0 →1. When Minus key is pressed, the content in the valid digit will change as 0 → 9 → 8 … → 1 → 0 → 9. The changed value will also be displayed in the 7-segment display.



7-segment display for the 4-digit set value



The 4 switches are used for moving the digits and modifying set values.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
76	ASC	(S) (D)	ASCII code conversion				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ASC: 11 steps
S																
D											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

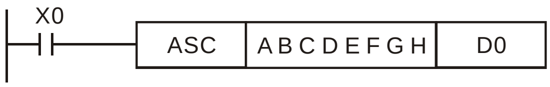
S: English letters to be converted into ASCII code (A~Z or a~z only)   **D**: Device for storing ASCII code

Explanation:

1. The ASC instruction converts 8 English letters stored in **S** and save the converted ASCII code in **D**. The value in **S** can be input by WPLSoft or ISPSoft.
2. If PLC is connected to a 7-segment display while executing ASC instruction, the error message can be displayed by English letters
3. Flag: M1161 (8/16 bit mode switch)

Program Example:

When X0 = ON, A~H is converted to ASCII code and stored in D0~D3.



	b15	b0
D0	42H (B)	41H (A)
D1	44H (D)	43H (C)
D2	46H (F)	45H (E)
D3	48H (H)	47H (G)
	High byte	Low byte

When M1161 = ON, every ASCII code converted from the letters will occupy the lower 8 bits (b7 ~ b0) of a register and the upper 8 bits are invalid (filled by 0), i.e. one register stores a letter

	b15	b0
D0	00 H	41H (A)
D1	00 H	42H (B)
D2	00 H	43H (C)
D3	00 H	44H (D)
D4	00 H	45H (E)
D5	00 H	46H (F)
D6	00 H	47H (G)
D7	00 H	48H (H)
	High byte	Low byte

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
77	PR	S D	Print (ASCII Code Output)				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	PR: 5 steps
S											*	*	*			
D		*														

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

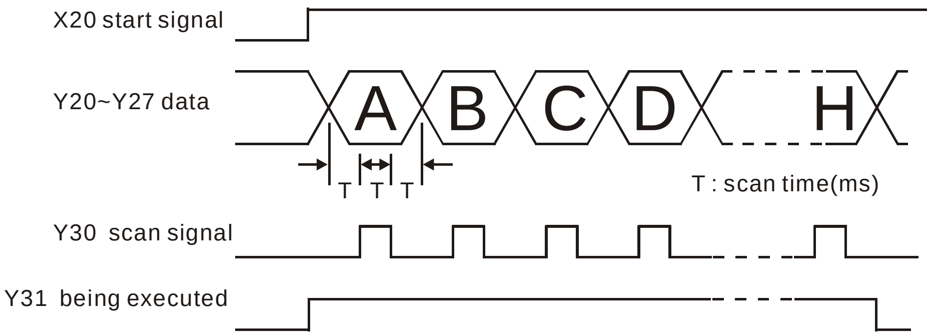
S: Device for storing ASCII code (occupies 4 consecutive devices) **D**: External ASCII code output points (occupies 10 consecutive devices)

Explanations:

1. This instruction will output the ASCII codes in the 4 registers starting from **S** through output points started from **D.**
2. $\mathbf{D_0}$ ~ $\mathbf{D_7}$ map to source data (ASCII code) directly in order, $\mathbf{D_{10}}$ is the scan signal and $\mathbf{D_{11}}$ is the execution flag.
3. This instruction can only be used twice in the program.
4. Flags: M1029 (PR execution completed); M1027 (PR output mode selection).

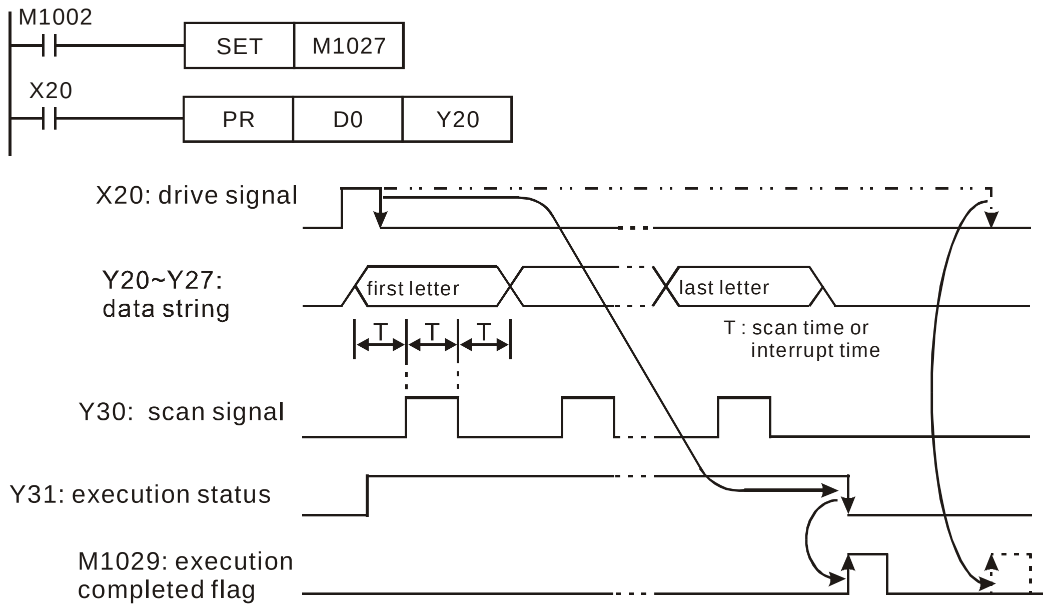
Program Example 1:

1. Use API 76 ASC to convert A ~ H into ASCII codes and store them in D0 ~ D3. After this, use this instruction to output the codes in sequence.
2. When M1027 = OFF and X20 = ON, the instruction will designate Y20 (lowest bit) ~ Y27 (highest bit) as the output points and Y30 as scan signals, Y31 as execution flag. In this mode, users can execute an output for 8 letters in sequence..
3. If X20 turns from ON → OFF during the execution of the instruction, the data output will be interrupted, and all the output points will be OFF. When X20 = ON again, the data output will start from the first letter again.



Program Example 2:

1. PR instruction supports ASCII data output of 8-bit data string when M1027 = OFF. When M1027 = ON, the PR instruction is able to execute an output of 1~16 bit data string.
2. When M1027 = ON and X20 = ON, this instruction will designate Y20 (lowest bit) ~ Y27 (highest bit) as the output points and Y30 as scan signals, Y31 as execution flag. In this mode, users can execute an output for 16 letters in sequence. In addition, if the drive contact X20 is OFF during execution, the data output will stop until a full data string is completed.
3. The data 00H (NULL) in a data string indicates the end of the string and the letters coming after will not be processed.
4. If the drive contact X20 is OFF during execution, the data output will stop until a full data string is completed. However, if X20 remains ON, execution completed flag M1029 will not be active as the timing diagram below.



Points to note:

1. Please use transistor output for the output points designated by this instruction.
2. When using this instruction, please fix the scan time or place this instruction in a timer interrupt subroutine.

3.6.9 Serial I/O

API	Mnemonic		PULSE	Function	Applicable to					STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2	SX2	SE	16-bit	32-bit
78	FROM	DFROM	✓	Read CR data from special modules	ES2 EX2	✓	✓	✓	✓	9	17
79	TO	DTO	✓	Write CR data into special modules	ES2 EX2	✓	✓	✓	✓	9	17
80	RS	-	-	Serial communication	✓	✓	✓	✓	✓	9	-
81	PRUN	DPRUN	✓	Parallel run	ES2 EX2	✓	✓	✓	✓	5	9
82	ASCII	-	✓	Convert HEX to ASCII	✓	✓	✓	✓	✓	7	-
83	HEX	-	✓	Convert ASCII to HEX	✓	✓	✓	✓	✓	7	-
84	CCD	-	✓	Check code	✓	✓	✓	✓	✓	7	-
85	VRRD	-	✓	Volume read	-	-	✓	✓	-	5	-
86	VRSC	-	✓	Volume scale read	-	-	✓	✓	-	5	-
87	ABS	DABS	✓	Absolute value	✓	✓	✓	✓	✓	3	5
88	PID	DPID	-	PID control	✓	✓	✓	✓	✓	9	17

API	Mnemonic			Operands	Function	Controllers
78	D	FROM	P	m1 m2 D n	Read CR data from Special Modules	ES2/EX2 SS2 SA2 SE SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F
m_1					*	*							*		
m_2					*	*							*		
D													*		
n					*	*							*		

Program Steps: FROM, FROMP: 9 steps; DFROM, DFROMP: 17 steps

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

m_1: No. of special module **m_2**: CR# in special module to be read **D**: Device for storing read data **n**: Number of data to be read at a time

Explanations:

1. PLC uses this instruction to read CR (Control register) data from special modules and built-in network communication ports.
2. Operand ranges for **m1**, **m2**, and **n**:

ES2/EX2/ES2-C/ES2-E:

Operand	**m1**	**m2**	**n** in the 16-bit instruction	**n** in the 32-bit instruction
Right-side module	0~7	0~255	1~4, 1~6 (V3.0 and above)	1~2, 1~3 (V3.0 and above)
Built-in network communication port (for ES2-E only)	108	0~255	1~(256-m2)	NOT recommended

SA2/SX2:

Operand	**m1**	**m2**	**n** in the 16-bit instruction	**n** in the 32-bit instruction
Right-side module	0~7	0~48	1~6*	1~3*
Left-side module (for 12SA2/20SX2)	100~107	0~255	1~(256-m2)	1~(256-m2)/2

* The maximum number of values which can be read by SA2 V2.6/SX2 V2.4 (below) is 4 (16-bit instruction/2 (32-bit instruction).

SE:

Operand	**m1**	**m2**	**n** in the 16-bit instruction	**n** in the 32-bit instruction
Right-side module	0~7	0~48	1~4, 1~6 (V1.4 and above)	1~2, 1~3 (V1.4 and above)
Left-side module (for 12SE only)	100~108	0~255	1~(256-m2)	1~(256-m2)/2 **m_1**=108* NOT recommended

Built-in network communication port (for 26SE only)	108	0~255	1~(256-m2)	NOT recommended

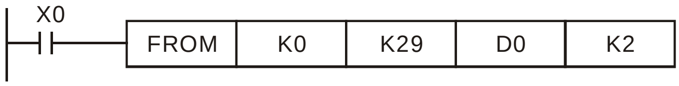
*** m**$_1$=108 indicates the parameter of the PLC CPU built-in network communication port. Refer to Appendix B.2 for more information on CR.

SS2:

Operand	**m1**	**m2**	**n** in the 16-bit instruction	**n** in the 32-bit instruction
Right-side module	0~7	0~48	1~4, 1~6 (V2.8 and above)	1~2, 1~3 (V2.8 and above)
Left-side module	Left-side modules are not supported.			

Program Example:

1. Read out the data in CR#29 of special module N0.0 to register D0 in PLC, and CR#30 of special module No.0 to register D1 in PLC. 2 consecutive 16-bit data are read at one time (**n** = 2).
2. When X0 = ON, the instruction executes; when X0 = OFF, the previous content in D0 and D1 won't be changed.



API	Mnemonic			Operands	Function	Controllers
79	D	TO	P	m1 m2 S n	Write CR data into Special Modules	ES2/EX2, SS2, SA2 SE, SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
m_1					*	*							*			TO, TOP: 9 steps
m_2					*	*							*			DTO, DTOP: 17 steps
S					*	*							*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

m_1: No. of special module **m_2**: CR# in special module to be written **S**: Data to be written in CR

n: Number of data to be written at a time

Explanations:

1. PLC uses this instruction to write data into CR (Control register) on special modules.
2. Setting range of **m_1**, **m_2**, **n**: refer to the explanation from FROM instruction for more information.

Program Example:

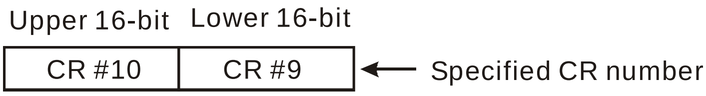
1. Use 32-bit instruction DTO to write the content in D11 and D10 into CR#13 and CR#12 of special module No.0. One 32-bit data is written at a time (**n** = 1)
2. When X0 = ON, the instruction executes; when X0 = OFF, the previous content in D10 and D11 won't be changed.

X0	DTO	K0	K12	D10	K1

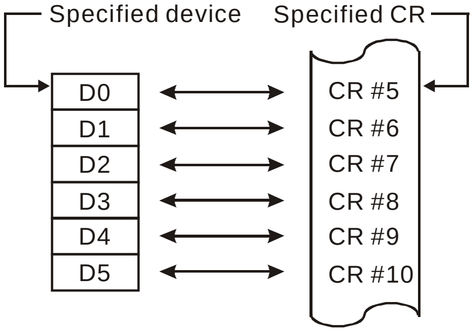
The rules for operand:

1. **m_1**: number of special module. The modules are numbered from 0 (closest to MPU) to 7 automatically by their distance from MPU. Maximum 8 modules are allowed to connect to MPU and will not occupy any digital I/O points
2. **m_2**: number of CR (Control Register). CR is the 16-bit memory built in the special module for control or monitor purpose, numbering in decimal. All operation status and settings of the special module are recorded in the CR.

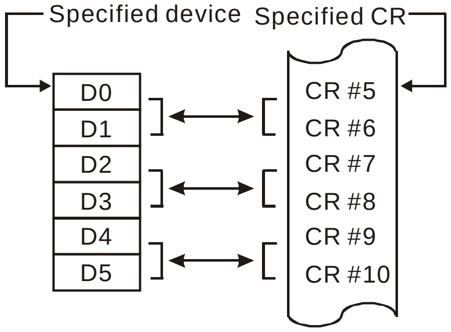
3. FROM/TO instruction reads/writes 1 CR at a time. DFROM/DTO instruction reads/writes 2 CRs at a time.



4. **n**: Number of data to be written at a time. **n** = 2 in 16-bit instruction has the same operation results as **n** = 1 in 32-bit instruction.



16-bit instruction when n=6



32-bit instruction when n=3

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
80	RS	S m D n	Serial Communication				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											RS: 9 steps
S													*			
m					*	*							*			
D													*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Start device for data to be sent **m**: Length of data to be sent (**m** = 0~255) **D**: Start device for data to be received **n**: Length of data to be received (**n** = 0~255)

Explanations:

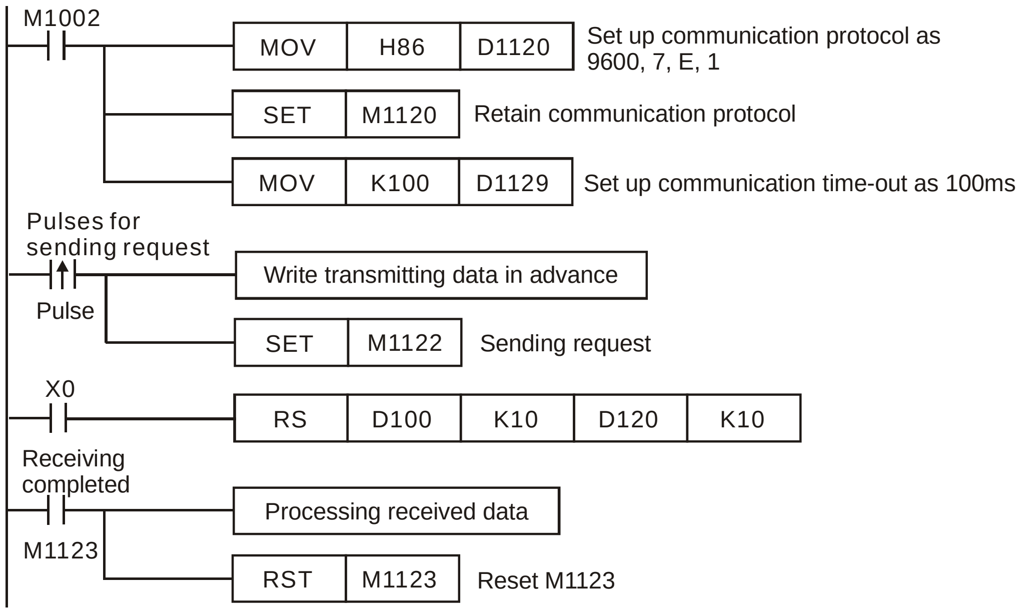
1. RS instruction is used for data transmitting and receiving between PLC and external/peripheral equipment (AC motor drive, etc.). Users have to pre-store word data in registers starting from **S**, set up data length **m**, specify the data receiving register **D** and the receiving data length **n**. If S and S are modified by an E device or an F device, the setting value of the E device or the F device can no be changed when the instruction is executed, otherwise a reading error or w writing error will occur.
2. RS instruction supports communication on COM1 (RS-232), COM2 (RS-485) and COM3 (RS-485). COM3 is only applicable to DVP-ES2/EX2/12SA2/12SE, and is not applicable to DVP-ES2-C.
3. Designate **m** as K0 if data sending is not required. Designate **n** as K0 if data receiving is not required.
4. Modifying the communication data during the execution of RS instruction is invalid.
5. There is no limitation on times of using this instruction, however, only 1 instruction can be executed on one communication port at the same time..
6. If a peripheral device is equipped with RS-485 communication, and the communication format of the device is open, the PLC and the device can transmit data by means of the instruction RS.
7. If the communication format of the peripheral device is Modbus, DVP series PLC offers handy communication instructions MODRD, MODWR, and MODRW, to work with the device.
8. If a Delta VFD series AC motor drive is used, the PLC provides the convenience instructions API 102 FWD, API 103 REV, API 104 STOP, API 105 RDST, and API 106 RSTEF. If a Delta ASD series servo drive is used, the PLC provides the convenience instruction API 206 ASDRW. If a Delta DMV series product is used, the PLC provides the convenience instruction API 295 DMVRW.

9. Please refer to the points to note below for more information about the flags and the special data registers which are related to RS-485 communication instructions.
10. Communication instructions including MODRW/MODWR/MODRD/RS should be used with flags including M1312/M1122/M1316. Set the flags first and then execute the communication instruction.
11. Executing RS instruction with the newly added M1263, you can set the data receiving as completed when the stop between data receiving is longer than what is set in D1168, only available for COM2. Refer to Program Example 7 below for more details. The supporting models are shown below:

Models	ES2/EX2	ES2-C	ES2-E	12SA2	SX2	SS2	12SE	26SE	28SA2
FW Version	V4.02	V3.8	V1.48	V3.04	V3.04	V4.02	V2.06	V2.06	V2.90

Program Example 1: COM2 RS-485

1. Write the data to be transmitted in advance into registers starting from D100 and set M1122 (Sending request) as ON.
2. When X10 = ON, RS instruction executes and PLC is ready for communication. D100 will then start to send out 10 data continuously. When data sending is over, M1122 will be automatically reset. (DO NOT apply RST M1122 in program). After approximate 1ms, PLC will start to receive 10 data and store the data in 10 consecutive registers starting from D120.
3. When data receiving is completed, M1123 will automatically be ON. When data processing on the received data is completed, M1123 has to be reset (OFF) and the PLC will be ready for communication again. However, DO NOT continuously execute RST M1123, i.e. it is suggested to connect the RST M1123 instruction after the drive contact M1123.

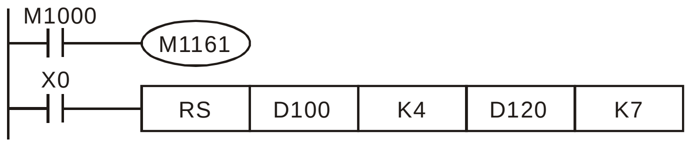


Program Example 2: COM2 RS-485

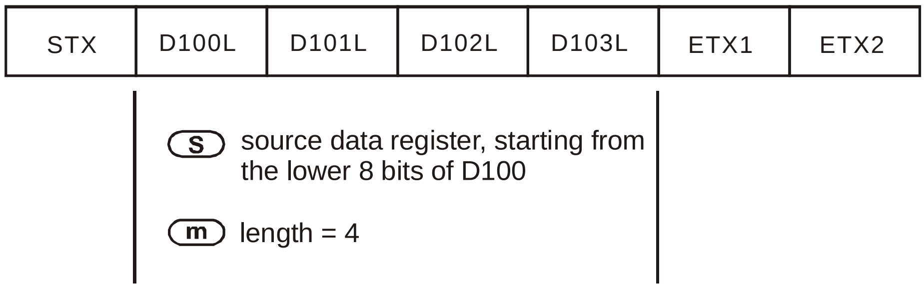
Switching between 8-bit mode (M1161 = ON) and 16-bit mode (M1161 = OFF)

8-bit mode:

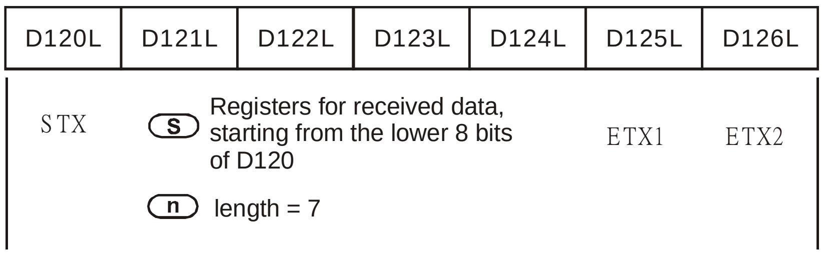
1. STX (Start of Text) and ETX (End of text) are set up by M1126 and M1130 together with D1124~D1126. When PLC executed RS instruction, STX and ETX will be sent out automatically.
2. When M1161 = ON, only the low byte (lower 8 bits) is valid for data communication, i.e. high byte will be ignored and low byte will be received and transmitted.



Sending data: (PLC -> external equipment)



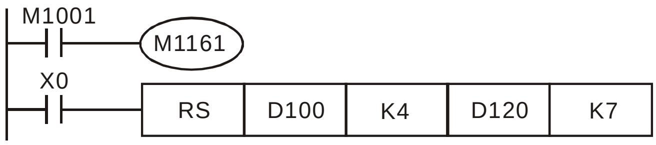
Receiving data: (External equipment -> PLC)



3. The STX and ETX of external equipments will be received by PLC in data receiving process, therefore, care should be taken on the setting of operand **n** (Length of data to be received).

16-bit mode:

1. STX (Start of Text) and ETX (End of text) are set up by M1126 and M1130 together with D1124~D1126. When PLC executed RS instruction, STX and ETX will be sent out automatically.
2. When M1161 = OFF, the 16-bit mode is selected, i.e. both high byte and low byte of the 16-bit data will be received and transmitted.



Sending data: (PLC -> external equipment)

STX	D100L	D100H	D101L	D101H	ETX1	ETX2

S Source data register, starting from the lower 8 bits of D100

m length = 4

Receiving data: (External equipment -> PLC)

D120L	D120H	D121L	D121H	D122L	D122H	D123L

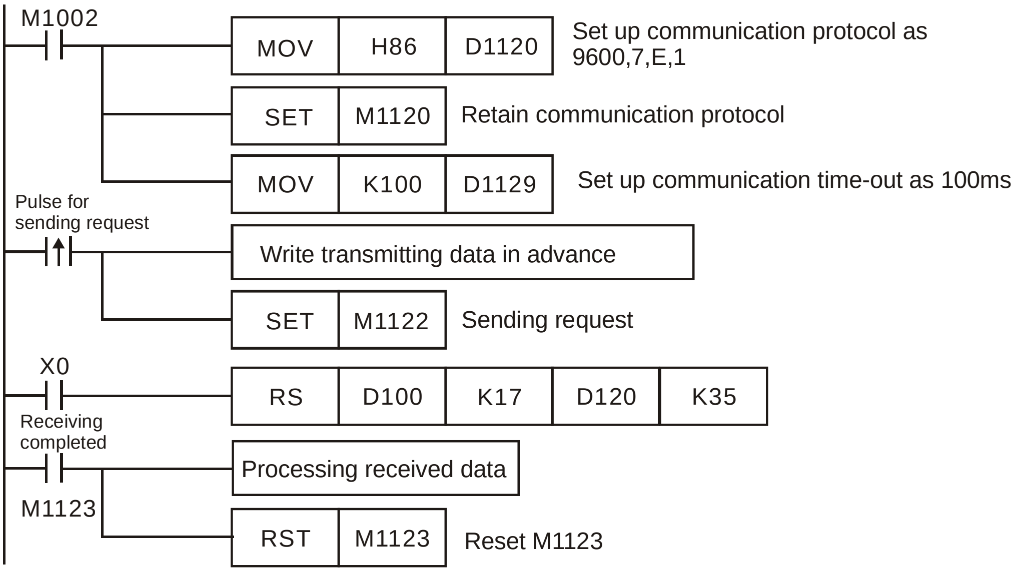
STX D Registers for received data, starting from the lower 8 bits of D120 ETX1 ETX2

n Length = 7

3. The STX and ETX of external equipments will be received by PLC in data receiving process, therefore, care should be taken on the setting of operand **n** (Length of data to be received)

Program Example 3: COM2 RS-485

1. Connect PLC to VFD-B series AC motor drives (AC motor drive in ASCII Mode; PLC in 16-bit mode and M1161 = OFF).
2. Write the data to be sent into registers starting from D100 in advance in order to read 6 data starting from address H2101 on VFD-B



PLC ⇨ VFD-B, PLC sends **": 01 03 2101 0006 D4 CR LF "**

VFD-B ⇨ PLC, PLC receives **": 01 03 0C 0100 1766 0000 0000 0136 0000 3B CR LF "**

Registers for sent data (PLC sends out messages)

<table>
<tr><th>Register</th><th colspan="2">Data</th><th colspan="2">Explanation</th></tr>
<tr><td>D100 low</td><td>': '</td><td>3A H</td><td colspan="2">STX</td></tr>
<tr><td>D100 high</td><td>'0'</td><td>30 H</td><td>ADR 1</td><td rowspan="2">Address of AC motor drive: ADR (1,0)</td></tr>
<tr><td>D101 low</td><td>'1'</td><td>31 H</td><td>ADR 0</td></tr>
<tr><td>D101 high</td><td>'0'</td><td>30 H</td><td>CMD 1</td><td rowspan="2">Instruction code: CMD (1,0)</td></tr>
<tr><td>D102 low</td><td>'3'</td><td>33 H</td><td>CMD 0</td></tr>
<tr><td>D102 high</td><td>'2'</td><td>32 H</td><td colspan="2" rowspan="4">Start data address</td></tr>
<tr><td>D103 low</td><td>'1'</td><td>31 H</td></tr>
<tr><td>D103 high</td><td>'0'</td><td>30 H</td></tr>
<tr><td>D104 low</td><td>'1'</td><td>31 H</td></tr>
<tr><td>D104 high</td><td>'0'</td><td>30 H</td><td colspan="2" rowspan="4">Number of data (counted by words)</td></tr>
<tr><td>D105 low</td><td>'0'</td><td>30 H</td></tr>
<tr><td>D105 high</td><td>'0'</td><td>30 H</td></tr>
<tr><td>D106 low</td><td>'6'</td><td>36 H</td></tr>
<tr><td>D106 high</td><td>'D'</td><td>44 H</td><td>LRC CHK 1</td><td rowspan="2">Error checksum: LRC CHK (0,1)</td></tr>
<tr><td>D107 low</td><td>'4'</td><td>34 H</td><td>LRC CHK 0</td></tr>
<tr><td>D107 high</td><td>CR</td><td>D H</td><td colspan="2" rowspan="2">END</td></tr>
<tr><td>D108 low</td><td>LF</td><td>A H</td></tr>
</table>

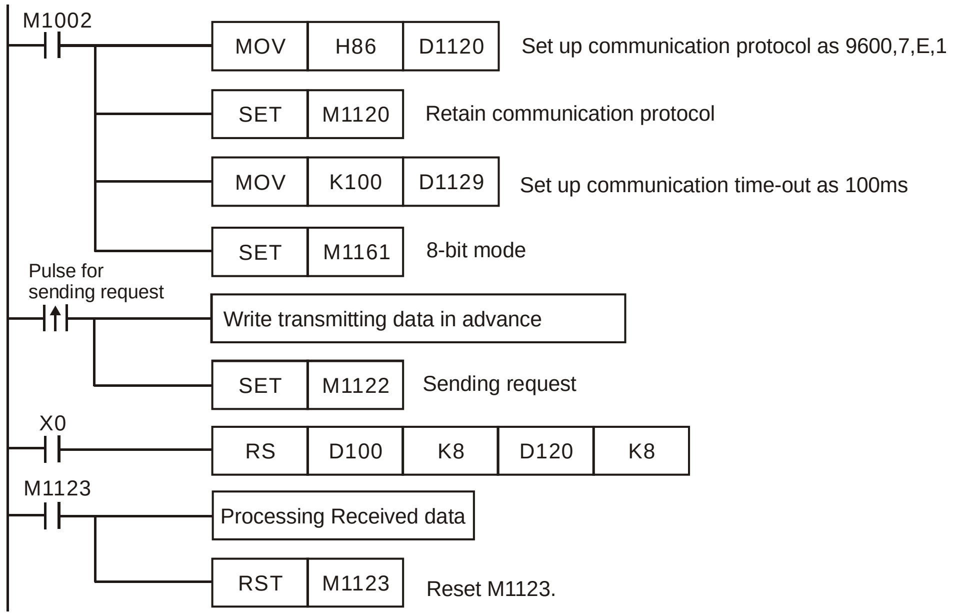
Registers for received data (VFD-B responds with messages)

Register	Data		Explanation
D120 low	‘: ’	3A H	STX
D120 high	‘0’	30 H	ADR 1
D121 low	‘1’	31 H	ADR 0
D121 high	‘0’	30 H	CMD 1
D122 low	‘3’	33 H	CMD 0
D122 high	‘0’	30 H	Number of data (counted by byte)
D123 low	‘C’	43 H	
D123 high	‘0’	30 H	Content of address 2101 H
D124 low	‘1’	31 H	
D124 high	‘0’	30 H	
D125 low	‘0’	30 H	
D125 high	‘1’	31 H	Content of address 2102 H
D126 low	‘7’	37 H	
D126 high	‘6’	36 H	
D127 low	‘6’	36 H	
D127 high	‘0’	30 H	Content of address 2103 H
D128 low	‘0’	30 H	
D128 high	‘0’	30 H	
D129 low	‘0’	30 H	
D129 high	‘0’	30 H	Content of address 2104 H
D130 low	‘0’	30 H	
D130 high	‘0’	30 H	
D131 low	‘0’	30 H	
D131 high	‘0’	30 H	Content of address 2105 H
D132 low	‘1’	31 H	
D132 high	‘3’	33 H	
D133 low	‘6’	36 H	
D133 high	‘0’	30 H	Content of address 2106 H
D134 low	‘0’	30 H	
D134 high	‘0’	30 H	
D135 low	‘0’	30 H	
D135 high	‘3’	33 H	LRC CHK 1
D136 low	‘B’	42 H	LRC CHK 0
D136 high	CR	D H	END
D137 low	LF	A H	

3. The status of Delta VFD series inverters can also be accessed by handy instruction API 105 RDST instruction through COM2/COM3 on PLC.

Program Example 4: COM2 RS-485

1. Connect PLC to VFD-B series AC motor drives (AC motor drive in RTU Mode; PLC in 16-bit mode and M1161 = ON).
2. Write the data to be sent into registers starting from D100 in advance. Write H12 (Forward running) into H2000 (VFD-B parameter address).



PLC ⇨ VFD-B, PLC sends: **01 06 2000 0012 02 07**

VFD-B ⇨ PLC, PLC receives: **01 06 2000 0012 02 07**

Registers for sent data (PLC sends out messages)

Register	Data	Explanation
D100 low	01 H	Address
D101 low	06 H	Function
D102 low	20 H	Data address
D103 low	00 H	
D104 low	00 H	Data content
D105 low	12 H	
D106 low	02 H	CRC CHK Low
D107 low	07 H	CRC CHK High

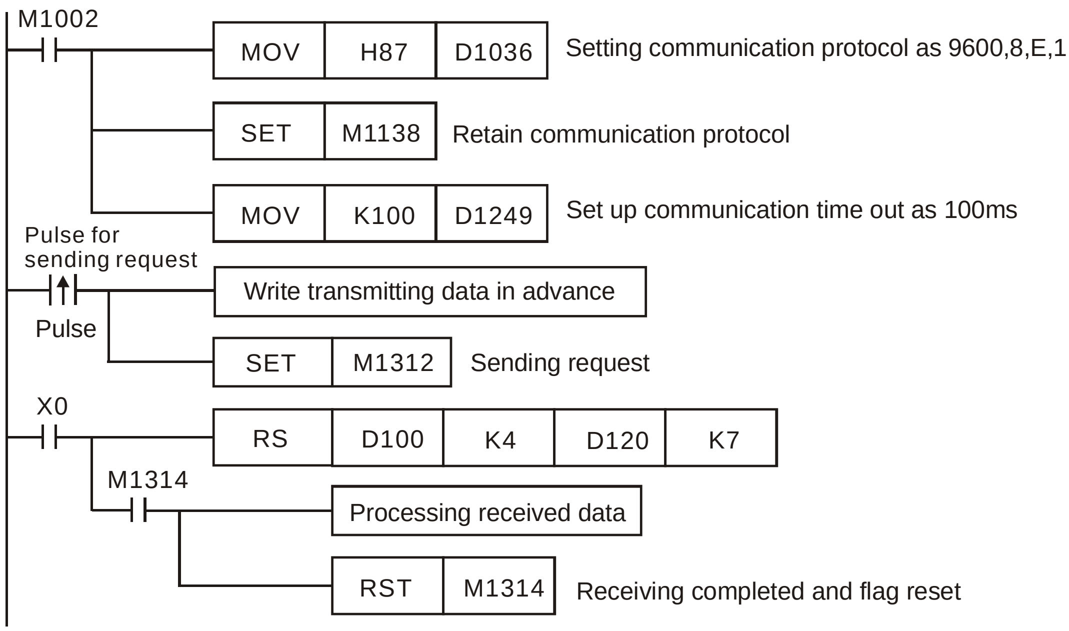
Registers for received data (VFD-B responds with messages)

Register	Data	Explanation
D120 low	01 H	Address
D121 low	06 H	Function
D122 low	20 H	Data address
D123 low	00 H	
D124 low	00 H	Data content
D125 low	12 H	
D126 low	02 H	CRC CHK Low
D127 low	07 H	CRC CHK High

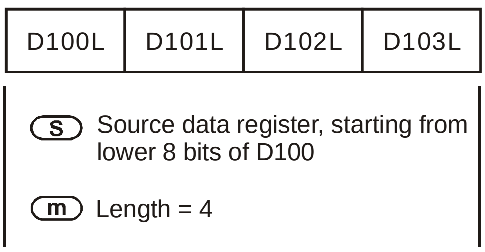
3. The forward running function of Delta's VFD series inverter can also be set by handy instruction API 102 FWD instruction through COM2/COM3 on PLC.

Program Example 5: COM1 RS-232

1. Only 8-bit mode is supported. Communication format and speed are specified by lower 8 bits of D1036.
2. STX/ETX setting function (M1126/M1130/D1124~D1126) is not supported.
3. High byte of 16-bit data is not supported. Only low byte is valid for data communication.
4. Write the data to be transmitted in advance into registers starting from D100 and set M1312 (COM1 sending request) as ON
5. When X0 = ON, RS instruction executes and PLC is ready for communication. D100 will then start to send out 4 data continuously. When data sending is over, M1312 will be automatically reset. (DO NOT apply RST M1312 in program). After approximate 1ms, PLC will start to receive 7 data and store the data in 7 consecutive registers starting from D120.
6. When data receiving is completed, M1314 will automatically be ON. When data processing on the received data is completed, M1314 has to be reset (OFF) and the PLC will be ready for communication again. However, DO NOT continuously execute RST M1314, i.e. it is suggested to connect the RST M1314 instruction after the drive contact M1314



Sending data: (PLC→External equipment)



Receving data: (External equipment→PLC)

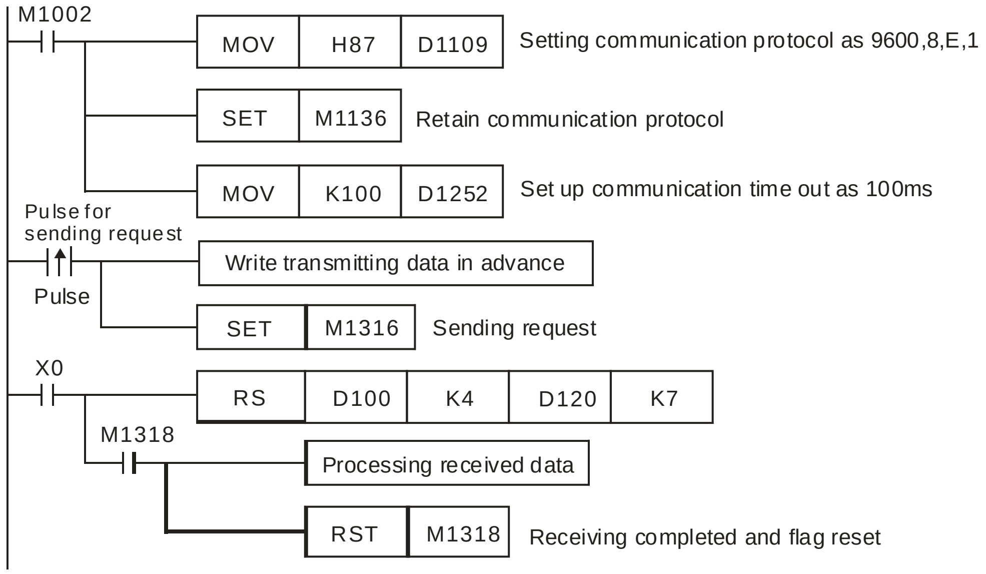
D120L	D121L	D122L	D123L	D124L	D125L	D126L

S Registers for received data, starting from lower 8 bits of D120

n Length = 7

Program Example 6: COM3 RS-485

1. Only 8-bit mode is supported. Communication format and speed are specified by lower 8 bits of D1109.
2. STX/ETX setting function (M1126/M1130/D1124~D1126) is not supported.
3. High byte of 16-bit data is not supported. Only low byte is valid for data communication.
4. Write the data to be transmitted in advance into registers starting from D100 and set M1316 (COM3 sending request) as ON
5. When X0 = ON, RS instruction executes and PLC is ready for communication. D100 will then start to send out 4 data continuously. When data sending is over, M1316 will be automatically reset. (DO NOT apply RST M1316 in program). After approximate 1ms, PLC will start to receive 7 data and store the data in 7 consecutive registers starting from D120.
6. When data receiving is completed, M1318 will automatically be ON. When data processing on the received data is completed, M1318 has to be reset (OFF) and the PLC will be ready for communication again. However, DO NOT continuously execute RST M1318, i.e. it is suggested to connect the RST M1318 instruction after the drive contact M1318.



Sending data: (PLC→External equipment)

D100L	D101L	D102L	D103L

S Source data register, starting from lower 8 bits of D100

m Length = 4

Receving data: (External equipment→PLC)

D120L	D121L	D122L	D123L	D124L	D125L	D126L

S Registers for received data, starting from lower 8 bits of D120

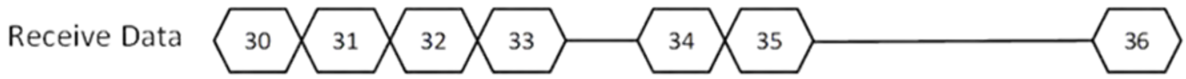
n Length = 7

Program Example 7: COM3 RS-485

Executing RS instruction with the newly added M1263, you can set the data receiving as completed, when the data receiving stops for a period of time that is longer than what you have set in D1168 (The specific end word to be detected for RS instruction to execute an interruption request). Be sure to set the time in D1168 shorter than the time you set in D1129; otherwise, it may be treated as communication timeout.

The flow of data receiving may NOT always be as regular or consistent. After receiving one word of data, the PLC CPU starts to time to determine if the data receiving continues and when the data receiving stops longer than the time set in D1168, the data receiving will be seen as completed.

Example:



We can see there are two noticeable stops in this diagram.

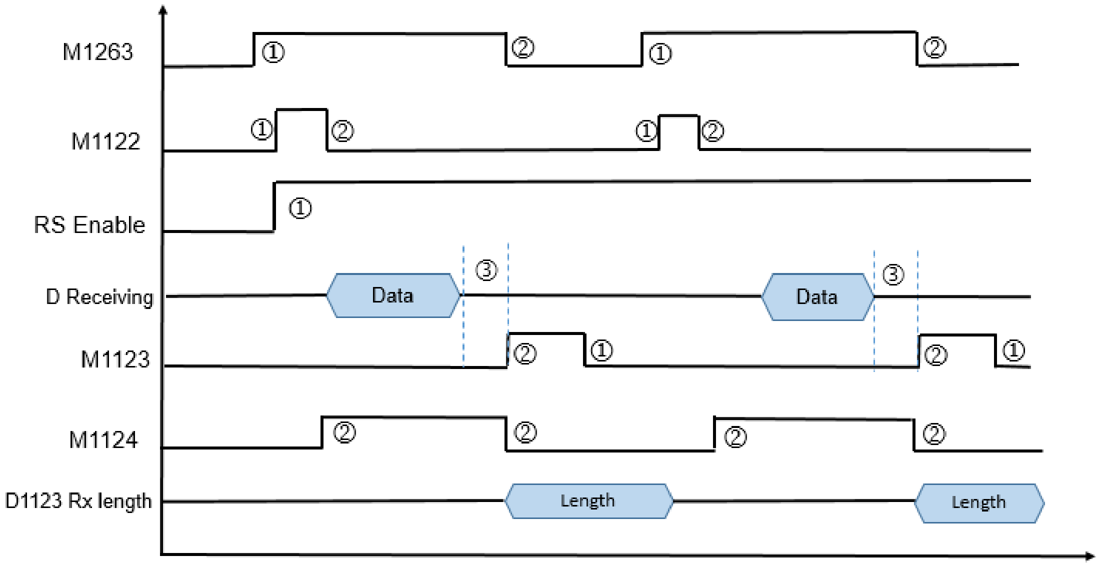
The flow of data receiving stopped between data 33 and data 34. But since the stopped time is less than what is set in D1168, the data 34 is treated as part of the data string and the data receiving continues.

The flow of data receiving stopped between data 35 and data 36. Since the stopped time is longer than what is set in D1168, the data 36 is NOT treated as part of the data string and the data receiving stops after data 35 is received.

1. Explanations on the data devices D and flags M.

M1263	Execute RS instruction with M1263 to set the data receiving as completed, when the data receiving stops for a period of time that is longer than what D1168 was set (The specific end word to be detected for RS instruction to execute an interruption request). If the receiving is not started, the detecting will not be started either.
D1168	When M1263 is ON, the value in D1168 indicates a specific end word to be seen as an interrupt request; unit: ms. It is suggested to set this value shorter than what the timeout value is set in D1129. When M1263 is OFF, and the received data is low-byte of a specific word, the interrupt (I150) is triggered.
D1123	When M1263 is OFF, the value in D1123 indicates the remaining bytes of data to be received. When M1263 is ON, the value in D1123 indicates the bytes of data that are received.
M1123	After data receiving is complete, M1123 will be switched to ON, and users should reset this flag to OFF.
M1124	When in the receiving mode, M1124 will be switched to ON, and after receiving is complete, this flag will be OFF.
M1129 D1129	When in the receiving mode, if the time before receiving started is longer than what is set in D1129, M1129 will be ON.

2. **Timing diagram:**



① Users set and reset	② PLC sets and resets
③ The data receiving stops for a period of time that is longer than what is set in D1168	

3. Settings for executing RS instruction:

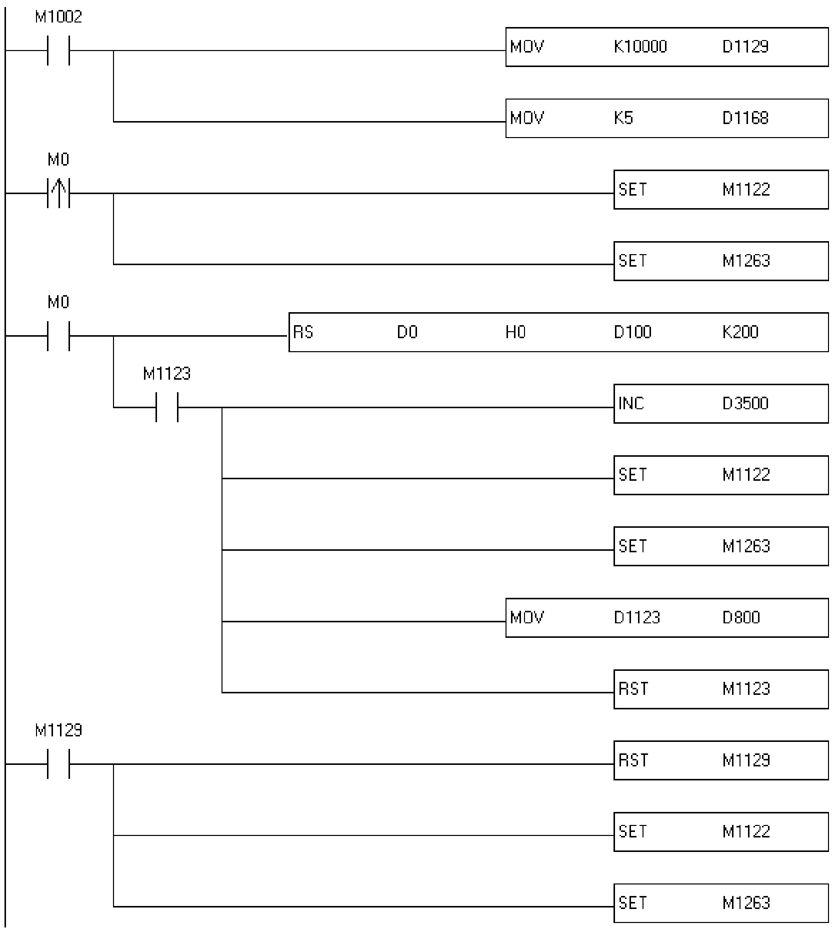
   When the sending data length is 0, that indicates it is in the receiving mode. If the sending data length is not 0, it will send that specific length of data first and once the sending is complete, it will be in the receiving mode. If executing RS instruction with M1263, the receiving data length should not be 0 and the value here should be set to a length that is bigger than the data length to be sent but less than 255.

4. Operational steps and explanation:

   Step 1: Use the rising-edge triggered on M0 to set flag M1263 to be ON and enable flag M1122 on COM2.

   Step 2: Specify the parameters for RS instruction, including where to store the received data and the maximum receiving/sending data length and then use M0 to enable the execution of RS instruction.

   Step 3: Once the completion flag M1123 is ON, the receiving is complete. If you need to receive data again, reset flag M1123 to OFF.



- Set the communication timeout to 10 seconds.
- A specific end word to be seen as an interrupt request
- Data receiving starts on COM2
- Set the data receiving as completed, when the data receiving stops for longer than what is set in D1168.
- Start executing RS instruction
- Accumulate the completion number of data receiving
- Data receiving starts on COM2 again
- Set the data receiving as completed, when the data receiving stops for longer than what is set in D1168.
- Move the received data length
- Reset the completion flag on COM2
- Reset the communication timeout flag on COM2
- Data receiving starts on COM2 again
- Set the data receiving as completed, when the data receiving stops for longer than what is set in D1168.

Points to note:

1. **PLC COM1 RS-232:** Associated flags (Auxiliary relays) and special registers (Special D) for communication instructions RS / MODRD

Flag	Function	Action
M1138	COM1 retain communication settings. Communication settings will be reset (changed) according to the content in D1036 after every scan cycle. Users can set ON M1138 if the communication protocol requires to be retained. When M1138 = ON, communication settings will not be reset (changed) when communication instructions are being processed, even if the content in D1036 is changed. <u>Supported communication instructions:</u> RS / MODRW	User sets and resets
M1139	COM1 ASCII / RTU mode selection, ON: RTU mode, OFF: ASCII mode. <u>Supported communication instructions:</u> RS / MODRW	User sets and resets
M1312	COM1 sending request. Before executing communication instructions, users need to set M1312 to ON by trigger pulse, so that the data sending and receiving will be started. When the communication is completed, PLC will reset M1312 automatically. <u>Supported communication instructions:</u> RS / MODRW	User sets and system resets
M1313	COM1 data receiving ready. When M1313 is ON, PLC is ready for data receiving <u>Supported communication instructions</u>: RS / MODRW	System
M1314	COM1 Data receiving completed. When data receiving of communication instructions is completed, M1314 will be ON. Users can process the received data when M1314 is ON. When data processing is completed, M1314 has to be reset by users. <u>Supported communication instructions</u>: RS / MODRW	System sets and user resets
M1315	COM1 receiving error. M1315 will be set ON when errors occur and the error code will be stored in D1250. <u>Supported communication instructions:</u> RS / MODRW	System sets and user resets

Special register	Function
D1036	COM1 (RS-232) communication protocol. Refer to the following table in point 4 for protocol setting.
D1167	The specific end word to be detected for RS instruction to execute an interruption request (I140) on COM1 (RS-232). Supported communication instructions: RS
D1121	COM1 (RS-232) and COM2 (RS-485) communication address.
D1249	COM1 (RS-232) Communication time-out setting (unit: ms). If users set up time-out value in D1249 and the data receiving time exceeds the time-out value, M1315 will be set ON and the error code K1 will be stored in D1250. M1315 has to be reset manually when time-out status is cleared.
D1250	COM1 (RS-232) communication error code. Supported communication instructions: MODRW

2. **PLC COM2 RS-485:** Associated flags (Auxiliary relays) and special registers (Special D) for communication instructions RS / MODRD / MODWR / FWD / REV / STOP / RDST / RSTEF / MODRW.

Flag	Function	Action
M1120	Retain communication settings. Communication settings will be reset (changed) according to the content in D1120 after every scan cycle. Users can set ON M1120 if the communication protocol requires to be retained. When M1120 = ON, communication settings will not be reset (changed) when communication instructions are being processed, even if the content in D1120 is changed.	User sets/resets
M1121	Data transmission ready. M1121 = OFF indicates that RS-485 in COM2 is transmitting	System sets
M1122	Sending request. Before executing communication instructions, users need to set M1122 to ON by trigger pulse, so that the data sending and receiving will be started. When the communication is completed, PLC will reset M1122 automatically.	User sets, system resets
M1123	Data receiving completed. When data receiving of communication instructions is completed, M1123 will be ON. Users can process the received data when M1123 is ON. When data processing is completed, M1123 has to be reset by users. Supported communication instructions: RS	System sets ON and user resets

<table>
<tr><th>Flag</th><th>Function</th><th>Action</th></tr>
<tr><td>M1124</td><td>Data receiving ready. When M1124 is ON, PLC is ready for data receiving.</td><td>System sets</td></tr>
<tr><td>M1125</td><td>Communication ready status reset. When M1125 is set ON, PLC resets the communication (transmitting/receiving) ready status. M1125 has to be reset by users after resetting the communication ready status.</td><td rowspan="3">User sets/resets</td></tr>
<tr><td>M1126</td><td>Set STX/ETX as user-defined or system-defined in RS communication. For details please refer to the table in point 5. M1126 only supports RS instruction.</td></tr>
<tr><td>M1130</td><td>Set STX/ETX as user-defined or system-defined in RS communication. For details please refer to the table in point 5. M1130 only supports RS instruction</td></tr>
<tr><td>M1127</td><td>COM2 (RS-485) data sending/receiving/converting completed. RS instruction is NOT supported.
<u>Supported communication instructions:</u>
MODRD / MODWR / FWD / REV / STOP / RDST / RSTEF / MODRW</td><td>System sets and user resets</td></tr>
<tr><td>M1128</td><td>Transmitting/receiving status indication.</td><td>System sets</td></tr>
<tr><td>M1129</td><td>Receiving time out. If users set up time-out value in D1129 and the data receiving time exceeds the time-out value, M1129 will be set ON.</td><td>System sets and user resets</td></tr>
<tr><td>M1131</td><td>In ASCII mode, M1131 = ON only when MODRD/RDST/MODRW data is being converted to HEX.
<u>Supported communication instructions:</u>
MODRD / RDST / MODRW</td><td rowspan="4">System sets</td></tr>
<tr><td>M1140</td><td>MODRD/MODWR/MODRW data receiving error
<u>Supported communication instructions:</u>
MODRD / MODWR / MODRW</td></tr>
<tr><td>M1141</td><td>MODRD/MODWR/MODRW parameter error
<u>Supported communication instructions:</u>
MODRD / MODWR/ MODRW</td></tr>
<tr><td>M1142</td><td>Data receiving error of VFD-A handy instructions.
<u>Supported communication instructions:</u>
FWD / REV / STOP / RDST / RSTEF</td></tr>
</table>

Flag	Function	Action
M1143	ASCII / RTU mode selection. ON : RTU mode, OFF: ASCII mode. Supported communication instructions: RS / MODRD / MODWR / MODRW (When M1177 = ON, FWD / REV / STOP / RDST / RSTEF can also be applied.	User sets and resets
M1161	8/16-bit mode. ON: 8-bit mode. OFF: 16-bit mode Supported communication instructions: RS	User sets
M1177	Enable the communication instruction for Delta VFD series inverter. ON: VFD-A (Default), OFF: other models of VFD Supported communication instructions: FWD / REV / STOP / RDST / RSTEF	
M1263	Execute RS instruction with M1263 to set the data receiving as completed, when the data receiving stops for a period of time that is longer than what D1168 was set (The specific end word to be detected for RS instruction to execute an interruption request). If the receiving is not started, the detecting will not be started either.	User sets and system resets

Special register	Function
D1038	Delay time of data response when PLC is SLAVE in COM2, COM3 RS-485 communication, Range: 0~10,000. (Unit: 0.1ms). By using EASY PLC LINK in COM2, D1038 can be set to send next communication data with delay. (unit: one scan cycle)
D1050~D1055	Converted data for Modbus communication data processing. PLC automatically converts the ASCII data in D1070~D1085 into Hex data and stores the 16-bit Hex data into D1050~D1055 Supported communication instructions: MODRD / RDST
D1070~D1085	Feedback data (ASCII) of Modbus communication. When PLC's RS-485 communication instruction receives feedback signals, the data will be saved in the registers D1070~D1085 and then converted into Hex in other registers. RS instruction is not supported.
D1089~D1099	Sent data of Modbus communication. When PLC's RS-485 communication instruction (MODRD) sends out data, the data will be stored in D1089~D1099. Users can check the sent data in these registers. RS instruction is not supported

Special register	Function
D1120	COM2 (RS-485) communication protocol. Refer to the following table in point 4 for protocol setting.
D1121	COM1 (RS-232) and COM2 (RS-485) PLC communication address when PLC is slave.
D1122	COM2 (RS-485) Residual number of words of transmitting data.
D1123	When M1263 is OFF, the value in D1123 indicates the remaining bytes of data to be received. When M1263 is ON, the value in D1123 indicates the bytes of data that are received.
D1124	COM2 (RS-485) Definition of start character (STX) Refer to the following table in point 3 for the setting. Supported communication instruction: RS
D1125	COM2 (RS-485) Definition of first ending character (ETX1) Refer to the following table in point 3 for the setting. Supported communication instruction: RS
D1126	COM2 (RS-485) Definition of second ending character (ETX2) Refer to the following table in point 3 for the setting. Supported communication instruction: RS
D1129	COM2 (RS-485) Communication time-out setting (unit: ms). If users set up time-out value in D1129 and the data receiving time exceeds the time-out value, M1129 will be set ON and the error code K1 will be stored in D1130. M1129 has to be reset manually when time-out status is cleared.
D1130	COM2 (RS-485) Error code returning from Modbus. RS instruction is not included. Supported communication instructions: MODRD / MODWR / FWD / REV / STOP / RDST / RSTEF / MODRW
D1168	When M1263 is ON, the value in D1168 indicates a specific end word to be seen as an interrupt request; unit: ms. It is suggested to set this value shorter than what the timeout value is set in D1129. When M1263 is OFF, and the received data is low-byte of a specific word, the interrupt (I150) is triggered. Supported communication instruction: RS
D1256~D1295	For COM2 RS-485 MODRW instruction. D1256~D1295 store the sent data of MODRW instruction. When MODRW instruction sends out data,

Special register	Function
	the data will be stored in D1256~D1295. Users can check the sent data in these registers. Supported communication instruction: MODRW
D1296~D1311	For COM2 RS-485 MODRW instruction. D1296~D1311 store the converted hex data from D1070 ~ D1085 (ASCII). PLC automatically converts the received ASCII data in D1070 ~ D1085 into hex data. Supported communication instruction: MODRW

3. **PLC COM3 RS-485:** Associated flags (Auxiliary relays) and special registers (Special D) for communication instructions RS / MODRW and FWD / REV / STOP / RDST / RSTEF when M1177 = ON.

Flag	Function	Action
M1136	COM3 retain communication settings. Communication settings will be reset (changed) according to the content in D1109 after every scan cycle. Users can set ON M1136 if the communication protocol requires to be retained. When M1136 = ON, communication settings will not be reset (changed) when communication instructions are being processed, even if the content in D1109 is changed	User sets and resets
M1320	COM3 ASCII / RTU mode selection. ON : RTU mode, OFF: ASCII mode.	
M1316	COM3 sending request. Before executing communication instructions, users need to set M1316 to ON by trigger pulse, so that the data sending and receiving will be started. When the communication is completed, PLC will reset M1316 automatically.	User sets, system resets
M1317	Data receiving ready. When M1317 is ON, PLC is ready for data receiving.	System sets
M1318	COM3 data receiving completed.	System sets, user resets
M1319	COM3 data receiving error. M1319 will be set ON when errors occur and the error code will be stored in D1252	System sets, user resets

Special register	Function
D1038	Delay time of data response when PLC is SLAVE in COM2, COM3 RS-485 communication, Range: 0~10,000. (unit: 0.1ms). By using EASY PLC LINK in COM2, D1038 can be set to send next communication data with delay. (unit: one scan cycle)
D1109	COM3 (RS-485) communication protocol. Refer to the following table in point 4 for protocol setting.
D1169	The specific end word to be detected for RS instruction to execute an interruption request (I160) on COM3 (RS-485). Supported communication instructions: RS
D1252	COM3 (RS-485) Communication time-out setting (ms). If users set up time-out value in D1252 and the data receiving time exceeds the time-out value, M1319 will be set ON and the error code K1 will be stored in D1253. M1319 has to be reset manually when time-out status is cleared.
D1253	COM3 (RS-485) communication error code
D1255	COM3 (RS-485) PLC communication address when PLC is Slave.

4. Corresponding table between COM ports and communication settings/status.

	COM1	COM2	COM3	Function Description
Protocol setting	M1138	M1120	M1136	Retain communication setting
	M1139	M1143	M1320	ASCII/RTU mode selection
	D1036	D1120	D1109	Communication protocol
	D1121	D1121	D1255	PLC communication address
Sending request	-	M1161	-	8/16 bit mode selection
	-	M1121	-	Indicate transmission status
	M1312	M1122	M1316	Sending request
	-	M1126	-	Set STX/ETX as user/system defined. (RS) RS)RSTX/ETX
	-	M1130	-	Set STX/ETX as user/system defined. (RS)
	-	D1124	-	Definition of STX (RS)
Sending request	-	D1125	-	Definition of ETX1 (RS)
	-	D1126	-	Definition of ETX2 (RS)
	D1249	D1129	D1252	Communication timeout setting (ms)
	-	D1122	-	Residual number of words of transmitting data
	-	D1256	-	Store the sent data of MODRW instruction.

	COM1	COM2	COM3	Function Description
		~ D1295		
	-	D1089 ~ D1099	-	Store the sent data of MODRD / MODWR / FWD / REV / STOP / RDST / RSTEF instruction
Data receiving	M1313	M1124	M1317	Data receiving ready
	-	M1125	-	Communication ready status reset
	-	M1128	-	Transmitting/Receiving status Indication
	-	D1123	-	When M1263 is OFF, the value in D1123 indicates the remaining bytes of data to be received. When M1263 is ON, the value in D1123 indicates the bytes of data that are received.
	-	D1070 ~ D1085	-	Store the feedback data of Modbus communication. RS instruction is not supported.
	D1167	D1168	D1169	Store the specific end word to be detected for executing interrupts I140/I150/I160 (RS) When M1263 is ON, the value in D1168 indicates a specific end word to be seen as an interrupt request; unit: ms. It is suggested to set this value shorter than what the timeout value is set in D1129.
Receiving completed	M1314	M1123	M1318	Data receiving completed
	-	M1127	-	COM2 (RS-485) data sending / receiving / converting completed. (RS instruction is not supported)
	-	M1131	-	ON when MODRD/RDST/MODRW data is being converted from ASCII to Hex
	-	D1296 ~ D1311	-	Store the converted HEX data of MODRW instruction.
	-	D1050 ~ D1055	-	Store the converted HEX data of MODRD instruction
Errors	M1315	-	M1319	Data receiving error

	COM1	COM2	COM3	Function Description
Errors	D1250	-	D1253	Communication error code
	-	M1129	-	COM2 (RS-485) receiving time out
	-	M1140	-	COM2 (RS-485) MODRD/MODWR/MODRW data receiving error
	-	M1141	-	MODRD/MODWR/MODRW parameter error (Exception Code exists in received data) Exception Code is stored in D1130
	-	M1142	-	Data receiving error of VFD-A handy instructions (FWD/REV/STOP/RDST/RSTEF)
	-	D1130	-	COM2 (RS-485) Error code returning from Modbus communication

5. Communication protocol settings: D1036(COM1 RS-232) / D1120(COM2 RS-485) / D1109(COM3 RS-485)

   ES2-C: V4.0 or later does NOT support 600 bps.

	Content		
b0	Data Length	**0:** 7 data bits	**1:** 8 data bits
b1 b2	Parity bit	00: None 01: Odd 11: Even	
b3	Stop bits	**0:** 1 bit	**1:** 2bits
b4 b5 b6 b7	Baud rate	0100(H4): 600 bps 0101(H5): 1200 bps 0110(H6): 2400 bps 0111(H7): 4800 bps 1000(H8): 9600 bps 1001(H9): 19200 bps 1010(HA): 38400 bps 1011(HB): 57600 bps 1100(HC): 115200 bps 1101(HD): 500000 bps (COM2 / COM3) 1110 (HE): 31250 bps (COM2 / COM3) 1111 (HF): 921000 bps (COM2 / COM3)	
b8 (D1120)	STX	0: None	1: D1124
b9 (D1120)	ETX1	0: None	1: D1125
b10 (D1120)	ETX2	0: None	1: D1126
b11~b15	N/A		

6. When RS instruction is applied for communication between a PLC and peripheral devices, usually the STX (Start of the text) and the ETX (End of the text) have to be defined. User can use D1124~D1126 to set the STX and the ETX by means of COM2, or use the STX and the ETX defined by the PLC. If the users use M1126, M1130, D1124~D1126 to set the STX and the ETX, b8~b10 in D1120 using the RS-485 communication protocol need to be set to 1. Please refer to the table below.

		M1130	
		0	1
M1126	0	D1124: user defined D1125: user defined D1126: user defined	D1124: H 0002 D1125: H 0003 D1126: H 0000 (no setting)
	1	D1124: user defined D1125: user defined D1126: user defined	D1124: H 003A (':') D1125: H 000D (CR) D1126: H 000A (LF)

7. Example of setting communication format in D1120:

Communication format:

Baud rate: 9600, 7, N, 2

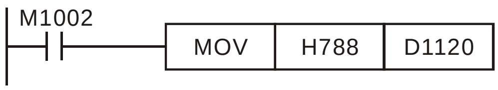
STX : ": "

ETX1 : "CR"

ETX2 : "LF"

Check to the table in point 4 and the set value H788 can be referenced corresponding to the baud rate. Set the value into D1120.

	b15															b0
D1120	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0
	0 N/A				7				8				8			



When STX, ETX1 and ETX2 are applied, care should be taken on setting the ON/OFF status of M1126 and M1130.

8. D1250(COM1)、D1253(COM3) communication error code:

Value	Error Description
H0001	Communication time-out
H0002	Checksum error
H0003	Exception Code exists
H0004	Command code error / data error
H0005	Communication data length error

9. Corresponding table between D1167~D1169 and the associated interrupt pointers. (Only lower 8 bits are valid)

COM Port	I1□0 interrupt	Special D
COM1	I140	D1167
COM2	I150	D1168
COM3	I160	D1169

10. Take standard MODBUS format for example:

ASCII mode

Field Name	Descriptions
STX	Start word = ‘: ’ (3AH)
Address Hi	Communication address:
Address Lo	The 8-bit address consists of 2 ASCII codes
Function Hi	Function code:
Function Lo	The 8-bit function code consists of 2 ASCII codes
DATA (n-1)	Data content:
…….	n × 8-bit data content consists of 2n ASCll codes
DATA 0	
LRC CHK Hi	LRC check sum:
LRC CHK Lo	8-bit check sum consists of 2 ASCll code
END Hi	End word:
END Lo	END Hi = CR (0DH), END Lo = LF(0AH)

The communication protocol is in Modbus ASCII mode, i.e. every byte is composed of 2 ASCII characters. For example, 64Hex is ‘64’ in ASCII, composed by ‘6’ (36Hex) and ‘4’ (34Hex).

Every character ‘0’…’9’, ‘A’…’F’ corresponds to an ASCII code.

Character	‘0’	‘1’	‘2’	‘3’	‘4’	‘5’	‘6’	‘7’
ASCII code	30H	31H	32H	33H	34H	35H	36H	37H

Character	‘8’	‘9’	‘A’	‘B’	‘C’	‘D’	‘E’	‘F’
ASCII code	38H	39H	41H	42H	43H	44H	45H	46H

Start word (STX): ‘: ’ (3AH)

Address:

‘0’ ‘0’: Broadcasting to all drives (Broadcast)

‘0’ ‘1’: toward the drive at address 01

‘0’ ‘F’: toward the drive at address 15

‘1’ ‘0’: toward the drive at address 16

and so on, max. address: 254 (‘F’ ‘E’)

Function code:

‘0’ ‘1’: Reading several bit devices

‘0’ ‘2’: Reading several bit devices (read-only devices)

‘0’ ‘3’: Reading several word devices

‘0’ ‘4’: Reading several word devices (read-only devices)

‘0’ ’5’: Writing a state in a single bit device

‘0’ ‘6’: Writing data in a single word device

‘0’ ’F’: Writing states in bit devices

‘1’ ‘0’: Writing data in word devices

‘1’ ‘7’: Reading word devices and writing data in word devices

Data characters:

The data sent by the user

LRC checksum:

LCR checksum is 2’s complement of the value added from Address to Data Characters.

For example: 01H + 03H + 21H + 02H + 00H + 02H = 29H. 2’s complement of 29H = D7H.

End word (END):

Fix the END as END Hi = CR (0DH), END Lo = LF (0AH)

Example:

Read 2 continuous data stored in the registers of the drive at address 01H (see the table below). The start register is at address 2102H.

Inquiry message:

STX	‘: ’
Address	‘0’
	‘1’
Function code	‘0’
	‘3’
Start address	‘2’
	‘1’
	‘0’
	‘2’
Number of data (count by word)	‘0’
	‘0’
	‘0’
	‘2’
LRC Checksum	‘D’
	‘7’

Response message:

STX	‘: ’
Address	‘0’
	‘1’
Function code	‘0’
	‘3’
Number of data (count by byte)	‘0’
	‘4’
Content of start address 2102H	‘1’
	‘7’
	‘7’
	‘0’
Content of address 2103H	‘0’
	‘0’
	‘0’
	‘0’

Inquiry message:

END	CR
	LF

Response message:

LRC Checksum	‘7’
	‘1’
END	CR
	LF

RTU mode

Field Name	Descriptions
START	Refer to the following explanation
Address	Communication address: n 8-bit binary
Function	Function code: n 8-bit binary
DATA (n-1)	Data: n × 8-bit data
…….	
DATA 0	
CRC CHK Low	CRC checksum: 16-bit CRC consists of 2 8-bit binary data
CRC CHK High	
END	Refer to the following explanation

START/END:

RTU Timeout Timer:

Baud rate(bps)	RTU timeout timer (ms)	Baud rate (bps)	RTU timeout timer (ms)
300	40	9,600	2
600	21	19,200	1
1,200	10	38,400	1
2,400	5	57,600	1
4,800	3	115,200	1

Address:

00 H: Broadcasting to all drives (Broadcast)

01 H: toward the drive at address 01

0F H: toward the drive at address 15

10 H: toward the drive at address 16

and so on, max. address: 254 (‘FE’)

Function code:

03 H: read contents from multiple registers

06 H: write one word into single register

10 H: write contents to multiple registers

Data characters:

The data sent by the user

CRC checksum: Starting from Address and ending at Data Content. The calculation is as follows:

Step 1: Set the 16-bit register (CRC register) = FFFFH

Step 2: Operate XOR on the first 8-bit message (Address) and the lower 8 bits of CRC register. Store the result in the CRC register.

Step 3: Right shift CRC register for a bit and fill “0” into the highest bit.

Step 4: Check the lowest bit (bit 0) of the shifted value. If bit 0 is 0, fill in the new value obtained at step 3 to CRC register; if bit 0 is NOT 0, operate XOR on A001H and the shifted value and store the result in the CRC register.

Step 5: Repeat step 3 – 4 to finish all operation on all the 8 bits.

Step 6: Repeat step 2 – 5 until the operation of all the messages are completed. The final value obtained in the CRC register is the CRC checksum. Care should be taken when placing the LOW byte and HIGH byte of the obtained CRC checksum.

Example:

Read 2 continuous data stored in the registers of the drive at address 01H (see the table below). The start register is at address 2102H

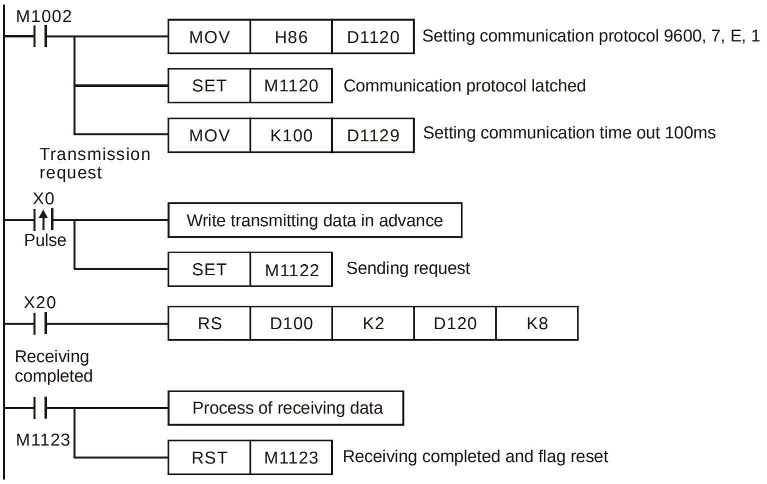
Inquiry message:

Field Name	Data (Hex)
Address	01 H
Function	03 H
Start data address	21 H
	02 H
Number of data (count by word)	00 H
	02 H
CRC CHK Low	6F H
CRC CHK High	F7 H

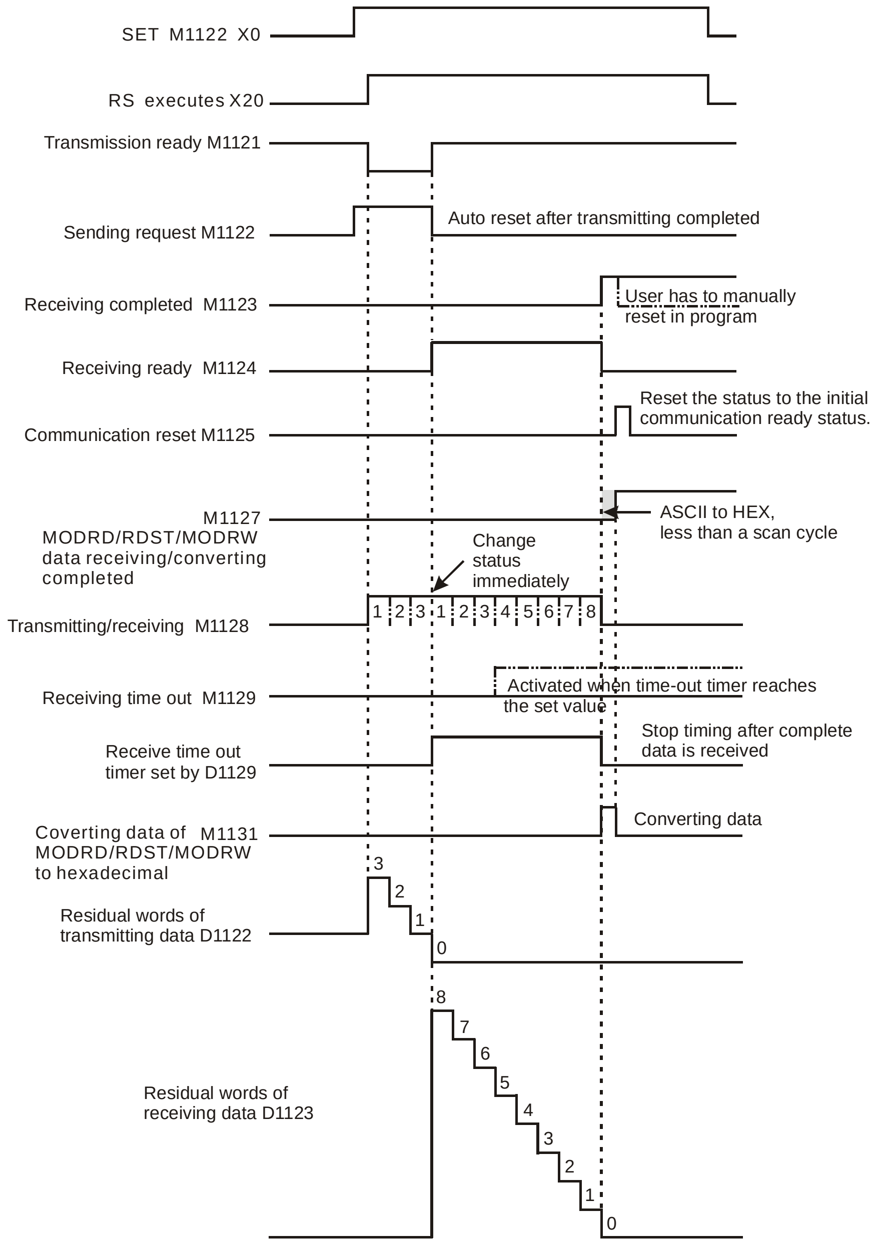
Response message:

Field Name	Data (Hex)
Address	01 H
Function	03 H
Number of data (count by byte)	04 H
Content of data address 2102H	17 H
	70 H
Content of data address 2103H	00 H
	00 H
CRC CHK Low	FE H
CRC CHK High	5C H

Example program of RS-485 communication:



Timing diagram:



API	Mnemonic			Operands	Function	Controllers			
81	D	PRUN	P	S D	Parallel Run	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S							*		*							PRUN, PRUNP: 5 steps
D								*	*							DPRUN, DPRUNP: 9 steps

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

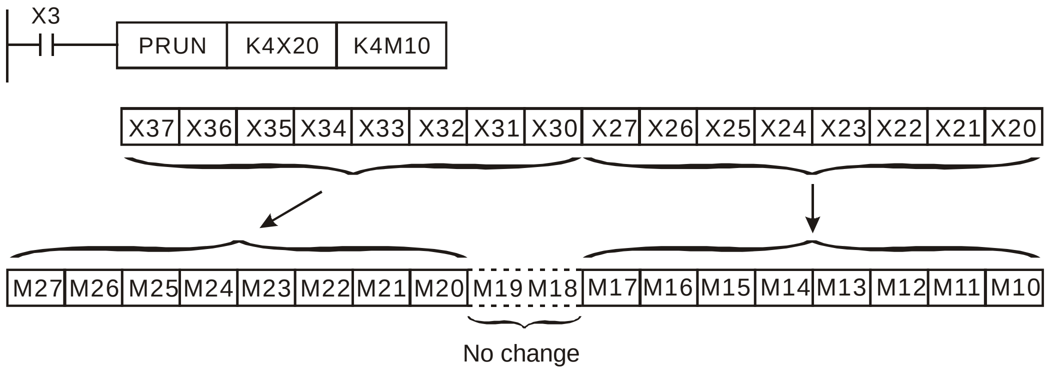
S: Source device **D**: Destination device

Explanations:

1. This instruction sends the content in **S** to **D** in the form of octal system
2. The start device of X, Y, M in KnX, KnY, KnM format should be a multiple of 10, e.g. X20, M20, Y20.
3. When operand **S** is specified as KnX, operand **D** should be specified as KnM.
4. When operand **S** is specified as KnM, operand **D** should be specified as KnY.

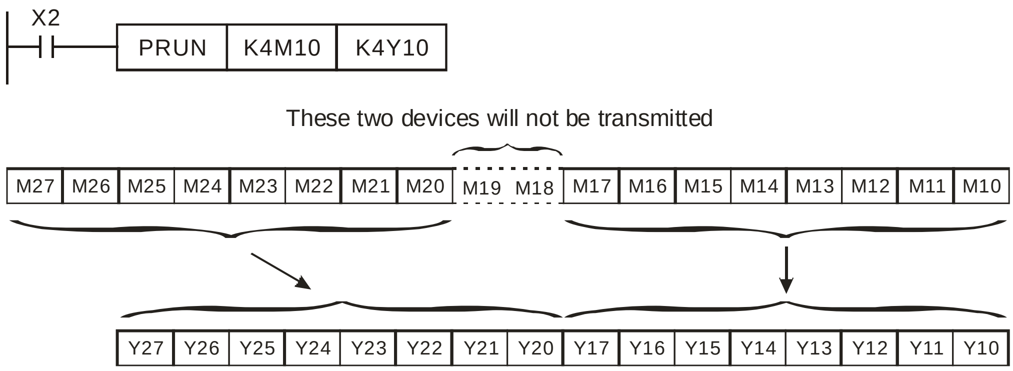
Program Example 1:

When X3 = ON, the contents of K4X20 will be sent to K4M10 in octal form.



Program Example 2:

When X2 = ON, the content in K4M10 will be sent to K4Y10 in octal form.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
82	ASCI	P	S D n	Convert Hex to ASCII				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											ASCI, ASCIP: 7 steps
S					*	*	*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

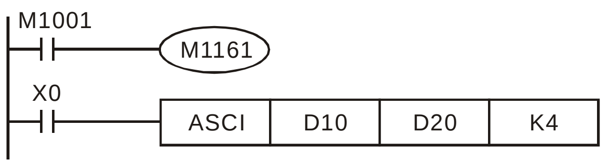
S: Source device **D**: Destination device **n**: Number of nibbles to be converted (**n** = 1~256)

Explanations:

1. 16-bit conversion mode: When M1161 = OFF, the instruction converts every nibble of the Hex data in **S** into ASCII codes and send them to the higher 8 bits and lower 8 bits of **D**. **n** = the converted number of nibbles.
2. 8-bit conversion mode: When M1161 = ON, the instruction converts every nibble of the Hex data in **S** into ASCII codes and send them to the lower 8 bits of **D**. **n** = the number of converted nibbles. (All higher 8 bits of **D** = 0).
3. Flag: M1161 (8/16 bit mode switch)
4. Available range for Hex data: 0~9, A~F

Program Example 1:

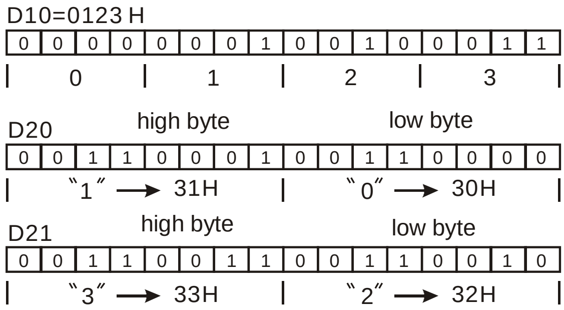
1. M1161 = OFF, 16-bit conversion.
2. When X0 = ON, convert the 4 hex values (nibbles) in D10 into ASCII codes and send the result to registers starting from D20.



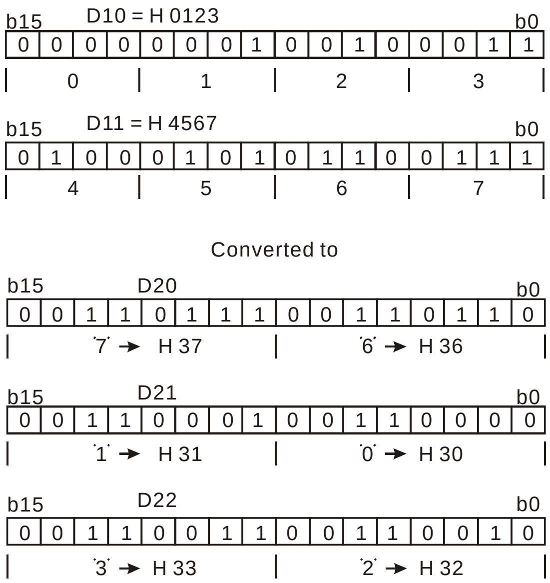
3. Assume:

(D10) = 0123 H	‘0’ = 30H	‘4’ = 34H	‘8’ = 38H
(D11) = 4567 H	‘1’ = 31H	‘5’ = 35H	‘9’ = 39H
(D12) = 89AB H	‘2’ = 32H	‘6’ = 36H	‘A’ = 41H
(D13) = CDEF H	‘3’ = 33H	‘7’ = 37H	‘B’ = 42H

4. When **n** = 4, the bit structure will be as:



5. When **n** is 6, the bit structure will be as:



6. When **n** = 1 to 16:

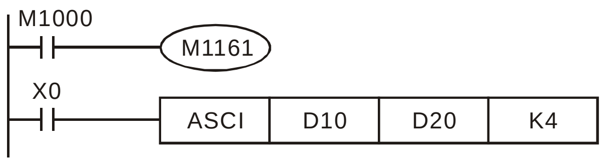
n / D	K1	K2	K3	K4	K5	K6	K7	K8
D20 low byte	"3"	"2"	"1"	"0"	"7"	"6"	"5"	"4"
D20 high byte		"3"	"2"	"1"	"0"	"7"	"6"	"5"
D21 low byte			"3"	"2"	"1"	"0"	"7"	"6"
D21 high byte				"3"	"2"	"1"	"0"	"7"
D22 low byte					"3"	"2"	"1"	"0"
D22 high byte						"3"	"2"	"1"
D23 low byte							"3"	"2"
D23 high byte								"3"
D24 low byte								
D24 high byte				No change				
D25 low byte								
D25 high byte								
D26 low byte								
D26 high byte								
D27 low byte								

n / D	K1	K2	K3	K4	K5	K6	K7	K8
D27 high byte								

n / D	K9	K10	K11	K12	K13	K14	K15	K16
D20 low byte	“B”	“A”	“9”	“8”	“F”	“E”	“D”	“C”
D20 high byte	“4”	“B”	“A”	“9”	“8”	“F”	“E”	“D”
D21 low byte	“5”	“4”	“B”	“A”	“9”	“8”	“F”	“E”
D21 high byte	“6”	“5”	“4”	“B”	“A”	“9”	“8”	“F”
D22 low byte	“7”	“6”	“5”	“4”	“B”	“A”	“9”	“8”
D22 high byte	“0”	“7”	“6”	“5”	“4”	“B”	“A”	“9”
D23 low byte	“1”	“0”	“7”	“6”	“5”	“4”	“B”	“A”
D23 high byte	“2”	“1”	“0”	“7”	“6”	“5”	“4”	“B”
D24 low byte	“3”	“2”	“1”	“0”	“7”	“6”	“5”	“4”
D24 high byte		“3”	“2”	“1”	“0”	“7”	“6”	“5”
D25 low byte			“3”	“2”	“1”	“0”	“7”	“6”
D25 high byte				“3”	“2”	“1”	“0”	“7”
D26 low byte					“3”	“2”	“1”	“0”
D26 high byte			No change			“3”	“2”	“1”
D27 low byte							“3”	“2”
D27 high byte								“3”

Program Example 2:

1. M1161 = ON, 8-bit conversion.
2. When X0 = ON, convert the 4 hex values (nibbles) in D10 into ASCII codes and send the result to registers starting from D20.



3. Assume:

(D10) = 0123 H	‘0’ = 30H	‘4’ = 34H	‘8’ = 38H
(D11) = 4567 H	‘1’ = 31H	‘5’ = 35H	‘9’ = 39H
(D12) = 89AB H	‘2’ = 32H	‘6’ = 36H	‘A’ = 41H
(D13) = CDEFH	‘3’ = 33H	‘7’ = 37H	‘B’ = 42H

4. When **n** is 2, the bit structure will be as:

D10=0123 H

0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1
0				1				2				3			

ASCII code of "2" in D20 is 32H

0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0
								3				2			

ASCII code of "3" in D21 is 33H

0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1
								3				3			

5. When **n** is 4, the bit structure will be as:

D10 = H 0123 (b15 … b0)

0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1
0				1				2				3			

Converted to

D20 (b15 … b0)

0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

'0' → H 30

D21 (b15 … b0)

0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1

'1' → H 31

D22 (b15 … b0)

0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0

'2' → H 32

D23 (b15 … b0)

0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1

'3' → H 33

6. When **n** = 1 ~ 16:

D \ n	K1	K2	K3	K4	K5	K6	K7	K8
D20	"3"	"2"	"1"	"0"	"7"	"6"	"5"	"4"
D21		"3"	"2"	"1"	"0"	"7"	"6"	"5"
D22			"3"	"2"	"1"	"0"	"7"	"6"
D23				"3"	"2"	"1"	"0"	"7"
D24					"3"	"2"	"1"	"0"
D25						"3"	"2"	"1"
D26							"3"	"2"
D27								"3"
D28								
D29				No change				
D30								
D31								
D32								
D33								
D34								
D35								

D \ n	K9	K10	K11	K12	K13	K14	K15	K16
D20	“B”	“A”	“9”	“8”	“F”	“E”	“D”	“C”
D21	“4”	“B”	“A”	“9”	“8”	“F”	“E”	“D”
D22	“5”	“4”	“B”	“A”	“9”	“8”	“F”	“E”
D23	“6”	“5”	“4”	“B”	“A”	“9”	“8”	“F”
D24	“7”	“6”	“5”	“4”	“B”	“A”	“9”	“8”
D25	“0”	“7”	“6”	“5”	“4”	“B”	“A”	“9”
D26	“1”	“0”	“7”	“6”	“5”	“4”	“B”	“A”
D27	“2”	“1”	“0”	“7”	“6”	“5”	“4”	“B”
D28	“3”	“2”	“1”	“0”	“7”	“6”	“5”	“4”
D29		“3”	“2”	“1”	“0”	“7”	“6”	“5”
D30			“3”	“2”	“1”	“0”	“7”	“6”
D31				“3”	“2”	“1”	“0”	“7”
D32					“3”	“2”	“1”	“0”
D33			No change			“3”	“2”	“1”
D34							“3”	“2”
D35								“3”

API	Mnemonic			Operands	Function	Controllers
83		HEX	P	(S) (D) (n)	Convert ASCII to HEX	ES2/EX2/EC5, SS2, SA2 SE, SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S					*	*	*	*	*	*	*	*	*			HEX, HEXP: 7 steps
D								*	*	*	*	*	*			
n					*	*										

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device **D**: Destination device **n**: number of bytes to be converted (**n** = 1~256)

Explanations:

1. 16-bit conversion mode: When M1161 = OFF, the instruction converts **n** bytes of ASCII codes starting from **S** into Hex data in byte mode and send them to high byte and low byte of **D**. **n** = the converted number of bytes.
2. 8-bit conversion mode: When M1161 = ON, the instruction converts **n** bytes (low bytes only) of ASCII codes starting from **S** into Hex data in byte mode and send them to the low byte of **D**. **n** = the converted number of bytes. (All higher 8 bits of **D** = 0)
3. If the ASCII code is not in the range of H30~H39 (0~9) or is not in the range H41~H46 (A~F), HEX will set M1067, and the conversion of the ASCII code into a hexadecimal value will stop.

Program Example 1:

1. M1161 = OFF: 16-bit conversion.
2. When X0 = ON, convert 4 bytes of ASCII codes stored in registers D20~ D21 into Hex value and send the result in byte mode to register D10. **n** = 4

3. Assume:

S	ASCII code	HEX conversion	S	ASCII code	HEX conversion
D20 low byte	H 43	“C”	D24 low byte	H 34	“4”
D20 high byte	H 44	“D”	D24 high byte	H 35	“5”
D21 low byte	H 45	“E”	D25 low byte	H 36	“6”
D21 high byte	H 46	“F”	D25 high byte	H 37	“7”
D22 low byte	H 38	“8”	D26 low byte	H 30	“0”

S	ASCII code	HEX conversion	S	ASCII code	HEX conversion
D22 high byte	H 39	“9”	D26 high byte	H 31	“1”
D23 low byte	H 41	“A”	D27 low byte	H 32	“2”
D23 high byte	H 42	“B”	D27 high byte	H 33	“3”

4. When **n** = 4, the bit structure will be as:

D20	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	1
	44H → ‘D’								43H → ‘C’							

D21	0	1	0	0	0	1	1	0	0	1	0	0	0	1	0	1
	46H → ‘F’								45H → ‘E’							

D10	1	1	0	0	1	1	0	1	1	1	1	0	1	1	1	1
	C				D				E				F			

5. When **n** = 1 ~ 16:

n \ D	D13	D12	D11	D10
1	The undesignated parts in the registers in use are all 0.			***C H
2				**CD H
3				*CDE H
4				CDEF H
5			***C H	DEF8 H
6			**CD H	EF89 H
7			*CDE H	F89A H
8			CDEF H	89AB H
9		***C H	DEF8 H	9AB4 H
10		**CD H	EF89 H	AB45 H
11		*CDE H	F89A H	B456 H
12		CDEF H	89AB H	4567 H
13	***C H	DEF8 H	9AB4 H	5670 H
14	**CD H	EF89 H	AB45 H	6701 H
15	*CDE H	F89A H	B456 H	7012 H
16	CDEF H	89AB H	4567 H	0123 H

Program Example 2:

1. M1161 = ON: 8-bit conversion.

2. Assume:

S	ASCII code	HEX conversion	S	ASCII code	HEX conversion
D20	H 43	"C"	D25	H 39	"9"
D21	H 44	"D"	D26	H 41	"A"
D22	H 45	"E"	D27	H 42	"B"
D23	H 46	"F"	D28	H 34	"4"
D24	H 38	"8"	D29	H 35	"5"
D30	H 36	"6"	D33	H 31	"1"
D31	H 37	"7"	D34	H 32	"2"
D32	H 30	"0"	D35	H 33	"3"

3. When **n** is 2, the bit structure will be as

D20									0	1	0	0	0	0	1	1
									43H → "C"							
D21									0	1	0	0	0	1	0	0
									44H → "D"							
D10	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	1
									C				D			

4. When **n** = 1 to 16:

D / n	D13	D12	D11	D10
1	The used registers which are not specified are all 0			***C H
2				**CD H
3				*CDE H
4				CDEF H
5			***C H	DEF8 H
6			**CD H	EF89 H
7			*CDE H	F89A H
8			CDEF H	89AB H
9		***C H	DEF8 H	9AB4 H

10		**CD H	EF89 H	AB45 H
11		*CDE H	F89A H	B456 H
12		CDEF H	89AB H	4567 H
13	***C H	DEF8 H	9AB4 H	5670 H
14	**CD H	EF89 H	AB45 H	6701 H
15	*CDE H	F89A H	B456 H	7012 H
16	CDEF H	89AB H	4567 H	0123 H

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
84		CCD	P	S D n	Check Code				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	CCD, CCDP: 7 steps
S							*	*	*	*	*	*	*			
D									*	*	*	*	*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

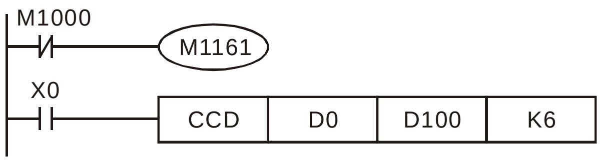
S: source data **D**: Destination device for storing check sum **n**: Number of byte (**n** = 1~256)

Explanations:

1. This instruction performs a sum check for ensuring the validity of the communication data.
2. 16-bit conversion: If M1161 = OFF, **n** bytes of data starting from low byte of **S** will be summed up, the checksum is stored in **D** and the parity bits are stored in **D+1**.
3. 8-bit conversion: If M1161 = ON, **n** bytes of data starting from low byte of **S** (only low byte is valid) will be summed up, the check sum is stored in **D** and the parity bits are stored in **D+1**.

Program Example 1:

1. M1161 = OFF, 16-bit conversion.
2. When X0 = ON, 6 bytes from low byte of D0 to high byte of D2 will be summed up, and the checksum is stored in D100 while the parity bits are stored in D101.



(S)	Content of data
D0 low byte	K100 = 0 1 1 0 0 1 0 0
D0 high byte	K111 = 0 1 1 0 1 1 1 1
D1 low byte	K120 = 0 1 1 1 1 0 0 0
D1 high byte	K202 = 1 1 0 0 1 0 1 0
D2 low byte	K123 = 0 1 1 1 1 0 1 1
D2 high byte	K211 = 1 1 0 1 0 0 1 1
D100	K867
D101	0 0 0 1 0 0 0 1

Total

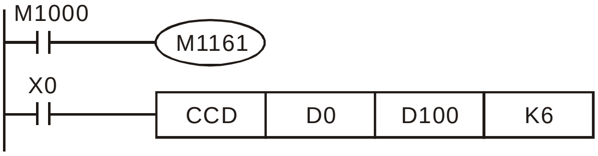
The parity is 1 when there is an odd number of 1.
The parity is 0 when there is an even number of 1.

D100	0	0	0	0	0	0	1	1	0	1	1	0	0	0	1	1
D101	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

⟸ Parity

Program Example 2:

1. M1161 = ON, 8-bit conversion.
2. When X0 = ON, 6 bytes from low byte of D0 to low byte of D5 will be summed up, and the checksum is stored in D100 while the parity bits are stored in D101.



(S)	Content of data
D0 low byte	K100 = 0 1 1 0 0 1 0 0
D1 low byte	K111 = 0 1 1 0 1 1 1 1
D2 low byte	K120 = 0 1 1 1 1 0 0 0
D3 low byte	K202 = 1 1 0 0 1 0 1 0
D4 low byte	K123 = 0 1 1 1 1 0 1 1
D5 low byte	K211 = 1 1 0 1 0 0 1 1
D100	K867
D101	0 0 0 1 0 0 0 1

Total

The parity is 1 when there is a odd number of 1.
The parity is 0 when there is a even number of 1.

D100	0	0	0	0	0	0	1	1	0	1	1	0	0	0	1	1
D101	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

⇐ Parity

API	Mnemonic		Operands	Function	Controllers				
					ES2/ EX2/ EC5	SS2	SA2	SX2	SE
85	VRRD	P	(S) (D)	Volume Read					

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	VRRD, VRRDP: 5 steps
S					*	*										
D								*	*	*	*	*	*			

PULSE					16-bit					32-bit				
ES2/ EX2/ EC5	SS2	SA2	SX2	SE	ES2/ EX2/ EC5	SS2	SA2	SX2	SE	ES2/ EX2/ EC5	SS2	SA2	SX2	SE

Operands:

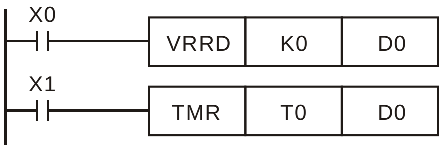
S: Variable resistor number (0~1) **D**: Destination device for storing read value

Explanations:

1. VRRD instruction is used to read the two variable resistors on PLC. The read value will be converted as 0 ~ 255 and stored in destination **D**.
2. If the VR volume is used as the set value of timer, the user only has to turn the VR knob and the set value of timer can be adjusted. When a value bigger than 255 is required, plus D with a certain constant.
3. Flags: M1178 and M1179. (See the Note)

Program Example:

1. When X0 = ON, the value of VR No.0 will be read out, converted into 8-bit BIN value (0~255), and stored in D0.
2. When X1 = ON, the timer which applies D0 as the set value will start timing.



Points to Note:

1. VR denotes Variable Resistor.
2. The PLC supports built-in 2 points of VR knobs which can be used with special D and M.

Device	Function
M1178	Enable knob VR0
M1179	Enable knob VR1
D1178	VR0 value
D1179	VR1 value

API	Mnemonic		Operands	Function	Controllers				
					ES2/ EX2/ EC5	SS2	SA2	SX2	SE
86	VRSC	P	(S) (D)	Volume Scale Read					

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	VRSC, VRSCP: 5 steps
S					*	*										
D								*	*	*	*	*	*			

PULSE					16-bit					32-bit				
ES2/ EX2/ EC5	SS2	SA2	SX2	SE	ES2/ EX2/ EC5	SS2	SA2	SX2	SE	ES2/ EX2/ EC5	SS2	SA2	SX2	SE

Operands:

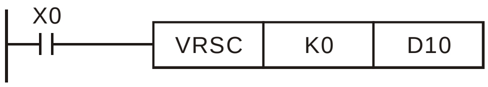
S: Variable resistor number (0~1) **D**: Destination device for storing scaled value

Explanations:

VRSC instruction reads the scaled value (0~10) of the 2 VRs on PLC and stores the read data in destination device **D** as an integer, i.e. if the value is between 2 graduations, the value will be rounded off.

Program Example 1:

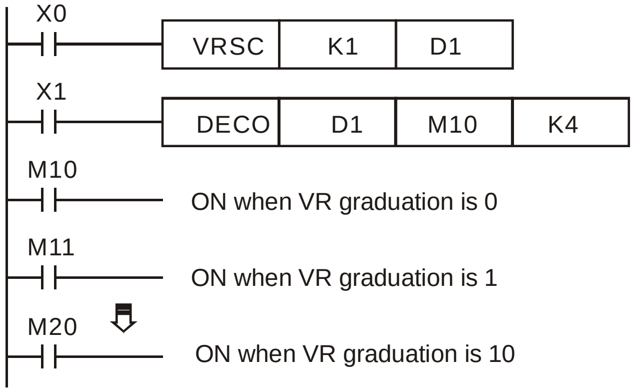
When X0 = ON, VRSC instruction reads the scaled value (0 to10) of VR No. 0 and stores the read value in device D10.



Program Example 2:

Apply the VR as digital switch: The graduations 0~10 of VR correspond to M10~M20, therefore only one of M10 ~M20 will be ON at a time. When M10~M20 is ON, use DECO instruction (API 41) to decode the scaled value into M10~M25.

1. When X0 = ON, the graduation (0~10) of VR No.1 will be read out and stored in D1.
2. When X1 = ON, DECO instruction will decode the graduation (0~10) into M10~M25.


X0
VRSC
K1
D1
X1
DECO
D1
M10
K4
M10
ON when VR graduation is 0
M11
ON when VR graduation is 1
M20
ON when VR graduation is 10

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
87	D	ABS	P	(D)	Absolute Value				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ABS, ABSP: 3 steps
D								*	*	*	*	*	*	*	*	DABS, DABSP: 5 steps

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

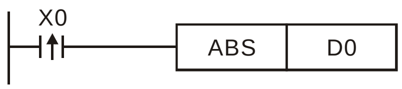
D: Device for absolute value operation

Explanation

1. The instruction conducts absolute value operation on **D**
2. This instruction is generally used in pulse execution mode (ABSP, DABSP).
3. If operand **D** uses index F, then only 16-bit instruction is available.
4. When it comes to the signed minimum value -32768, whether it is in 16-bit or 32-bit format, there is no absolute value. The signed minimum value stays -32768.

Program Example:

When X0 goes from OFF to ON, ABS instruction obtains the absolute value of the content in D0. Before the execution of this instruction, the value in D0 is -1234, and after the execution of this instruction, the value in D0 is 1234.



API	Mnemonic		Operands	Function	Controllers			
88	D	PID	S1 S2 S3 D	PID control	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1													*			PID : 9 steps DPID: 17 steps
S_2													*			
S_3													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

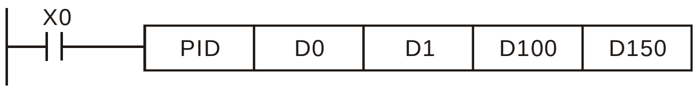
S_1: Set value (SV) **S_2**: Present value (PV) **S_3**: Parameter setting (for 16-bit instruction, uses 20 consecutive devices, for 32-bit instruction, uses 21 consecutive devices) **D**: Output value (MV)

Explanations:

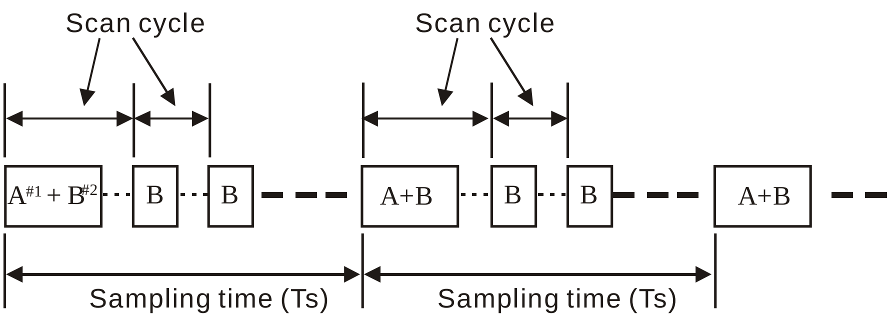
1. This instruction is specifically for PID control. PID operation will be executed only when the sampling time is reached. PID refers to “proportion, integration and derivative”. PID control is widely applied to many mechanical, pneumatic and electronic equipment.
2. After all the parameters are set up, PID instruction can be executed and the results will be stored in **D**. **D** has to be unlatched data register. (If users want to designate a latched data register area, please clear the latched registers to 0 in the beginning of user program.

Program Example:

1. Complete the parameter setting before executing PID instruction.
2. When X0 = ON, the instruction will be executed and the result will be stored in D150. When X0 = OFF, the instruction will not be executed and the previous data in D150 will stay intact.



3. Timing chart of the PID operation (max. operation time is approx. 80us)



Note: #1→ The time for equation calculation during PID operation (approx. 72us)
#2→ The PID operation time without equation calculation (approx. 8us)

Points to note:

1. There is no limitation on the times of using this instruction. However, the register No. designated in $\mathbf{S_3}$~ $\mathbf{S_3}$+19 cannot be repeated.
2. For 16-bit instruction, $\mathbf{S_3}$ occupies 20 registers. In the program example above, the area designated in $\mathbf{S_3}$ is D100 ~ D119.
3. Before the execution of PID instruction, users have to transmit the parameters to the designated register area by MOV instruction. If the designated registers are latched, use MOVP instruction to transmit all parameters only once
4. Settings of $\mathbf{S_3}$ in the 16-bit instruction:

Device No.	Function	Setup Range	Explanation
$\mathbf{S_3}$:	Sampling time (T_S)	1~2,000 (unit: 10ms)	Time interval between PID calculations and updates of MV. If T_S = 0, PID instruction will not be enabled. If T_S is less than 1 program scan time, PID instruction sets $\mathbf{S_3}$ as 1 program scan time, i.e. the minimum T_S has to be longer than the program scan time.
$\mathbf{S_3}$+1:	Propotional gain (K_P)	0~30,000(%)	The proportion for magnifying/minifying the error between SV and PV.
$\mathbf{S_3}$+2:	Integral gain (K_I)	0~30,000(%)	The proportion for magnifying/minifying the integral value (The accumulated error). For control mode K0~K8.
	Integral time constant (T_I)	0~30,000 (ms)	For control mode K10
$\mathbf{S_3}$+3:	Derivative gain (K_D)	-30,000~30,000 (%)	The proportion for magnifying/minifying the derivative value (The rate of change of the process error). For control mode K0~K8
	Derivative time constant (T_D)	-30,000~30,000 (ms)	For control mode K10

Device No.	Function	Setup Range	Explanation
S_3+4:	Control mode	0: Automatic control 1: Forward control (E = SV - PV). 2: Reverse control (E = PV - SV). 3: Auto-tuning of parameter exclusively for the temperature control. The device will automatically become K4 when the auto-tuning is completed and K_P, K_I and K_D is set with appropriate value (not avaliable in the 32-bit instruction). 4: Exclusively for the adjusted temperature control (not avaliable in the 32-bit instruction). 5: Automatic mode with MV upper/lower bound control. When MV reaches upper/lower bound, the accumulation of integral value stops. 7: Manual control 1: User set an MV. The accumulated integral value increases according to the error. It is suggested that the control mode should be used in a control environment which change more slowly. DVP-ES2/DVP-EX2/DVP-SS2/DVP-SA2/DVP-SX2 series PLCs whose version is 2.00 (or above), and DVP-SE series PLCs whose version is 1.00 (or above) are supported. 8: Manual control 2: User set an MV. The accumulated integral value will stop increasing. When the control mode becomes the automatic mode (the control mode K5 is used), the instruction PID outputs an appropriate accumulated integral value according to the last MV. DVP-ES2/DVP-EX2/DVP-SS2/DVP-SA2/DVP-SX2 series PLCs whose version is 2.00 (or above), and DVP-SE series PLCs whose version is 1.00 (or above) are supported. 9: When in Automatic mode, if the MV exceeds the upper/lower limits, MV calculates according to upper/lower limits to calculate the accumulated integral calculus. 10: TI / TD mode: The control changes the integra gain and the differential gain into integral time constant and differential time constant.	
S_3+5:	Tolerable range for error (E)	0~32,767	E = the error between SV and PV. If **S_3** +5 is set as 5, when E is between -5 and 5, E will be 0. When **S_3** +5 = K0, the function will not be enabled.
S_3+6:	Upper bound of output value (MV)	-32,768~32,767	Ex: if **S_3**+6 is set as 1,000, MV will be 1,000 when it exceeds 1,000. **S_3**+6 has to be bigger or equal to **S_3**+7, otherwise the upper bound and lower bound value will switch.
S_3+7:	Lower bound of output value (MV)	-32,768~32,767	Ex: if **S_3**+7 is set as -1,000, MV will be -1,000 when it is smaller than -1,000..
S_3+8:	Upper bound of integral value	-32,768~32,767	Ex: if **S_3**+8 is set as 1,000, the integral value will be 1,000 when it is bigger than 1,000 and the integration will stop. **S_3**+8 has to be bigger or equal **S_3** +9; otherwise the upper bound and lower bound value will switch

Device No.	Function	Setup Range	Explanation
S_3+9:	Lower bound of integral value	-32,768~32,767	Ex: if **S_3**+9 is set as -1,000, the integral value will be -1,000 when it is smaller than -1,000 and the integration will stop.
S_3+10, 11:	Accumulated integral value	Available range of 32-bit floating point	The accumulated integral value is usually for reference. Users can clear or modify it (in 32-bit floating point) according to specific needs.
S_3 +12:	The previous PV	-32,768~32,767	The previous PV is usually for reference. Users can clear or modify it according to specific needs.
S_3+13 ~ **S_3**+19	For system use only..		

5. For **S_3**+1~3, when parameter setting exceeds its range, the upper / lower bound will be selected as the set value.
6. If the direction setting (Forward / Reverse) exceeds its range, it will be set to 0.
7. PID instruction can be used in interruption subroutines, step ladders and CJ instruction.
8. The maximum error of sampling time T_S = **-** (1 scan time + 1ms) ~ + (1 scan time). When the error affects the output, please fix the scan time or execute PID instruction in timer interrupt.
9. PV of PID instruction has to be stable before PID operation executes. If users need to take the value input from AIO modules for PID operation, care should be taken on the A/D conversion time of these modules
10. For 32-bit instruction, **S_3** occupies 21 registers. In the program example above, the area designated in **S_3** will be D100 ~ D120. Before the execution of PID instruction, users have to transmit the parameters to the designated register area by MOV instruction. If the designated registers are latched, use MOVP instruction to transmit all parameters only once.
11. Parameter table of 32-bit **S_3**:

Device No.	Function	Set-point range	Explanation
S_3	Sampling time (T_S)	1~2,000 (unit: 10ms)	Time interval between PID calculations and updates of MV. If T_S= 0, PID instruction will not be enabled. If T_S is less than 1 program scan time, PID instruction sets **S_3** as 1 program scan time, i.e. the minimum T_S has to be longer than the program scan time.

Device No.	Function	Set-point range	Explanation
S_3+1	Proportional gain (K_P)	0~30,000 (%)	The proportion for magnifying/minifying the error between SV and PV.
S_3+2	Integration gain (K_I)	0~30,000 (%)	The proportion for magnifying/minifying the integral value (The accumulated error). For control mode K0~K2, K5.
	Integral time constant (T_I)	0~30,000 (ms)	For control mode K10
S_3+3	Derivative gain (K_D)	-30,000~30,000 (%)	The proportion for magnifying/minifying the derivative value (The rate of change of the process error). For control mode K0~K2, K5.
	Derivative time constant (T_D)	-30,000~30,000 (ms)	For control mode K10
S_3+4	Control mode	0: Automatic control 1: Forward control (E = SV - PV). 2: Reverse control (E = PV - SV). 5: Automatic mode with MV upper/lower bound control. When MV reaches upper/lower bound, the accumulation of integral value stops. 10: T_I / T_D mode with MV upper/lower bound control. When MV reaches upper/lower bound, the accumulation of integral value stops.	
S_3+5, 6	Tolerable range for error (E), 32-bit	0~2,147,483,647	E = the error between SV and PV. If S_3 +5 is set as 5, when E is between -5 and 5, E will be 0. When S_3 +5 = K0, the function will not be enabled.
S_3+7, 8	Upper bound of output value (MV) , 32-bit	-2,147,483,648~ 2,147,483,647	Ex: if S_3+6 is set as 1,000, MV will be 1,000 when it exceeds 1,000. S_3+6 has to be bigger or equal to S_3+7, otherwise the upper bound and lower bound value will switch
S_3+9, 10	Lower bound of output value (MV) , 32-bit	-2,147,483,648~ 2,147,483,647	Ex: if S_3+7 is set as -1,000, MV will be -1,000 when it is smaller than -1,000.
S_3+11, 12	Upper bound of integral value, 32-bit	-2,147,483,648~ 2,147,483,647	Ex: if S_3+8 is set as 1,000, the integral value will be 1,000 when it is bigger than 1,000 and the integration will stop. S_3+8 has to be bigger or equal S_3 +9; otherwise the upper bound and lower bound value will switch.
S_3+13, 14	Lower bound of integral value, 32-bit	-2,147,483,648~ 2,147,483,647	Ex: if S_3+9 is set as -1,000, the integral value will be -1,000 when it is smaller than -1,000 and the integration will stop.
S_3+15, 16	Accumulated integral value, 32-bit	Available range of 32-bit floating point	The accumulated integral value is usually for reference. Users can clear or modify it (in 32-bit floating point) according to specific needs.

Device No.	Function	Set-point range	Explanation
S3+17, 18	The previous PV, 32-bit	-2,147,483,648~ 2,147,483,647	The previous PV is usually for reference. Users can clear or modify it according to specific needs.
S3+19, 20	For system use only.		

12. The explanation of 32-bit **S3** and 16-bit **S3** are almost the same. The difference is the capacity of **S3**+5 ~ **S3**+20.

PID Equations:

1. When control mode (**S3**+4) is selected as K0, K1, K2 and K5:
   - In this control mode, PID operation can be selected as Automatic, Forward, Reverse and Automatic with MV upper/lower bound control modes. Forward / Reverse direction is designated in **S3**+4. Other relevant settings of PID operation are set by the registers designated in **S3** ~ **S3**+5.
   - PID equation for control mode k0~k2:

     $$MV = K_P * E(t) + K_I * E(t)\frac{1}{S} + K_D * PV(t)S$$

     where

     MV : Output value

     K_P : Proprotional gain

     $E(t)$: Error value

     $PV\ (t)$: Present measured value

     $SV\ (t)$: Target value

     K_D : Derivative gain

     $PV(t)S$: Derivative value of PV(t)

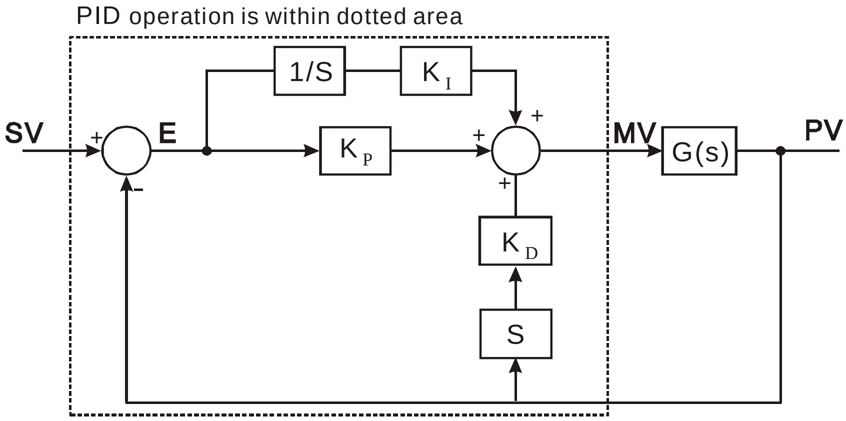
     K_I : Integral gain

     $E(t)\frac{1}{S}$: Integral value of E(t)

   - When E(t) is smaller than 0 as the control mode is selected as forward or inverse, E(t) will be regarded as "0"

Control mode	PID equation
Forward, automatic	E(t) = SV – PV
Inverse	E(t) = PV – SV

   - Control diagram:
     In diagram below, S is derivative operation, referring to "(PV– previous PV) ÷ sampling time". 1 / S is integral operation, referring to "previous integral value + (error value × sampling time)". G(S) refers to the device being controlled.



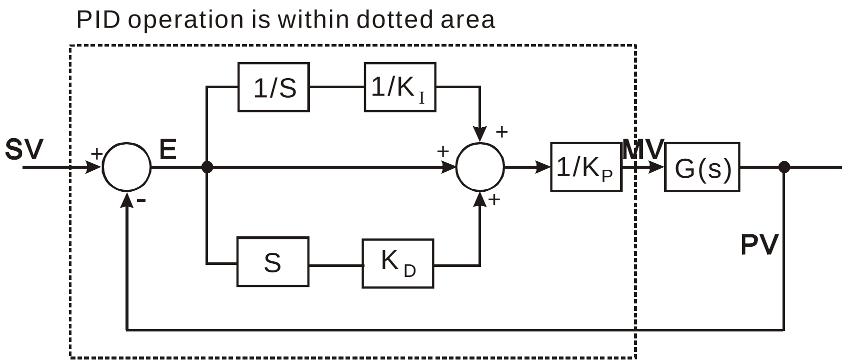
- The equation above illustrates that this operation is different from a general PID operation on the application of the derivative value. To avoid the fault that the transient derivative value could be too big when a general PID instruction is first executed, our PID instruction monitors the derivative value of the PV. When the variation of PV is excessive, the instruction will reduce the output of MV/.

2. When control mode (**S_3**+4) is selected as K3 and K4:
   - The equation is exclusively for temperature control will be modified as:

$$MV = \frac{1}{K_P}\left[E(t) + \frac{1}{K_I}\left(E(t)\frac{1}{S}\right) + K_D * E(t)S\right],$$

where $\mathrm{E}(\mathrm{t}) = \mathrm{SV}(\mathrm{t}) - \mathrm{PV}(\mathrm{t})$

   - Control diagram:
     In diagram below, $1/K_I$ and $1/K_P$ refer to “divided by K_I” and “divided by K_P”. Because this mode is exclusively for temperature control, users have to use PID instruction together with GPWM instruction. See **Application 3** for more details



   - This equation is exclusively designed for temperature control. Therefore, when the sampling time (T_S) is set as 4 seconds (K400), the range of output value (MV) will be K0 ~ K4,000 and the cycle time of GPWM instruction used together has to be set as 4 seconds (K4000) as well.

- If users have no idea on parameter adjustment, select K3 (auto-tuning). After all the parameters are adjusted (the control direction will be automatically set as K4), users can modify the parameters to better ones according to the adjusted results.

3. When control mode (**S**$_3$+4) is selected as K10:
   - **S**$_3$+2 (K_I) and **S**$_3$+3 (K_D) in this mode will be switched to parameter settings of Integral time constant (T_I) and Derivative time constant (T_D).
   - When output value (MV) reaches the upper bound, the accumulated integral value will not increase. Also, when MV reaches the lower bound, the accumulated integral value will not decrease.
   - The equation for this mode will be modified as:

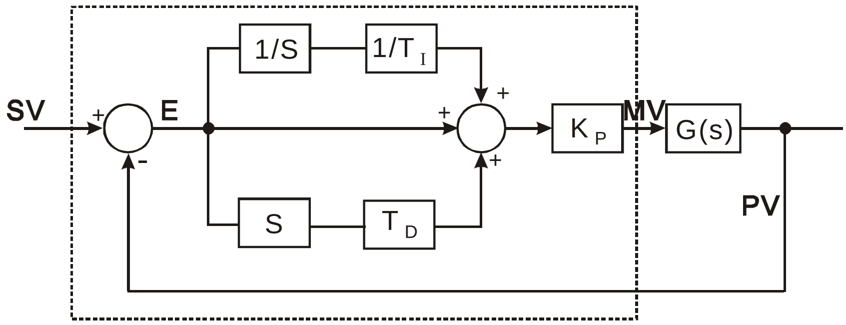
$$MV = K_P \times \left[E(t) + \frac{1}{T_I} \int E(t)dt + T_D \frac{d}{dt} E(t) \right]$$

Where

$$E(t) = SV(t) - PV(t)$$

Control diagram:

PID operation is within dotted area



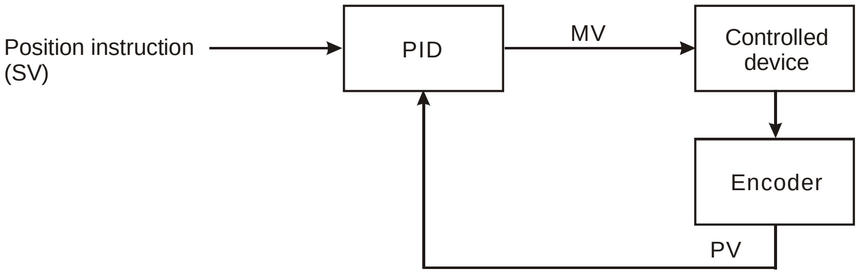
Notes and suggestion:

1. **S**$_3$ + 3 can only be the value within 0 ~ 30,000.
2. There are a lot of circumstances where PID instruction can be applied; therefore, please choose the control functions appropriately. For example, when users select parameter auto-tuning for the temperature (**S**$_3$ + 4 = K3), the instruction can not be used in a motor control environment otherwise improper control may occur.
3. When you adjust the three main parameters, K_P, K_I and K_D (**S**$_3$ + 4 = K0 ~ K2), please adjust K_P first (according to your experiences) and set K_I and K_D as 0. When the output can roughly be controlled, proceed to increase K_I and K_D (see example 4 below for adjustment methods). K_P = 100 refers to 100%, i.e. the proportional gain to the error is 1. K_P < 100% will decrease the error and K_P > 100% will increase the error
4. When temperature auto-tuning function is selected(**S**$_3$ + 4 = K3, K4), it is suggested that store the parameters in D register in latched area in case the adjusted parameters will

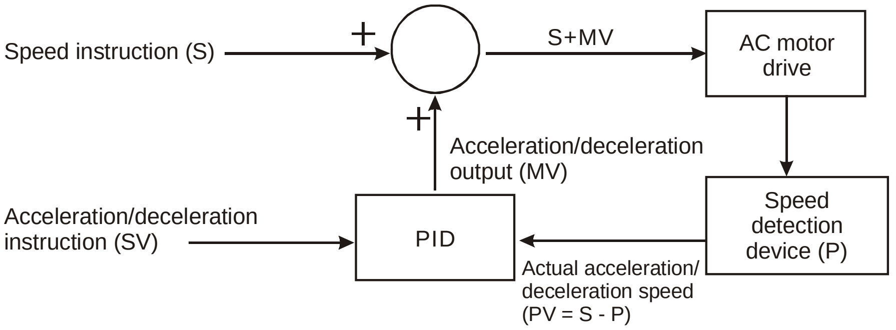
disappear after the power is cut off. There is no guarantee that the adjusted parameters are suitable for every control requirement. Therefore, users can modify the adjusted parameters according to specific needs, but it is suggested to modify only K_I or K_D.

5. PID instruction has to be controlled with many parameters; therefore care should be taken when setting each parameter in case the PID operation is out of control.

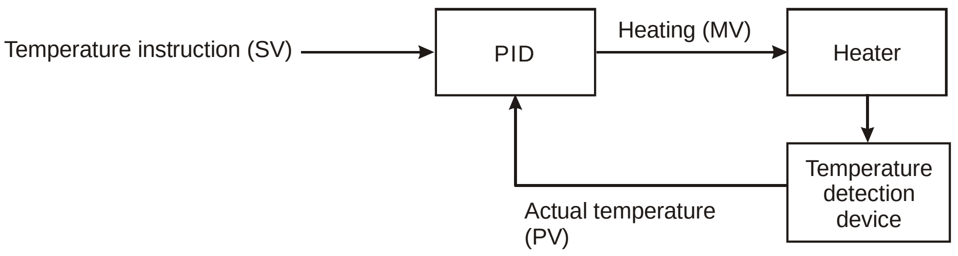
Example 1: Block diagram of application on positioning (**S3**+4 = 0)



Example 2: Block diagram of application on AC motor drive (**S3**+4 = 0)



Example 3: Block diagram of application on temperature control (**S3**+4 = 1)

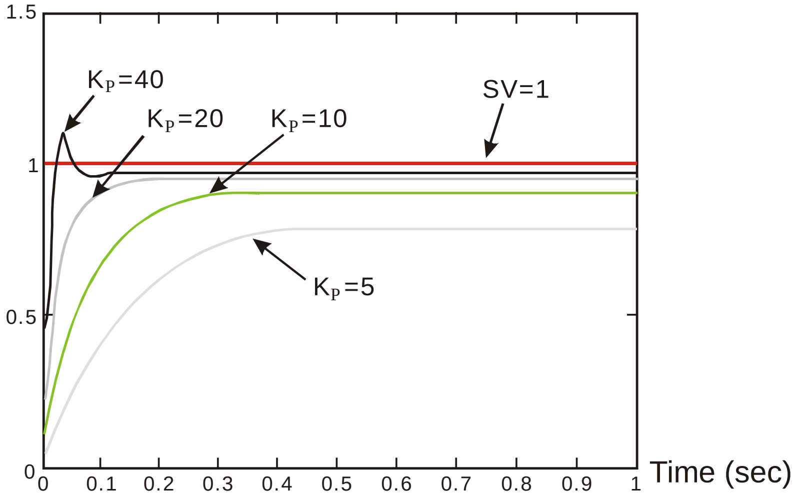


Example 4: PID parameters adjustment

Assume that the transfer function of the controlled device G(S) in a control system is a first-order function $G(s)=\frac{b}{s+a}$ (model of general motors), SV = 1, and sampling time (T_S) = 10ms. Suggested steps for adjusting the parameters are as follows:

Step1:

Set K_I and K_D as 0, and K_P as 5, 10, 20, 40. Record the SV and PV respectively and the results are as the figure below.



Step 2:

When K_P is 40, response overshoot occurs, so we will not select it.

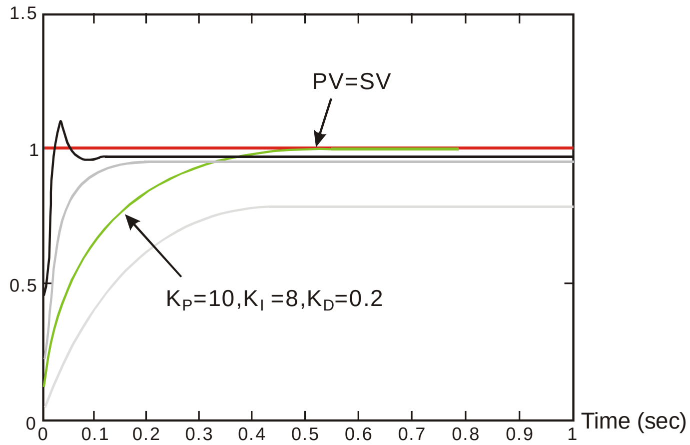
When K_P is 20, PV response is close to SV and won't overshoot, but transient MV will be to large due to a fast start-up. We can put it aside and observe if there are better curves.

When K_P is 10, PV response is close to SV and is smooth. We can consider using it.

When K_P is 5, the response is too slow. So we won't use it.

Step 3:

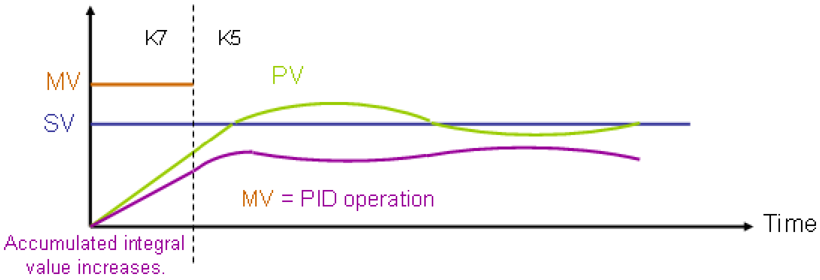
Select K_P = 10 and increase K_I gradually, e.g. 1, 2, 4, 8. K_I should not be bigger than K_P. Then, increase K_D as well, e.g. 0.01, 0.05, 0.1, 0.2. K_D should not exceed 10% of K_P. Finally we obtain the figure of PV and SV below.



Note: The example is only for reference. Users have to adjust parameters according to the condition of the actual control system.

Example 5: Transition between the manual mode (K7) and the automatic mode (K5)

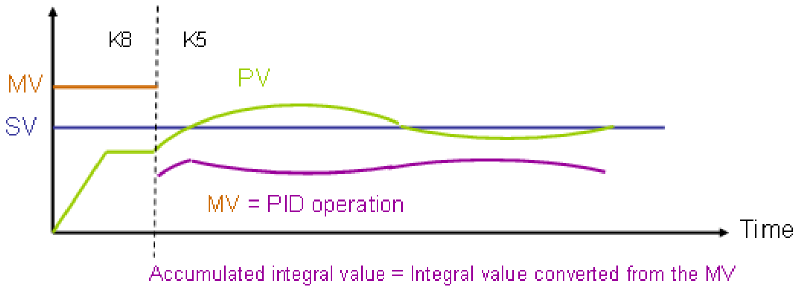
If the setting of the PID parameters is complete, and the control mode is the manual mode (K7), the control curve will be as shown below.



If the control mode becomes the automatic mode (K5), the output value MV changes from the output value set by users to the output value of the PID operation.

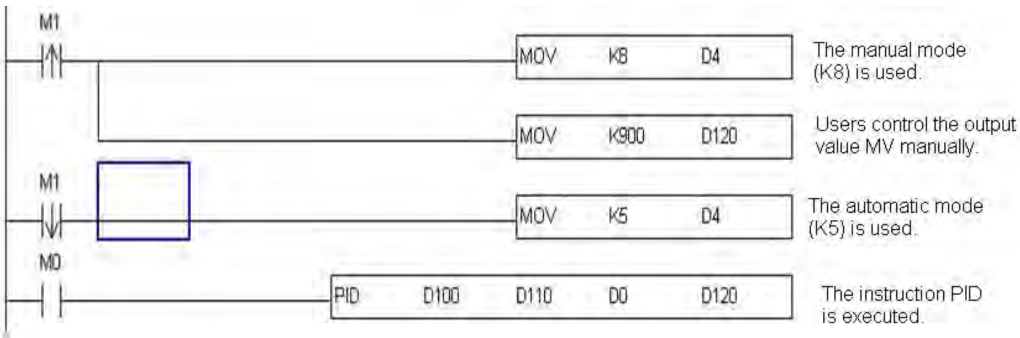
Example 6: Transition between the manual mode (K8) and the automatic mode (K5)

If the setting of the PID parameters is complete, and the control mode is the manual mode (K8), the control curve will be as shown below.



If the control mode becomes the automatic mode (K5), the accumulated integral value will be the integral value converted from the last MV, and the accumulated integral value will be converted into the output value of the PID operation.

The program for example 5 and program 6 are shown below. In the figure below, ,M0 is a flag for enabling the instruction PID. When M1 is On, the manual mode is used. When M1 is Off, the automatic mode is used.



Application 1:

PID instruction in pressure control system. (Use block diagram of example 1)

Control purpose:

Enabling the control system to reach the target pressure.

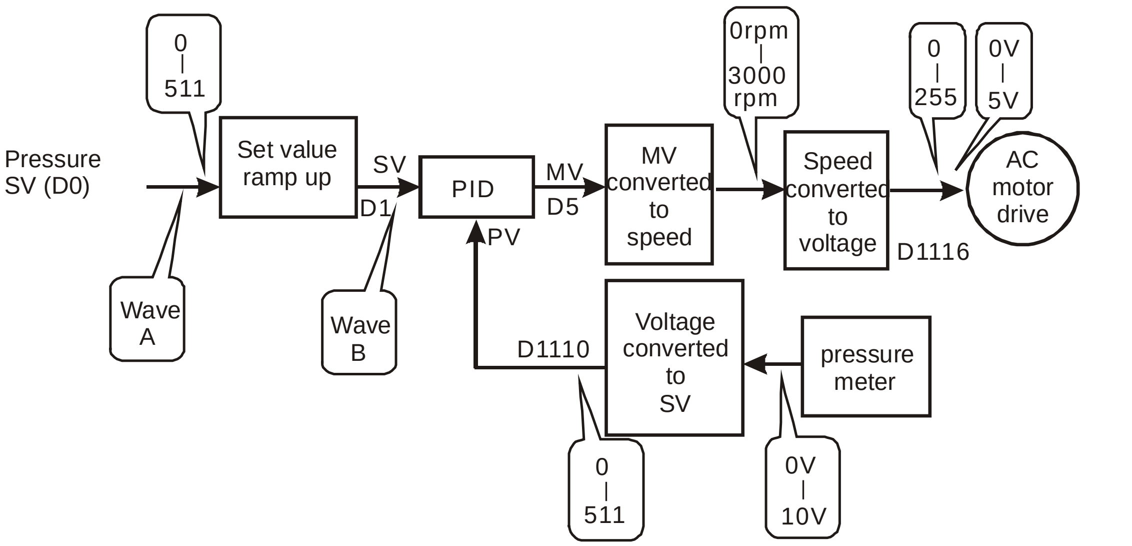
Control properties:

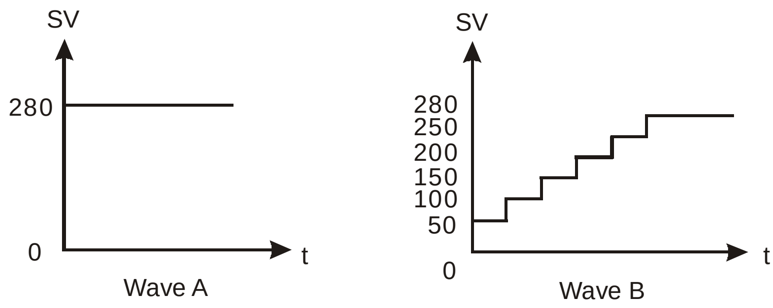
The system requires a gradual control. Therefore, the system will be overloaded or out of control if the process progresses too fast.

Suggested solution:

Solution 1: Longer sampling time

Solution 2: Using delay instruction. See the figure below

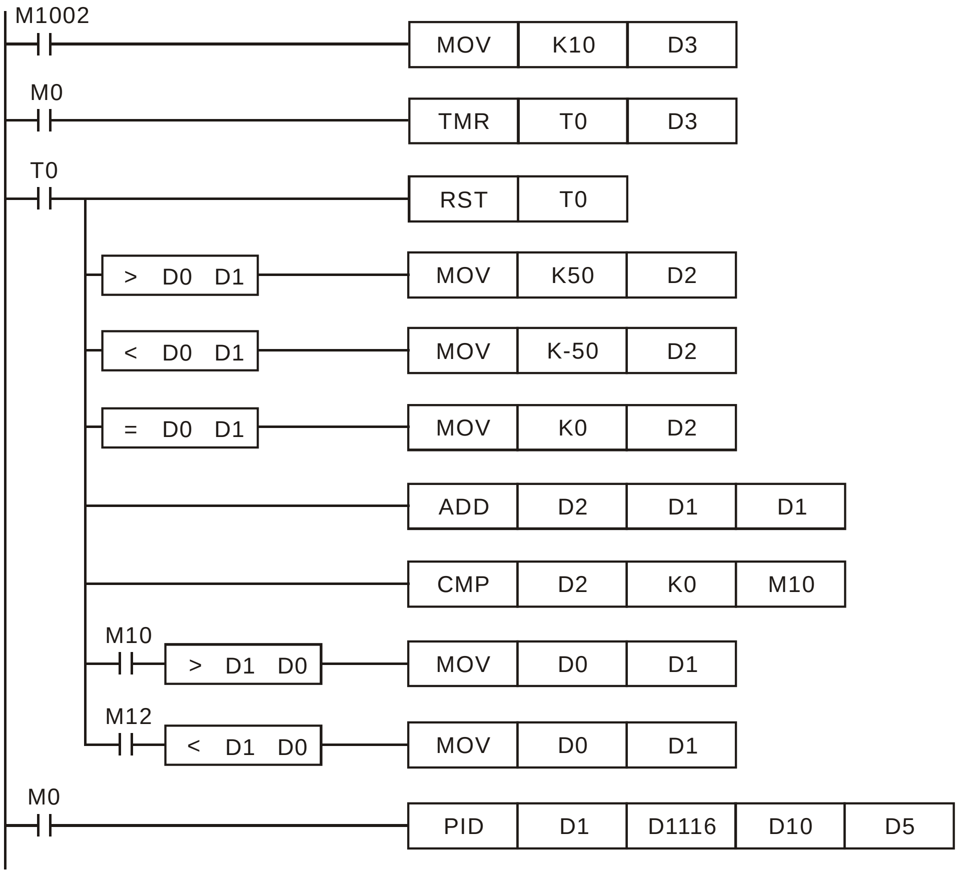




D2 stores increased value of each shift
D3 stores the time interval of each shift

Values in can modify D2 and D3 according to actual requirement

Example program of SV ramp up function:



Application 2:

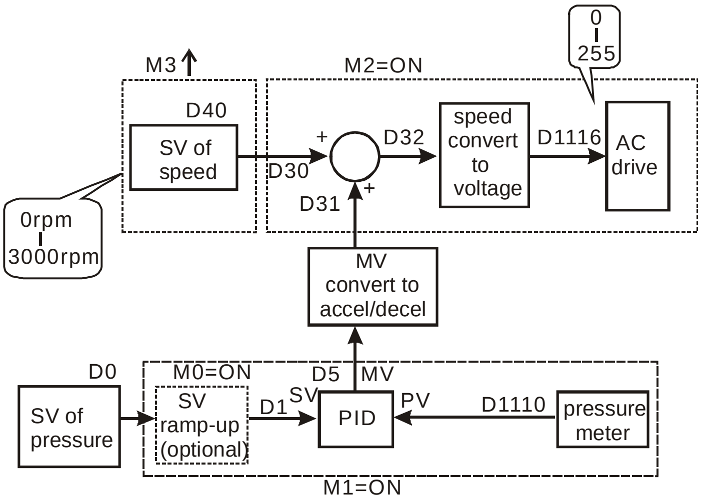
Speed control system and pressure control system work individually (use diagram of Example 2)

<u>Control purpose:</u>

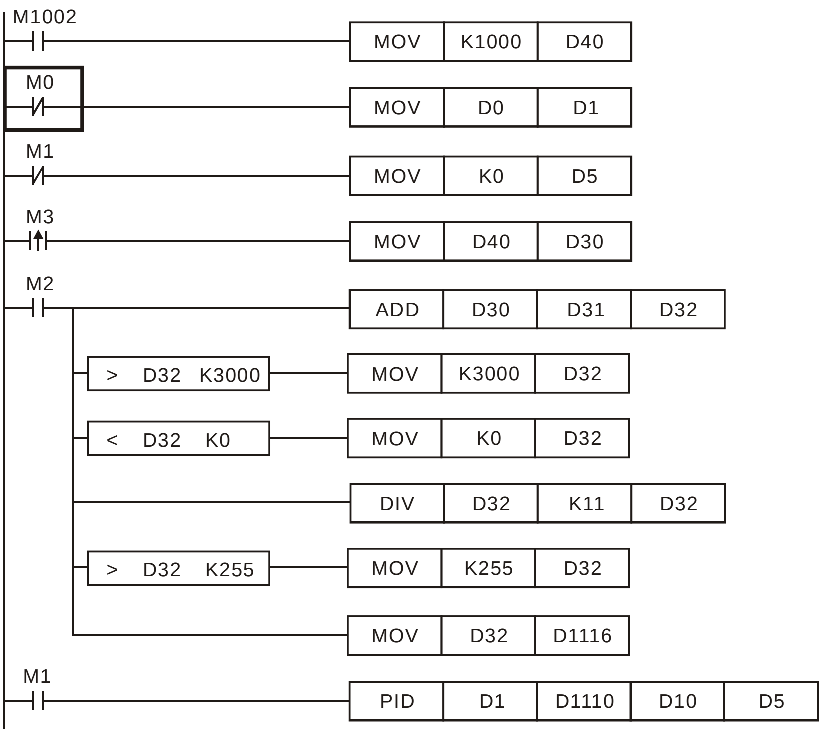
After the speed control operates in open loop for a period of time, adding pressure control system (PID instruction) to perform a close loop control.

<u>Control properties:</u>

Since the speed and pressure control systems are not interrelated, we have to structure an open loop for speed control first following by a close loop pressure control. If users afraid that the pressure control system changes excessively, consider adding the SC ramp-up function illustrated in **Application 1** into this control. See the control diagram below.



Part of the example program:



Application 3:

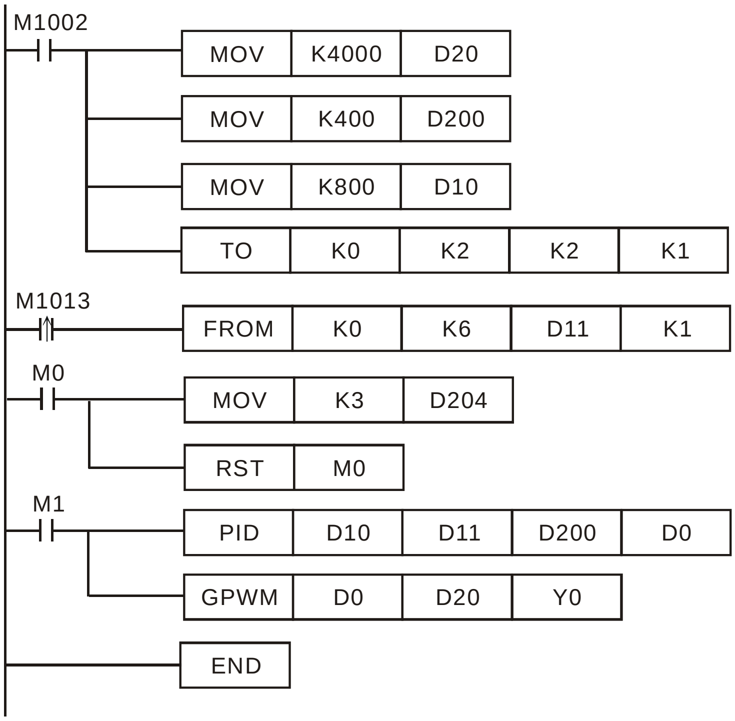
Using auto-tuning for temperature control

<u>Control purpose:</u>

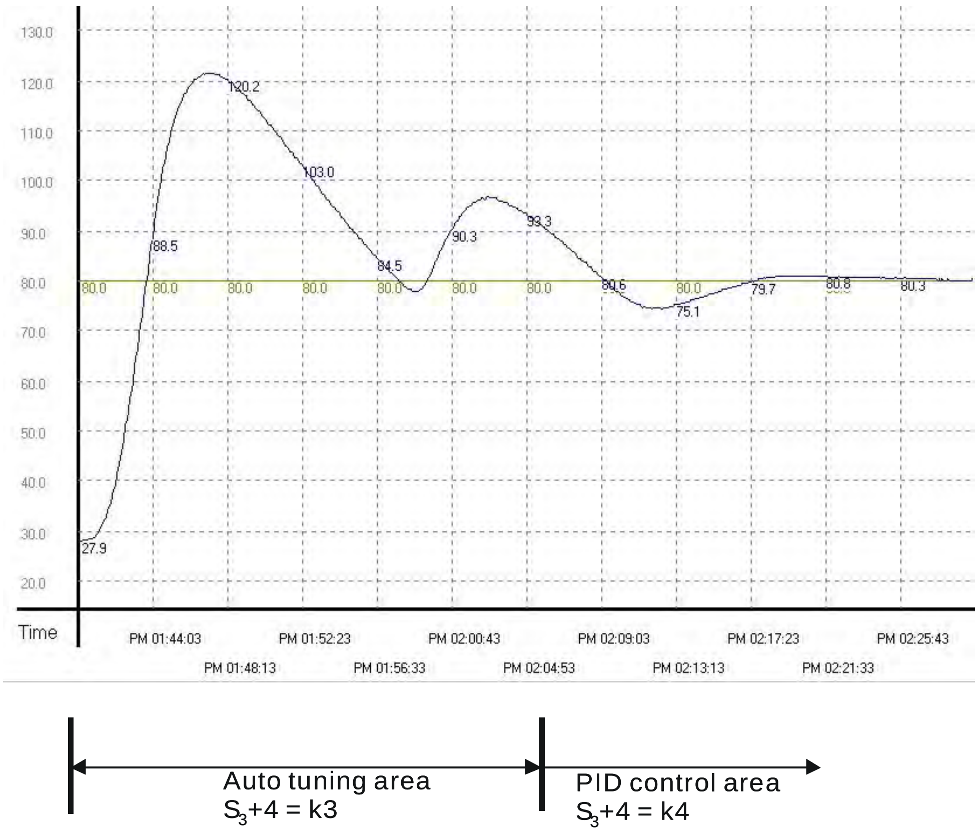
Calculating optimal parameter of PID instruction for temperature control

<u>Control properties:</u>

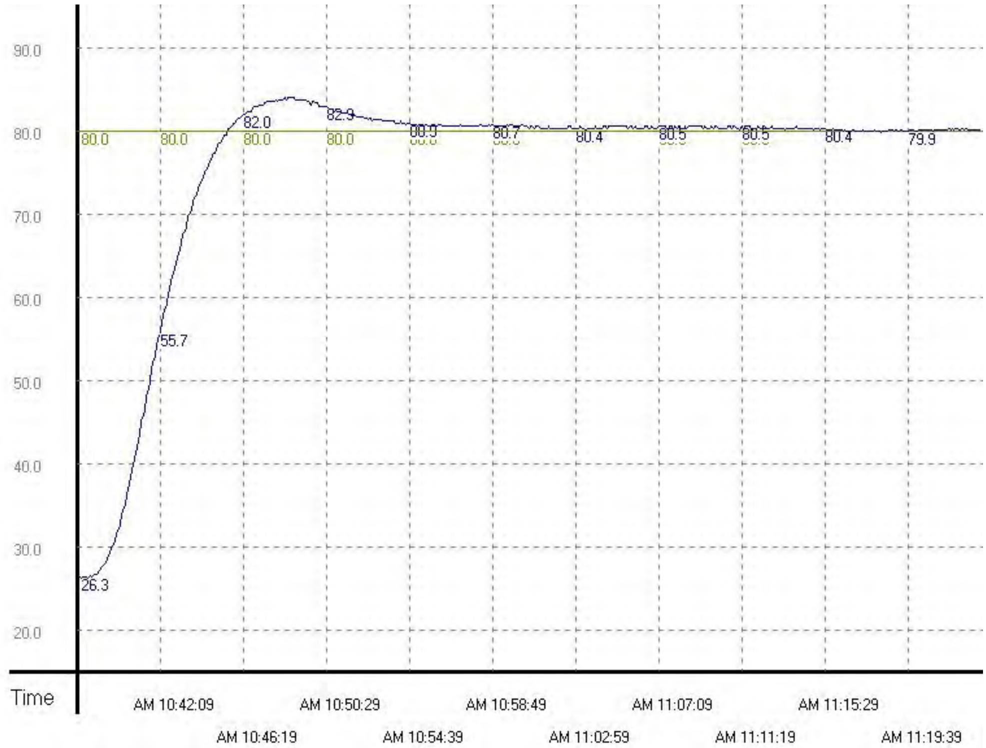
Users may not be familiar with a new temperature environment. In this case, selecting auto-tuning (**S**3+4 = K3) for an initial adjustment is suggested. After initial tuning is completed, the instruction will auto modify control mode to the mode exclusively for adjusted temperature (**S**3+4 = K4). In this example, the control environment is a heating oven. See the example program below.



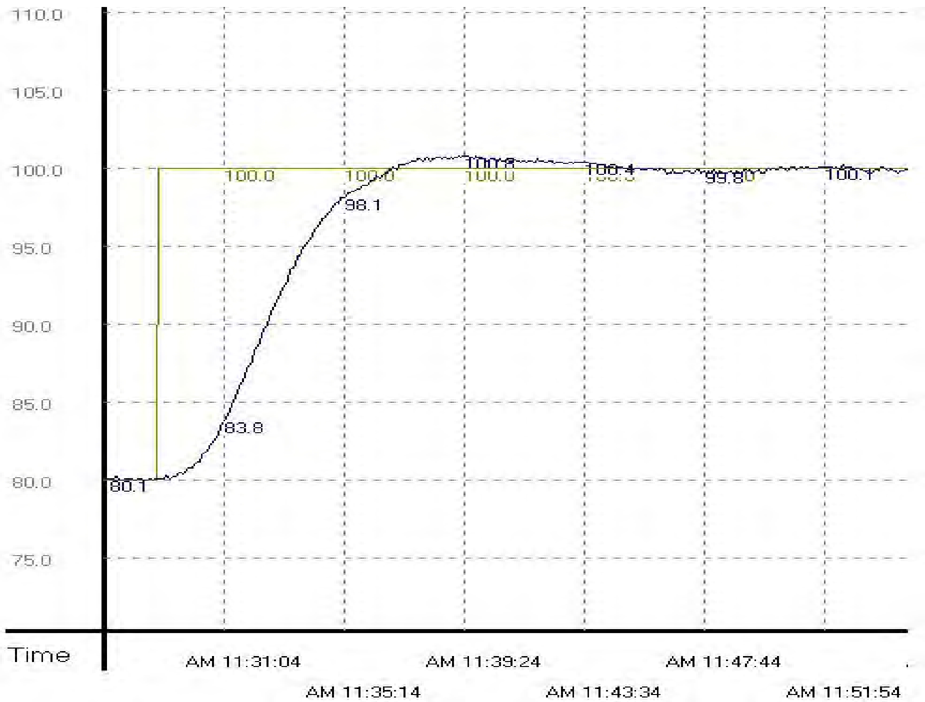
Results of initial auto-tuning



Results of using adjusted parameters generated by initial auto-tuning function.



From the figure above, we can see that the temperature control after auto-tuning is working fine and it spent only approximately 20 minutes for the control. Next, we modify the target temperature from 80°C to 100°C and obtain the result below.



From the result above, we can see that when the parameter is 100°C, temperature control works fine and costs only 20 minutes same as that in 80°C.

3.6.10 Basic Instructions

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
89	PLS	-	-	Rising-edge output	✓	✓	✓	✓	3	-
90	LDP	-	-	Rising–edge detection operation	✓	✓	✓	✓	3	-
91	LDF	-	-	Falling–edge detection operation	✓	✓	✓	✓	3	-
92	ANDP	-	-	Rising-edge series connection	✓	✓	✓	✓	3	-
93	ANDF	-	-	Falling-edge series connection	✓	✓	✓	✓	3	-
94	ORP	-	-	Rising-edge parallel connection	✓	✓	✓	✓	3	-
95	ORF	-	-	Falling-edge parallel connection	✓	✓	✓	✓	3	-
96	TMR	-	-	Timer	✓	✓	✓	✓	4	-
97	CNT	DCNT	-	Counter	✓	✓	✓	✓	4	6
98	INV	-	-	Inverse operation	✓	✓	✓	✓	1	-
99	PLF	-	-	Falling-edge output	✓	✓	✓	✓	3	-
258	ATMR	-	-	Contact type timer	✓	✓	✓	✓	5	-

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
89	PLS	S	Rising-edge output				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	PLS: 3 steps
S		*	*													

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

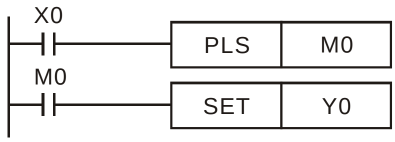
S: Rising pulse output device

Explanations:

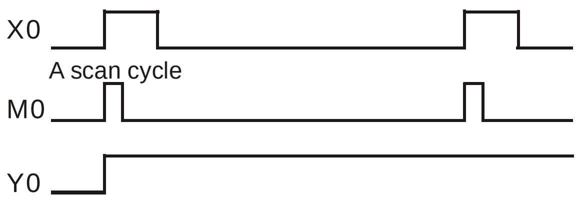
When X0 goes from OFF to ON (Rising-edge trigger), PLS instruction executes and **S** generates a cycle pulse for one operation cycle.

Program Example:

Ladder Diagram:



Timing Diagram:



Instruction Code:		Operation:
LD	X0	; Load NO contact of X0
PLS	**M0**	; M0 rising-edge output
LD	M0	; Load NO contact of M0
SET	Y0	; Y0 latched (ON)

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
90	LDP	(S)	Rising–edge detection operation				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	LDP: 3 steps
S	*	*	*	*							*	*				

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: device to be rising-edge triggered

Explanations:

LDP should be connected to the left side bus line. When the associated device **S** is driven from OFF to ON, LDP will be ON for one scan cycle.

Program Example:

Ladder Diagram:

Instruction Code:		Operation:
LDP	**X0**	; Load rising-edge contact X0
AND	X1	; Connect NO contact X1 in series
OUT	Y1	; Drive Y1 coil

Points to Note:

If the associated rising-edge contact is ON before PLC is power on, the contact will be activated after PLC is power on.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
91	LDF	(S)	Falling–edge detection operation				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	LDF: 3 steps
S	*	*	*	*							*	*				

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: device to be falling pulse triggered

Explanations:

LDF should be connected to the left side bus line. When the associated device **S** is driven from ON to OFF, LDF will be ON for one scan cycle.

Program Example:

Ladder Diagram:

Instruction Code:		Operation:
LDF	**X0**	; Load falling-edge contact X0
AND	X1	; Connect NO contact X1 in series.
OUT	Y1	; Drive Y1 coil

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
92	ANDP	S	Rising-edge series connection				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ANDP: 3 steps
S	*	*	*	*							*	*				

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: rising-edge contact to be connected in series

Explanations:

ANDP instruction is used in the series connection of the rising-edge contact.

Program Example:

Ladder Diagram:

Instruction Code:		Operation:
LD	X0	; Load NO contact of X0
ANDP	**X1**	; X1 rising-edge contact in series connection
OUT	Y1	; Drive Y1 coil

API	Mnemonic	Operands	Function	Controllers			
93	ANDF	S	Falling-edge series connection	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ANDF: 3 steps
S	*	*	*	*							*	*				

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: falling edge contact to be connected in series

Explanations:

ANDF instruction is used in the series connection of the falling-edge contact.

Program Example:

Ladder Diagram:

Instruction Code:		Operation:
LD	X0	; Load NO contact of X0
ANDF	**X1**	; X1 falling-edge contact in series connection
OUT	Y1	; Drive Y1 coil

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
94	ORP	S	Rising-edge parallel connection				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ORP: 3 steps
S	*	*	*	*							*	*				

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

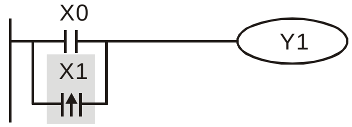
S: rising-edge contact to be connected in parallel

Explanations:

ORP instruction is used in the parallel connection of the rising-edge contact.

Program Example:

Ladder Diagram:



Instruction Code:

		Operation:
LD	X0	; Load NO contact of X0
ORP	**X1**	; X1 rising-edge contact in parallel connection
OUT	Y1	; Drive Y1 coil

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
95	ORF	S	Falling-edge parallel connection				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ORF: 3 steps
S	*	*	*	*							*	*				

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

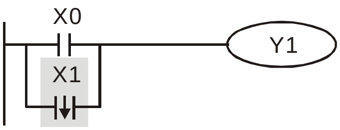
S: falling-edge contact to be connected in parallel

Explanations:

ORF instruction is used in the parallel connection of the falling-edge contact..

Program Example:

Ladder Diagram:



Instruction Code:		Operation:
LD	X0	; Load NO contact of X0
ORF	**X1**	; X1 falling-edge contact in parallel connection
OUT	Y1	; Drive Y1 coil

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
96	TMR	S1 S2	Timer				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TMR: 5 steps
S_1											*					
S_2					*								*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: No. of timer (T0~T255) **S_2**: Set value (K0~K32,767, D0~D9,999)

Explanations:

When TMR instruction is executed, the specific coil of timer is ON and the timer is enabled. When the set value of timer is achieved, the associated NO/NC contact will be driven.

Program example:

Ladder Diagram:

Instruction Code:			Operation:
LD	X0		; Load NO contact X0
TMR	**T5**	**K1000**	; T5 timer setting is K1000

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
97	CNT	S1 S2	16-bit counter				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	CNT: 5 steps
S_1												*				
S_2					*								*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: No. of 16-bit counter (C0~C199) **S_2**: Set value (K0~K32,767, D0~D9,999)

Explanations:

1. When the CNT instruction is executed, the specific coil of counter is driven from OFF to ON once, which means the count value of counter will be added by7 1. When the accumulated count value achieves the set value, the associated NO/NC contact will be driven.
2. When set value of counter is achieved and the counter is driven again, the count value and the status of the associated contact will remain intact. If users need to restart the counting or clear the count value, please use RST instruction.

Program example:

Ladder Diagram:

Instruction Code:			Operation:
LD	X0		; Load NO contact X0
CNT	**C20**	**K100**	; C20 counter setting is K100

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
97	DCNT	S1 S2	32-bit counter				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DCNT: 9 steps
S_1												*				
S_2					*								*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: No. of 32-bit counter (C200~C254)

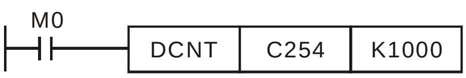
S_2: Set value (K-2,147,483,648~K2,147,483,647, D0~D9,999)

Explanations:

1. DCNT is the startup instruction for the 32-bit counters C200 to C254.
2. For general counting up/down counters C200~C231(SS2/SA2/SE/SX2: C200~C232), the present value will plus 1 or minus 1 according to the counting mode set by flags M1200~M1231 when instruction DCNT is executed.
3. For high speed counters C232~C254(SS2/SA2/SE/SX2: C233~C254), when the specified high speed counter input is triggered by pulse, the counters will start counting. For details about high-speed input terminals (X0~X7) and counting modes (count up/down), please refer to section **2.12 C (Counter).**
4. When DCNT instruction is OFF, the counter will stop counting, but the count value will not be cleared. Users can use RST instruction to remove the count value and reset the contact, or use DMOV instruction to move a specific value into the register. For high-speed counters C232~C254, use specified external input point to clear the count value and reset the contacts.

Program Example:

Ladder Diagram:



Instruction Code:	Operation:
LD M0	; Load NO contact M0
DCNT C254 K1000	; C254 counter setting is K1000

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
98	INV	-	Inverse operation				

OP	Descriptions	Program Steps
N/A	Invert the current result of the internal PLC operations	INV: 1 step

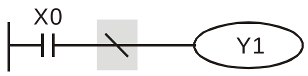
PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Explanations:

INV instruction inverts the logical operation result.

Program Example:

Ladder Diagram:



Instruction Code: Operation:

LD	X0	; Load NO contact X0
INV		; Invert the operation result
OUT	Y1	; Drive Y1 coil

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
99	PLF	(S)	Falling-edge output				

Type / OP	Bit Devices				Word devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		PLF: 3 steps
S		*	*														

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

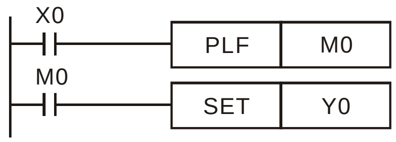
S: Falling pulse output device

Explanations:

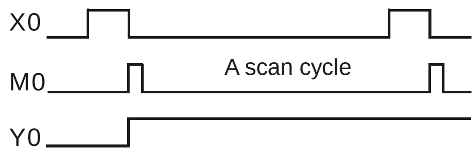
When X0 goes from ON to OFF (Falling-edge trigger), PLS instruction executes and **S** generates a cycle pulse for one operation cycle.

Program Example:

Ladder Diagram:



Timing Diagram:



Instruction Code:		Operation:
LD	X0	; Load NO contact X0
PLF	**M0**	; M0 falling-edge output
LD	M0	; Load NO contact M0
SET	Y0	; Y0 latched (ON)

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
258	ATMR	S1 S2	Contact type timer				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																ATMR: 5 steps
S_1											*					
S_2					*	*							*			

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Timer number (T0~T255) **S_2**: Setting value (K0~K32,767, D0~D9,999)。

Explanations:

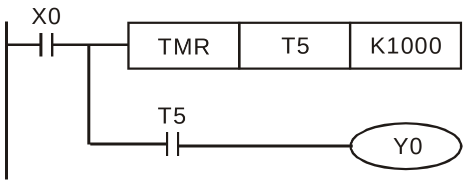
1. DVP-ES2/EX2 series PLCs whose version is 3.20/DVP-SS2 series PLCs whose version is 3.00/DVP-SA2 series PLCs whose version is 2.60/DVP-SE series PLCs whose version is 1.20/DVP-SX2 series PLCs whose version is v2.40 (or above) are supported.
2. When the instruction ATMR is executed, the coil of the timer specified is driven. When the timer value is equal to the setting value, the state of the normally-open contact is On, and the normally-closed contact is Off.

Normally-open contact	On
Normally-closed contact	Off

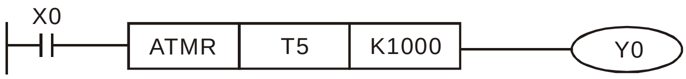
Program Example:

When the normally-open contact X0 is On, the timer T5 begins to measure time intervals. If the timer value is larger than or equal to K1000, the normally-open contact Y0 will be On.

Ladder diagram (The instruction TMR is used.)



Ladder diagram (The instruction ATMR is used.)



3.6.11 Communication Instructions

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2	SS2	SA2 SE	SX2	16-bit	32-bit
100	MODRD	-	-	Read Modbus data	✓	✓	✓	✓	7	-
101	MODWR	-	-	Write Modbus Data	✓	✓	✓	✓	7	-
102	FWD	-	-	Forward Operation of VFD	✓	✓	✓	✓	7	-
103	REV	-	-	Reverse Operation of VFD	✓	✓	✓	✓	7	-
104	STOP	-	-	Stop VFD	✓	✓	✓	✓	7	-
105	RDST	-	-	Read VFD Status	✓	✓	✓	✓	5	-
106	RSTEF	-	-	Reset Abnormal VFD	✓	✓	✓	✓	5	-
107	LRC	-	✓	LRC checksum	✓	✓	✓	✓	7	-
108	CRC	-	✓	CRC checksum	✓	✓	✓	✓	7	-
113	ETHRW	-	-	Ethernet communication	ES2-E	-	✓	✓	9	-
150	MODRW	-	-	MODBUS Read/ Write	✓	✓	✓	✓	11	-
206	ASDRW	-	-	ASDA servo drive R/W	-	✓	✓	✓	7	-
295	DMVRW	-	-	DMV Communication Command	-	✓	-	-	9	-
337	ETHRS	-	-	Self-defined Ethernet communication Command	ES2-E	-	✓	✓	13	-

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
100	MODRD	S1 S2 n	Read Modbus Data				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MODRD: 7 steps
S_1					*	*							*			
S_2					*	*							*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

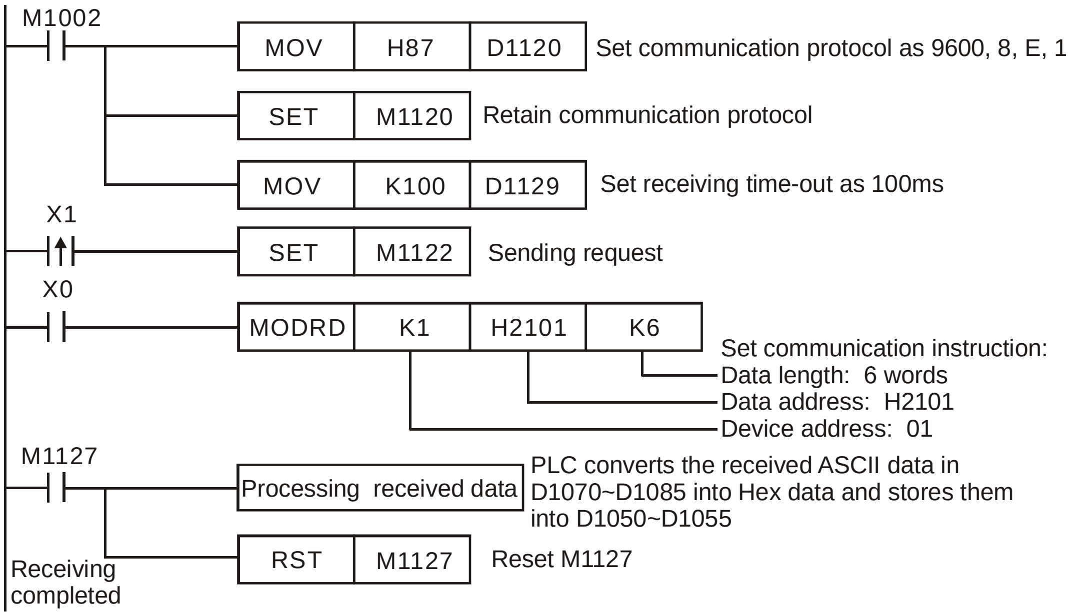
S_1: Device address (K0~K254)   **S_2**: Data address   **n**: Data length (K1＜**n**≦K6)

Explanations:

1. MODRD instruction supports COM2 (RS-485).
2. MODRD is an instruction exclusively for peripheral communication equipment in MODBUS ASCII/RTU mode. The built-in RS-485 communication ports in Delta VFD drives (except for VFD-A series) are all compatible with MODBUS communication format. MODRD can be used for communication (read data) of Delta drives.
3. If the address of **S_2** is illegal for the designated communication device, the device will respond with an error, PLC will record the error code in D1130 and M1141 will be ON.
4. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1085. After data receiving is completed, PLC will check the validity of the data automatically. If there is an error, M1140 will be ON.
5. The feedback data are all ASCII codes in ASCII mode, so PLC will convert the feedback data into hex data and store them in D1050 ~ D1055. D1050 ~ D1055 is invalid in RTU mode.
6. If peripheral device receives a correct record (data) from PLC after M1140/M1141 = ON, the peripheral device will send out feedback data and PLC will reset M1140/M1141 after the validity of data is confirmed.
7. There is no limitation on the times of using this instruction, but only one instruction can be executed at a time on the same COM port.
8. Rising-edge contact (LDP, ANDP, ORP) and falling-edge contact (LDF, ANDF, ORF) can not be used with MODRD instruction, otherwise the data stored in the receiving registers will be incorrect.
9. For associated flags and special registers, please refer to **Points to note** of API 80 RS instruction.

Program Example 1:

Communication between PLC and VFD-B series AC motor drives (ASCII Mode, M1143 = OFF)



PLC → VFD-B , PLC transmits: “01 03 2101 0006 D4”

VFD-B → PLC , PLC receives: “01 03 0C 0100 1766 0000 0000 0136 0000 3B”

Registers for data to be sent (sending messages)

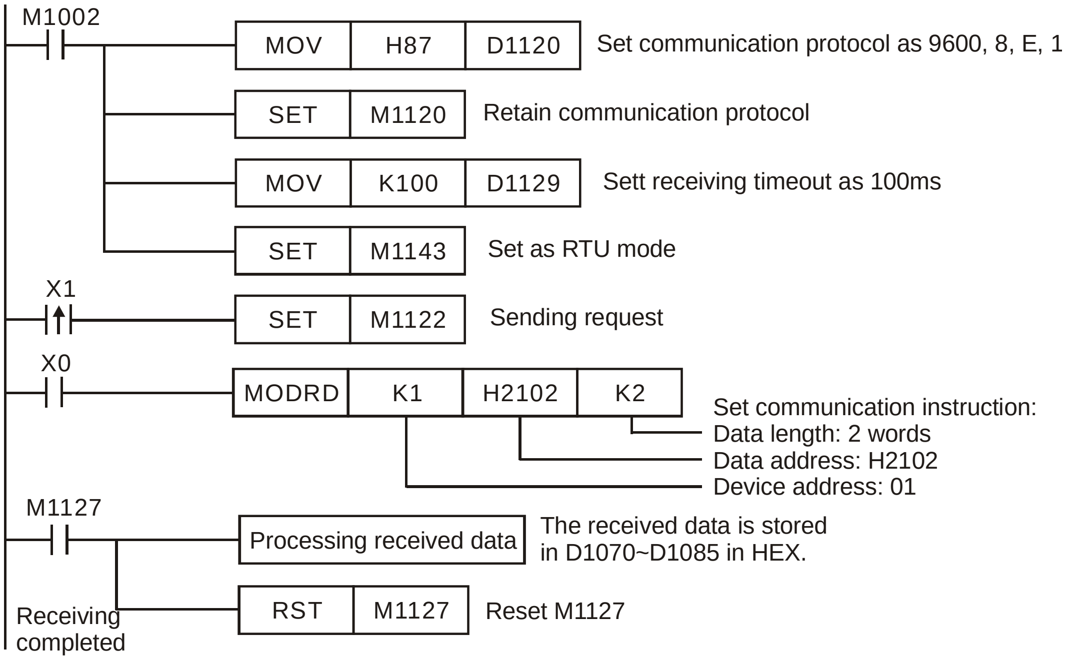
Register	Data		Descriptions	
D1089 low byte	‘0’	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D1089 high byte	‘1’	31 H	ADR 0	
D1090 low byte	‘0’	30 H	CMD 1	Command code: CMD (1,0)
D1090 high byte	‘3’	33 H	CMD 0	
D1091 low byte	2’	32 H	Starting data address	
D1091 high byte	‘1’	31 H		
D1092 low byte	‘0’	30 H		
D1092 high byte	‘1’	31 H		
D1093 low byte	‘0’	30 H	Number of data (count by word)	
D1093 high byte	‘0’	30 H		
D1094 low byte	‘0’	30 H		
D1094 high byte	‘6’	36 H		
D1095 low byte	‘D’	44 H	LRC CHK 1	Checksum: LRC CHK (0,1)
D1095 high byte	‘4’	34 H	LRC CHK 0	

Registers for received data (responding messages)

Register	Data		Descriptions	
D1070 low byte	'0'	30 H	ADR 1	
D1070 high byte	'1'	31 H	ADR 0	
D1071 low byte	'0'	30 H	CMD 1	
D1071 high byte	'3'	33 H	CMD 0	
D1072 low byte	'0'	30 H	Number of data (count by byte)	
D1072 high byte	'C'	43 H		
D1073 low byte	'0'	30 H	Content of address 2101 H	0100 H PLC automatically converts ASCII codes and store the converted value in D1050
D1073 high byte	'1'	31 H		
D1074 low byte	'0'	30 H		
D1074 high byte	'0'	30 H		
D1075 low byte	'1'	31 H	Content of address 2102 H	1766 H PLC automatically converts ASCII codes and store the converted value in D1051
D1075 high byte	'7'	37 H		
D1076 low byte	'6'	36 H		
D1076 high byte	'6'	36 H		
D1077 low byte	'0'	30 H	Content of address 2103 H	0000 H PLC automatically converts ASCII codes and store the converted value in D1052
D1077 high byte	'0'	30 H		
D1078 low byte	'0'	30 H		
D1078 high byte	'0'	30 H		
D1079 low byte	'0'	30 H	Content of address 2104 H	0000 H PLC automatically converts ASCII codes and store the converted value in D1053
D1079 high byte	'0'	30 H		
D1080 low byte	'0'	30 H		
D1080 high byte	'0'	30 H		
D1081 low byte	'0'	30 H	Content of address 2105 H	0136 H PLC automatically converts ASCII codes and store the converted value in D1054
D1081 high byte	'1'	31 H		
D1082 low byte	'3'	33 H		
D1082 high byte	'6'	36 H		
D1083 low byte	'0'	30 H	Content of address 2106 H	0000 H PLC automatically converts ASCII codes and store the converted value in D1055
D1083 high byte	'0'	30 H		
D1084 low byte	'0'	30 H		
D1084 high byte	'0'	30 H		
D1085 low byte	'3'	33 H	LRC CHK 1	
D1085 high byte	'B'	42 H	LRC CHK 0	

Program Example 2:

Communication between PLC and VFD-B series AC motor drive (RTU Mode, M1143= ON)



PLC → VFD-B , PLC transmits: 01 03 2102 0002 6F F7

VFD-B → PLC, PLC receives: 01 03 04 1770 0000 FE 5C

Registers for data to be sent (sending messages)

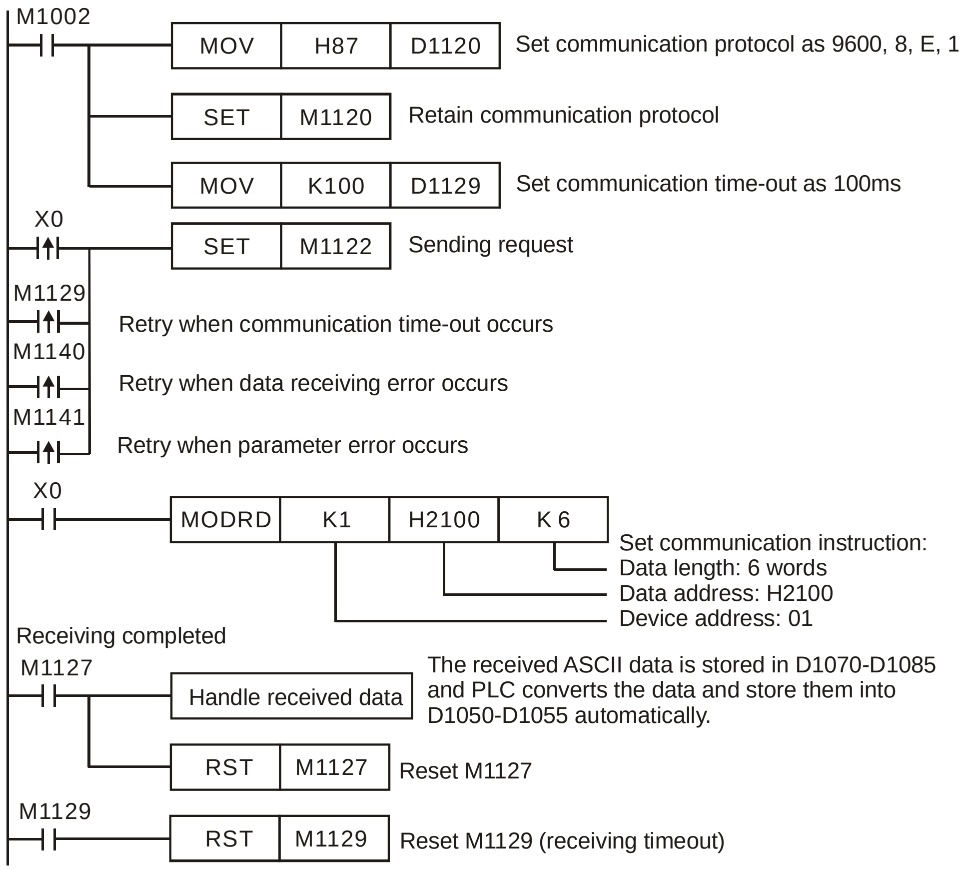
Register	Data	Descriptions
D1089 low byte	01 H	Address of AC motor drive
D1090 low byte	03 H	Command code of AC motor drive
D1091 low byte	21 H	Starting data address
D1092 low byte	02 H	
D1093 low byte	00 H	Number of data (count by word)
D1094 low byte	02 H	
D1095 low byte	6F H	CRC CHK Low
D1096 low byte	F7 H	CRC CHK High

Registers for received data (responding messages)

Register	Data	Descriptions
D1070 low byte	01 H	Address of AC motor drive
D1071 low byte	03 H	Command code of AC motor drive
D1072 low byte	04 H	Number of data (count by byte)
D1073 low byte	17 H	Content of address 2102 H
D1074 low byte	70 H	
D1075 low byte	00 H	Content of address 2103 H
D1076 low byte	00 H	
D1077 low byte	FE H	CRC CHK Low
D1078 low byte	5C H	CRC CHK High

Program Example 3:

1. In the communication between PLC and VFD-B series AC motor drive (ASCII Mode, M1143 = OFF), executes Retry when communication time-out, data receiving error or parameter error occurs.
2. When X0 = ON, PLC will read the data of address H2100 in device 01(VFD-B) and stores the data in ASCII format in D1070 ~ D1085. PLC will automatically convert the data and store them in D1050 ~ D1055.
3. M1129 will be ON when communication time-out occurs. The program will trigger M1129 and send request for reading the data again.
4. M1140 will be ON when data receiving error occurs. The program will trigger M1140 and send request for reading the data again.
5. M1141 will be ON when parameter error occurs. The program will trigger M1141 and send request for reading the data again.



API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
101	MODWR	S1 S2 n	Write Modbus Data				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																MODWR: 7 steps
S_1					*	*							*			
S_2					*	*							*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

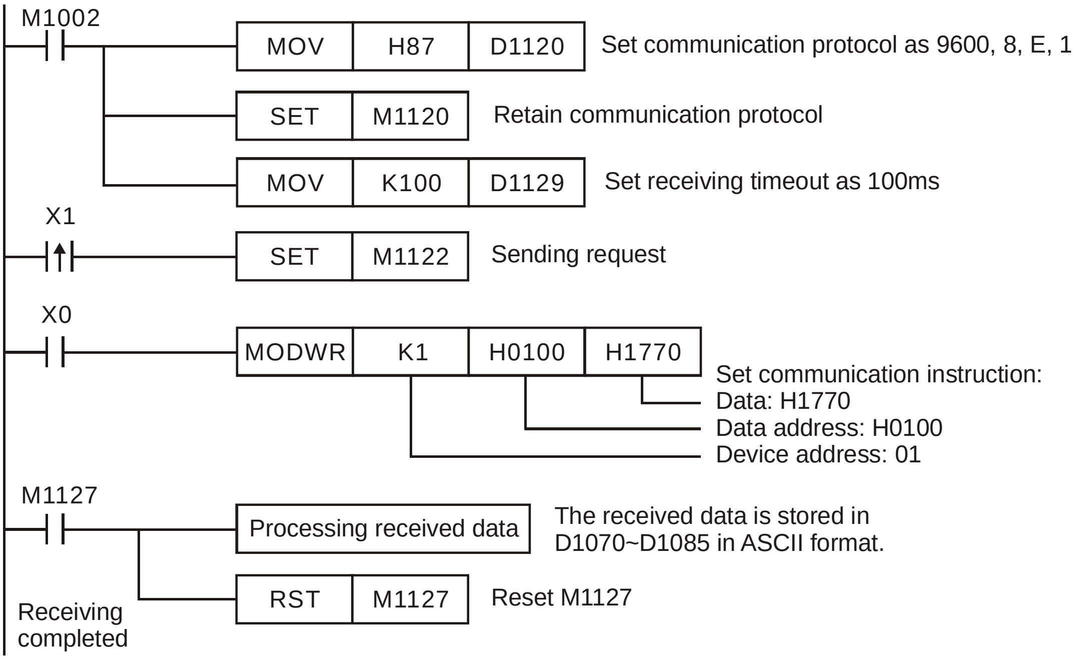
S_1: Device address (K0~K254) **S_2**: Data address **n**: Data to be written

Explanations:

1. MODWR instruction supports COM2 (RS-485).
2. MODWR is an instruction exclusively for peripheral communication equipment in MODBUS ASCII/RTU mode. The built-in RS-485 communication ports in Delta VFD drives (except for VFD-A series) are all compatible with MODBUS communication format. MODRD can be used for communication (write data) of Delta drives.
3. If the address of **S_2** is illegal for the designed communication device, the device will respond with an error, PLC will record the error code in D1130 and M1141 will be ON. For example, if 8000H is invalid to VFD-B, M1141 will be ON and D1130 = 2. For error code explanations, please see the user manual of VFD-B.
4. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1085. After data receiving is completed, PLC will check the validity of the data automatically. If there is an error, M1140 will be ON
5. If peripheral device receives a correct record (data) from PLC after M1140/M1141 = ON, the peripheral device will send out feedback data and PLC will reset M1140/M1141 after the validity of data is confirmed.
6. There is no limitation on the times of using this instruction, but only one instruction can be executed at a time on the same COM port.
7. If rising-edge contacts (LDP, ANDP, ORP) or falling-edge contacts (LDF, ANDF, ORF) is used before MODWR instruction, sending request flag M1122 has to be executed as a requirement.
8. For associated flags and special registers, please refer to **Points to note** of API 80 RS instruction

Program Example 1:

Communication between PLC and VFD-B series AC motor drives (ASCII Mode, M1143 = OFF)



PLC → VFD-B, PLC transmits: “01 06 0100 1770 71 ”

VFD-B → PLC, PLC receives: “01 06 0100 1770 71 ”

Registers for data to be sent (sending messages)

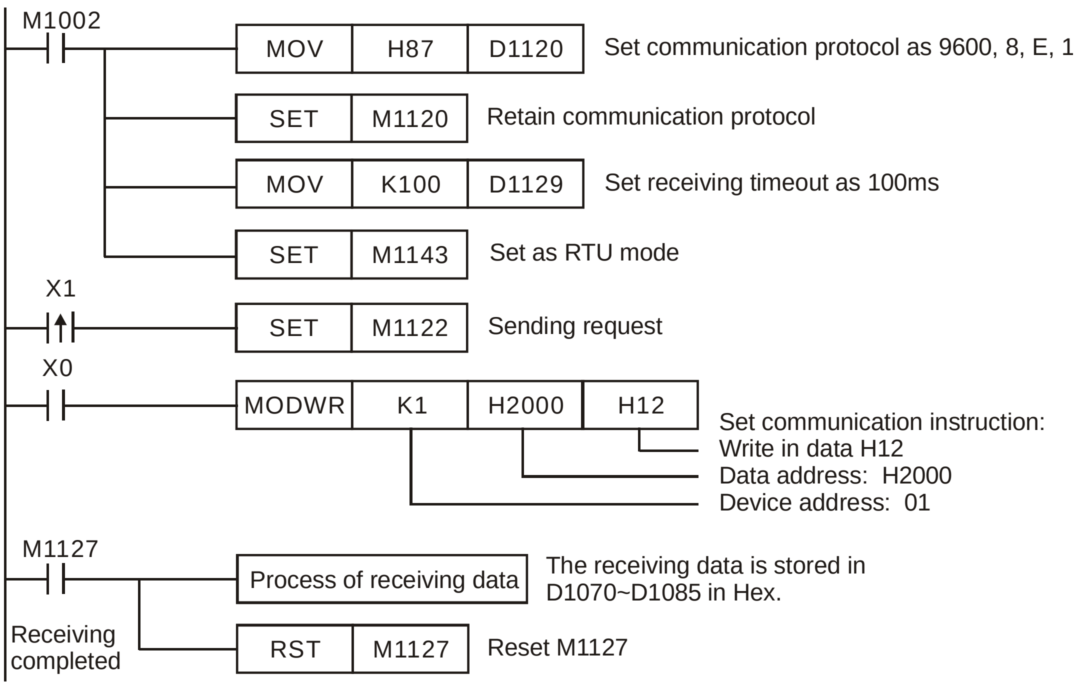
Register	Data		Descriptions	
D1089 low	‘0’	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D1089 high	‘1’	31 H	ADR 0	
D1090 low	‘0’	30 H	CMD 1	Command code of AC motor drive: CMD (1,0)
D1090 high	‘6’	36 H	CMD 0	
D1091 low	‘0’	30 H	Data address	
D1091 high	‘1’	31 H		
D1092 low	‘0’	30 H		
D1092 high	‘0’	30 H		
D1093 low	‘1’	31 H	Data contents	
D1093 high	‘7’	37 H		
D1094 low	‘7’	37 H		
D1094 high	‘0’	30 H		
D1095 low	‘7’	37 H	LRC CHK 1	Checksum: LRC CHK (0,1)
D1095 high	‘1’	31 H	LRC CHK 0	

Registers for received data (responding messages)

Register	Data		Descriptions
D1070 low	‘0’	30 H	ADR 1
D1070 high	‘1’	31 H	ADR 0
D1071 low	‘0’	30 H	CMD 1
D1071 high	‘6’	36 H	CMD 0
D1072 low	‘0’	30 H	Data address
D1072 high	‘1’	31 H	
D1073 low	‘0’	30 H	
D1073 high	‘0’	30 H	
D1074 low	‘1’	31 H	Data content
D1074 high	‘7’	37 H	
D1075 low	‘7’	37 H	
D1075 high	‘0’	30 H	
D1076 low	‘7’	37 H	LRC CHK 1
D1076 high	‘1’	31 H	LRC CHK 0

Program Example 2:

Communication between PLC and VFD-B series AC motor drives (RTU Mode, M1143 = ON)



PLC → VFD-B, PLC transmits: 01 06 2000 0012 02 07

VFD-B → PLC, PLC receives: 01 06 2000 0012 02 07

Registers for data to be sent (sending messages)

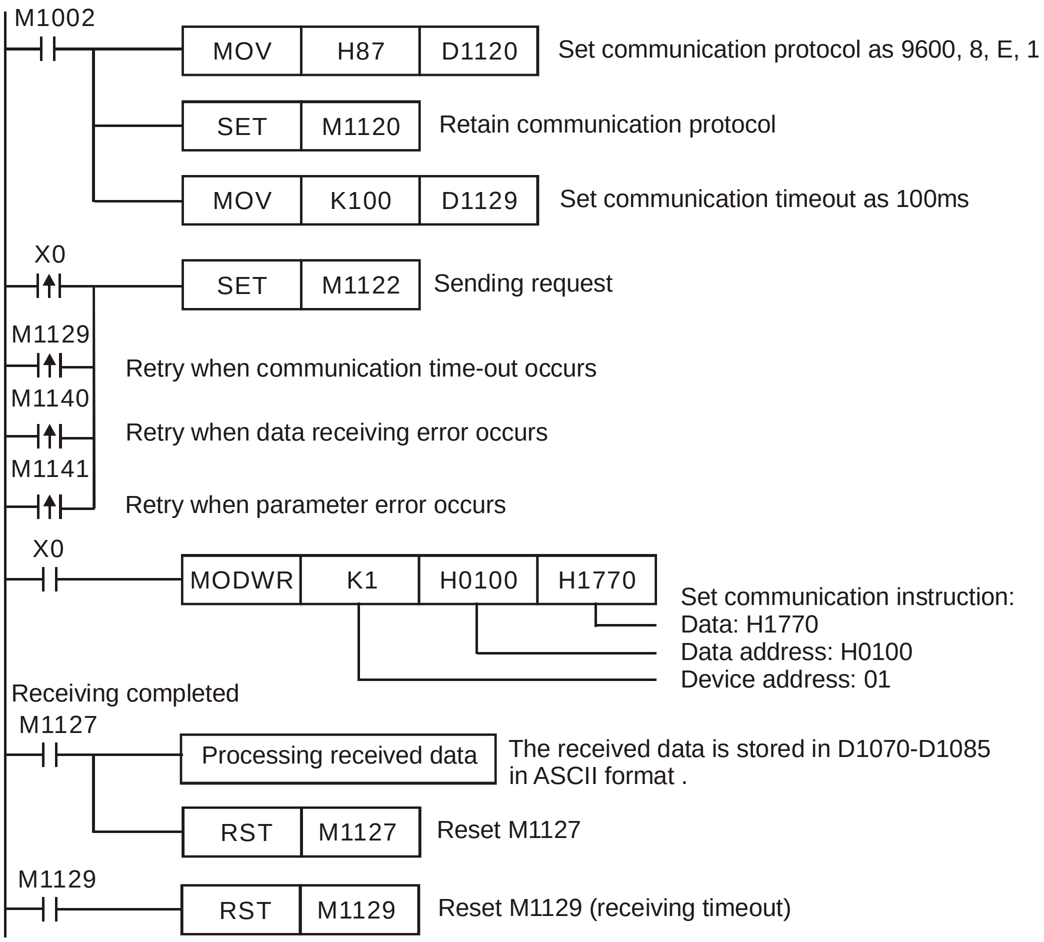
Register	Data	Descriptions
D1089 low	01 H	Address of AC motor drive
D1090 low	06 H	Command code of AC motor drive
D1091 low	20 H	Data address
D1092 low	00 H	
D1093 low	00 H	Data content
D1094 low	12 H	
D1095 low	02 H	CRC CHK Low
D1096 low	07 H	CRC CHK High

Registers for received data (responding messages)

Register	Data	Descriptions
D1070 low	01 H	Address of AC motor drive
D1071 low	06 H	Command code of AC motor drive
D1072 low	20 H	Data address
D1073 low	00 H	
D1074 low	00 H	Data content
D1075 low	12 H	
D1076 low	02 H	CRC CHK Low
D1077 low	07 H	CRC CHK High

Program Example 3:

1. In the communication between PLC and VFD-B series AC motor drive (ASCII Mode, M1143 = OFF), executes Retry when communication time-out, data receiving error or parameter error occurs
2. When X0 = ON, PLC will write data H1770 (K6000) into address H0100 in device 01 (VFD-B).
3. M1129 will be ON when communication time-out occurs. The program will trigger M1129 and send request for reading the data again.
4. M1140 will be ON when data receiving error occurs. The program will trigger M1140 and send request for reading the data again.
5. M1141 will be ON when parameter error occurs. The program will trigger M1141 and send request for reading the data again.


M1002
MOV
H87
D1120
Set communication protocol as 9600, 8, E, 1
SET
M1120
Retain communication protocol
MOV
K100
D1129
Set communication timeout as 100ms
X0
SET
M1122
Sending request
M1129
Retry when communication time-out occurs
M1140
Retry when data receiving error occurs
M1141
Retry when parameter error occurs
X0
MODWR
K1
H0100
H1770
Set communication instruction:
Data: H1770
Data address: H0100
Device address: 01
Receiving completed
M1127
Processing received data
The received data is stored in D1070-D1085
in ASCII format .
RST
M1127
Reset M1127
M1129
RST
M1129
Reset M1129 (receiving timeout)

API	Mnemonic	Operands	Function	Controllers			
102	FWD	S1 S2 n	Forward Operation of VFD	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			FWD: 7 steps
S_2					*	*							*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

API	Mnemonic	Operands	Function	Controllers			
103	REV	S1 S2 n	Reverse Operation of VFD	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			REV: 7 steps
S_2					*	*							*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
104	STOP	S1 S2 n	Stop VFD				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	STOP: 7 steps
S_1					*	*							*			
S_2					*	*							*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

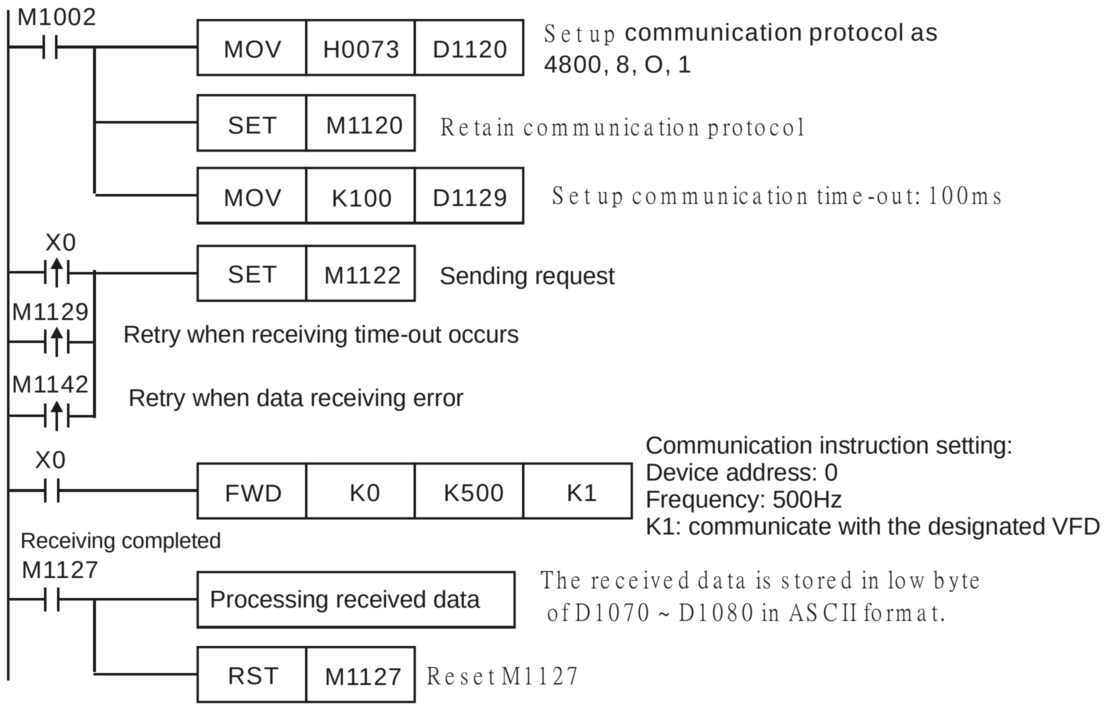
S_1: Device address **S_2**: Operation frequency of VFD **n**: Operation mode

Explanations:

1. M1177 = OFF (Default), FWD, REV, STOP instructions support COM2(RS-485).
2. M1177= ON, FWD, REV, STOP instructions support COM2(RS-485), COM3(RS-485).
3. M1177 has to be set up in advance for selecting the target model of VFD. When M1177 = OFF (Default), FWD, REV, STOP instructions support Delta's VFD-A inverter. When M1177 = ON, these instructions support other models of VFD inverters, e.g. VFD-B, VFD.
4. There is no limitation on the times of using FWD, REV, STOP instruction, however only one instruction can be executed on single COM port at a time.
5. If rising-edge (LDP, ANDP, ORP) or falling-edge (LDF, ANDF, ORF) contacts are used before FWD, REV, STOP instructions, sending request flags M1122 (COM2) / M1316 (COM3) has to be enabled in advance for obtaining correct operation.
6. For detailed information of associated flags and special registers, please refer to RS instruction.
7. M1177 = OFF, only Delta VFD-A is supported and the definition of each operand is:
   a) **S_1** = Address of VFD-A. Range of **S_1**: K0 ~ K31
   b) **S_2** = Operation frequency of VFD. Set value for VFD A-type inverter: K0 ~ K4,000 (0.0Hz ~ 400.0Hz).
   c) **n** = Communication mode. Range: K1 ~ K2. **n** = 1: communicate with VFD at designated address. **n** = 2: communicate with all connected VFDs. .
   d) The feedback data from the peripheral equipment will be stored in D1070 ~ D1080 After data receiving is completed, PLC will check if all data are correct automatically. If there is an error, M1142 will be ON. When **n** = 2, PLC will not receive any data.

Program Example: COM2 (RS-485)

1. Communication between PLC and VFD-A series inverter. Retry for communication time-out and data receiving error.



PLC ⇨ VFD-A, PLC sends: "**C ♥ ☺ 0001 0500** "

VFD-A ⇨ PLC, PLC receives: "**C ♥ ♠ 0001 0500** "

Registers for data to be sent (sending messages)

Register	Data		Descriptions
D1089 low	'C'	43 H	Header of control string
D1090 low	'♥'	03 H	Checksum
D1091 low	'☺'	01 H	Command acknowledgement (communication mode)
D1092 low	'0'	30 H	Communication address
D1093 low	'0'	30 H	
D1094 low	'0'	30 H	
D1095 low	'1'	31 H	
D1096 low	'0'	30 H	Operation command
D1097 low	'5'	35 H	
D1098 low	'0'	30 H	
D1099 low	'0'	30 H	

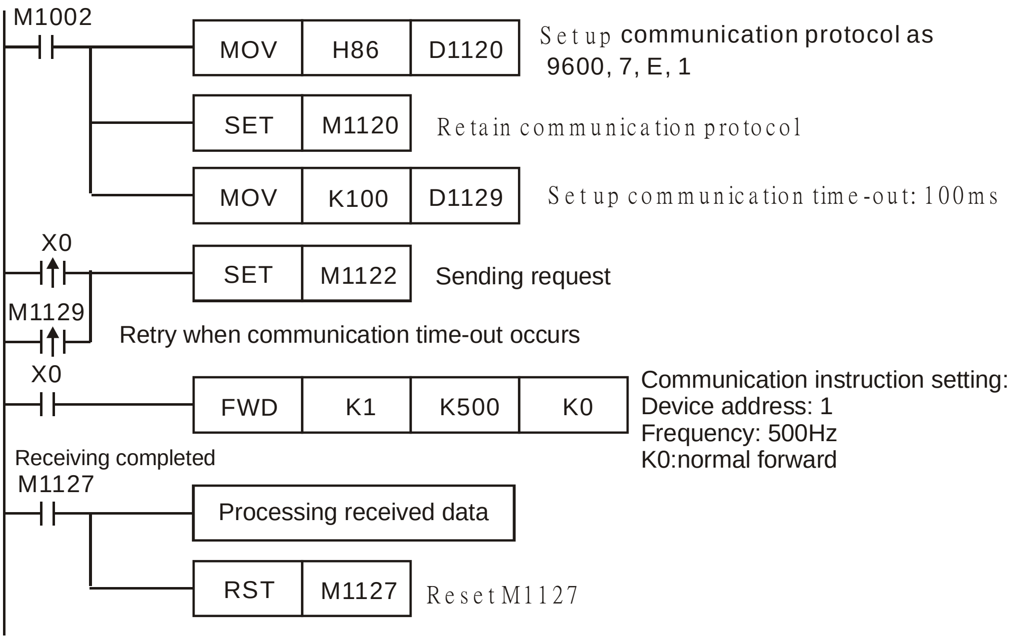
Registers for received data (responding messages)

Register	DATA		Explanation
D1070 low	‘C’	43 H	Header of control string
D1071 low	‘♥’	03 H	Checksum
D1072 low	‘♠’	06 H	Acknowledge back. (Check feedback data) (correct: 06H, Error: 07 H)
D1073 low	‘0’	30 H	Communication address
D1074 low	‘0’	30 H	
D1075 low	‘0’	30 H	
D1076 low	‘1’	31 H	
D1077 low	‘0’	30 H	Operation command
D1078 low	‘5’	35 H	
D1079 low	‘0’	30 H	
D1080 low	‘0’	30 H	

2. M1177 = ON, other Delta VFDs are supoported
   a) **S_1** = Address of VFD-A. Range of **S_1**: K0 ~ K255, when **S_1** is specified as K0, PLC will broadcast to all connected VFDs.
   b) **S_2** = **Running** frequency of VFD. Please refer to manuals of specific VFD. In STOP instruction, operand **S_2** is reserved.
   c) **n = Operation** mode.
      - In FWD instruction: **n** = 0 → Forward mode; **n** = 1 → Forward JOG. Other values will be regarded as normal forward mode.
      - In REV instruction: **n** = 0 → Reverse mode; **n** = 1 → Reverse JOG. Other values will be regarded as normal reverse mode
      - In STOP instruction: operand **n** is reserved.
   d) When Forward JOG is selected in FWR instruction, set value in **S_2** is invalid. If users need to modify the JOG frequency, please refer to manuals of specific VFDs.

Program Example: COM2 (RS-485)

Communication between PLC and VFD-B series inverter (ASCII Mode, M1143 = OFF), Retry when communication time-out occurs.



PLC ⇨ VFD, PLC sends: “:**01 10 2000 0002 04 0012 01F4 C2** ”

VFD ⇨ PLC, PLC sends: “:**01 10 2000 0002 CD** ”

Data to be sent (sending messages)

Data		Descriptions	
‘0’	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
‘1’	31 H	ADR 0	
‘1’	31 H	CMD 1	Command code: CMD (1,0)
‘0’	30 H	CMD 0	
‘2’	32 H	Data Address	
‘0’	30 H		
‘0’	30 H		
‘0’	30 H		
‘0’	30 H	Data content	
‘0’	30 H		
‘0’	30 H		
‘2’	32 H		
‘0’	30 H	Byte Count	
‘4’	34 H		
‘0’	30H	Data content 1	H1: forward operation
‘0’	30 H		
‘1’	31 H		
‘2’	32 H		
‘0’	30 H	Data content 2	Operation frequency = K500Hz H01F4
‘1’	31 H		
‘F’	46 H		
‘4’	34 H		
‘C’	43 H	LRC CHK 1	Error checksum: LRC CHK (0,1)
‘2’	32 H	LRC CHK 0	

Received data (responding messages)

Data		Descriptions
'0'	30 H	ADR 1
'1'	31 H	ADR 0
'1'	31 H	CMD 1
'0'	30 H	CMD 0
'2'	32 H	Data Address
'0'	30 H	
'0'	30 H	
'0'	30 H	
'0'	30 H	Number of Register
'0'	30 H	
'0'	30 H	
'2'	32 H	
'C'	43 H	LRC CHK 1
'D'	44 H	LRC CHK 0

API	Mnemonic	Operands	Function	Controllers			
105	RDST	(S) (n)	Read VFD Status	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	RDST: 5 steps
S					*	*							*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Device address **n**: Status content to be retrieved

Explanations:

1. M1177 = OFF (Default), RDST instruction supports COM2(RS-485).
2. M1177= ON, RDST instruction supports COM2(RS-485), COM3(RS-485).
3. M1177 has to be set up in advance for selecting the target model of VFD. When M1177 = OFF (Default), RDST instruction supports Delta's VFD-A inverter. When M1177 = ON, the instruction supports other models of VFD inverters, e.g. VFD-B, VFD.
4. There is no limitation on the times of using RDST instruction, however only one instruction can be executed on single COM port at a time
5. Rising-edge contacts (LDP, ANDP, ORP) and falling-edge contacts (LDF, ANDF, ORF) can not be used with RDST instructions. Otherwise, the data in receiving registers will be incorrect.
6. For detailed information of associated flags and special registers, please refer to RS instruction.
7. M1177 = OFF, only VFD-A is supported
   a) Range of **S**: K0 ~ K31
   b) Range of **n**: K0 ~ K3
   c) **n**: Status content to be retrieved
      n=0, frequency
      n=1, output frequency
      n=2, output current
      n=3, Operation command
   d) The feedback data consists of 11 bytes (refer to VFD-A user manual), and will be stored in low bytes of D1070 ~ D1080.

      "**Q, S, B, Uu, Nn, ABCD**"

Feedback	Explanation	Data storage
Q	Header of question string: 'Q' (51H).	D1070 low
S	Checksum: 03H.	D0171 low
B	Acknowledge back. Correct: 06H, Error: 07H.	D1072 low
U	Communication address (range: 00~31). Displayed in ASCII format.	D1073 low
U		D1074 low
N	Status content to be retrieved (00 ~ 03). Displayed in ASCII format.	D1075 low
		D1076 low
A	Retrieved status content. The content of "ABCD" differs according to value 00~03 set in NN. 00 ~ 03 indicates frequency, current and operation mode respectively. Please refer to the explanations below for details.	D1077 low
B		D1078 low
C		D1079 low
D		D1080 low

Feedback	Explanation	Data storage
	Nn = “00” Frequency command = ABC.D (Hz) Nn = “01” Output frequency = ABC.D (Hz) Nn = “02” Output current = ABC.D (A) PLC will automatically convert the ASCII characters ”ABCD” into D1050. For example, ”ABCD” = “0600”, PLC will convert ABCD into K0600 (0258 H) and store it in the special register D1050.	
	Nn = “03” Operation command	

‘A’ =	‘0’	Stop,	‘5’	JOG (forward)
	‘1’	Forward operation	‘6’	JOG (reverse)
	‘2’	Stop,	‘7’	JOG (reverse)
	‘3’	Reverse operation	‘8’	Abnormal
	‘4’	JOG (forward),		
	PLC will automatically convert the ASCII character in ”A” into D1051. For example, ”A” = “3”, PLC will convert A into K3 and store it in the special register D1051.			

	b7	b6	b5	b4	Frequency reference source
‘B’ =	0	0	0	0	Digital keypad
	0	0	0	1	1st Step Speed
	0	0	1	0	2nd Step Speed
	0	0	1	1	3rd Step Speed
	0	1	0	0	4th Step Speed
	0	1	0	1	5th Step Speed
	0	1	1	0	6th Step Speed
	0	1	1	1	7th Step Speed
	1	0	0	0	JOG frequency
	1	0	0	1	Analog input frequency command
	1	0	1	0	RS-485 communication interface
	1	0	1	1	Up/Down control

b3	=	0	Non-DC braking stop	1	DC braking stop
b2	=	0	Non-DC braking start	1	DC braking start
b1	=	0	Forward	1	Reverse
b0	=	0	Stop	1	Run

PLC will store bit status of ”B” in special auxiliary relay M1168 (b0) ~ M1175 (b7).

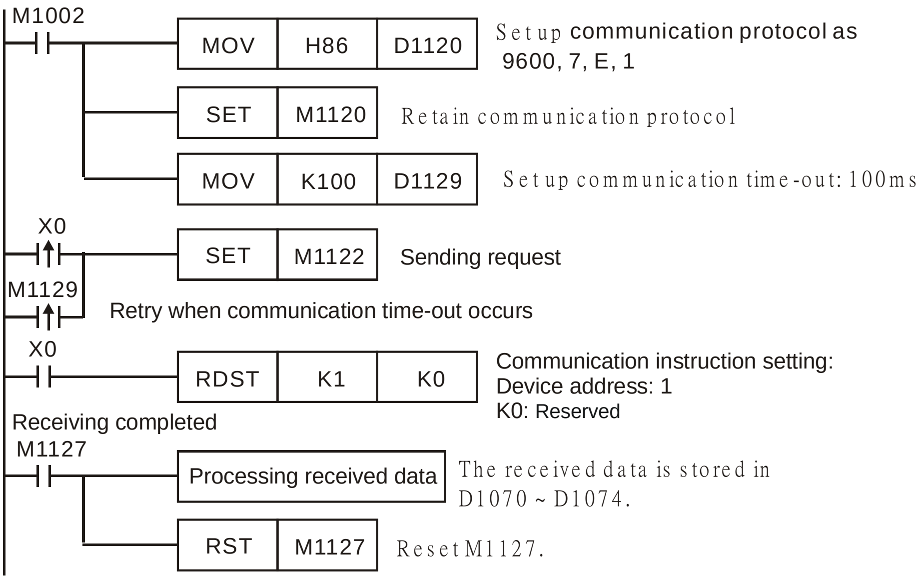
“CD” =	“00”	No error	“10”	OcA
	“01”	oc	“11”	Ocd
	“02”	ov	“12”	Ocn
	“03”	oH	“13”	GFF
	“04”	oL	“14”	Lv
	“05”	oL1	“15”	Lv1
	“06”	EF	“16”	cF2
	“07”	cF1	“17”	bb
	“08”	cF3	“18”	oL2
	“09”	HPF	“19”	

PLC will automatically convert the ASCII characters in ”CD” into D1052. For example, ”CD” = “16”, PLC will convert CD into K16 and store it in the special register D10512

8. M1177 = ON, other Delta VFDs are supoported
   a) Range of **S**$_1$: K1 ~ K255
   b) The instruction will read VFD status at parameter address 2100H~2104H (Please refer to user manual of specific VFD for details.) and store the feedback data in D1070~D1074. However, the content in D1070~D1074 will not be updated when receiving error or timeout occurs. Therefore, please check the status of receiving completed flag before applying the received data

Program Example: COM2 (RS-485)

1. Communication between PLC and VFD-B series inverter (ASCII Mode, M1143 = OFF). Retry when communication time-out occurs.
2. Read VFD status at parameter address 2100H~2104H and store the received data in D1070 ~ D1074.



PLC ⇨ VFD-B, PLC sends: "**:01 03 2100 0005 D6** "

VFD-B ⇨ PLC, PLC receives: ":01 03 0A 00C8 7C08 3E00 93AB 0000 2A "

Data to be sent (sending messages)

Data		Descriptions	
'0'	30 H	ADR 1	AC drive address : ADR (1,0)
'1'	31 H	ADR 0	
'0'	30 H	CMD 1	Command code: CMD (1,0)
'3'	33 H	CMD 0	
2'	32 H	Starting data address	
'1'	31 H		
'0'	30 H		
'0'	30 H		
'0'	30 H	Number of data (count by word)	
'0'	30 H		
'0'	30 H		
'5'	35 H		
'D'	44 H	LRC CHK 1	Error checksum: LRC CHK (0,1)
'6'	36 H	LRC CHK 0	

Received data (responding messages)

<table>
<tr><th colspan="2">Data</th><th colspan="2">Descriptions</th></tr>
<tr><td>‘0’</td><td>30 H</td><td colspan="2">ADR 1</td></tr>
<tr><td>‘1’</td><td>31 H</td><td colspan="2">ADR 0</td></tr>
<tr><td>‘0’</td><td>30 H</td><td colspan="2">CMD 1</td></tr>
<tr><td>‘3’</td><td>33 H</td><td colspan="2">CMD 0</td></tr>
<tr><td>‘0’</td><td>30 H</td><td colspan="2" rowspan="2">Number of data (count by byte)</td></tr>
<tr><td>‘A’</td><td>41 H</td></tr>
<tr><td>‘0’</td><td>30 H</td><td rowspan="4">Content of address 2100 H</td><td rowspan="4">PLC automatically converts ASCII codes and store the converted value in D1070 = 00C8 H</td></tr>
<tr><td>‘0’</td><td>30 H</td></tr>
<tr><td>‘C’</td><td>43 H</td></tr>
<tr><td>‘8’</td><td>38 H</td></tr>
<tr><td>‘7’</td><td>37 H</td><td rowspan="4">Content of address 2101 H</td><td rowspan="4">PLC automatically converts ASCII codes and store the converted value in D1071 = 7C08 H</td></tr>
<tr><td>‘C’</td><td>43 H</td></tr>
<tr><td>‘0’</td><td>30 H</td></tr>
<tr><td>‘8’</td><td>38 H</td></tr>
<tr><td>‘3’</td><td>33 H</td><td rowspan="4">Content of address 2102 H</td><td rowspan="4">PLC automatically converts ASCII codes and store the converted value in D1072 = 3E00 H</td></tr>
<tr><td>‘E’</td><td>45 H</td></tr>
<tr><td>‘0’</td><td>30 H</td></tr>
<tr><td>‘0’</td><td>30 H</td></tr>
<tr><td>‘9’</td><td>39 H</td><td rowspan="4">Content of address 2103H</td><td rowspan="4">PLC automatically converts ASCII codes and store the converted value in D1073 = 93AB H</td></tr>
<tr><td>‘3’</td><td>33 H</td></tr>
<tr><td>‘A’</td><td>41 H</td></tr>
<tr><td>‘B’</td><td>42 H</td></tr>
<tr><td>‘0’</td><td>30 H</td><td rowspan="4">Content of address 2104 H</td><td rowspan="4">PLC automatically converts ASCII codes and store the converted value in D1074 = 0000 H</td></tr>
<tr><td>‘0’</td><td>30 H</td></tr>
<tr><td>‘0’</td><td>30 H</td></tr>
<tr><td>‘0’</td><td>30 H</td></tr>
<tr><td>‘2’</td><td>32 H</td><td colspan="2">LRC CHK 1</td></tr>
<tr><td>‘A’</td><td>41 H</td><td colspan="2">LRC CHK 0</td></tr>
</table>

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
106	RSTEF	(S) (n)	Reset Abnormal VFD				

Type / OP	Bit Devices X	Y	M	S	Word devices K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S					*	*							*			RSTEF: 5 steps
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Address of communication device **n**: Operation mode

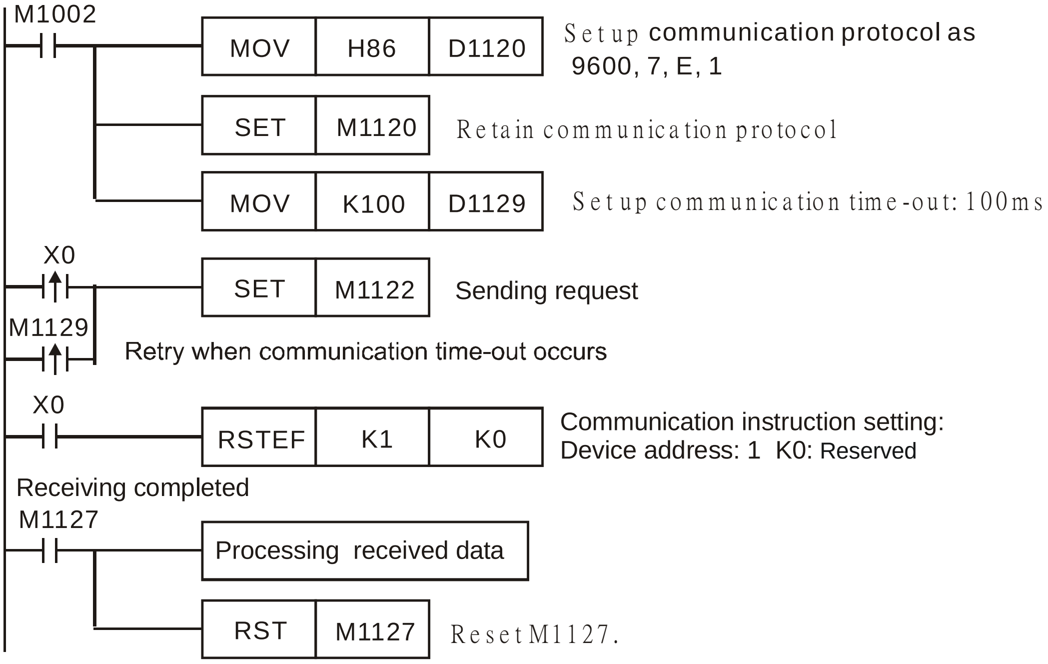
Explanations:

1. M1177 = OFF (Default), RSTEF instruction supports COM2(RS-485).
2. M1177= ON, RSTEF instruction supports COM2(RS-485), COM3(RS-485).
3. M1177 has to be set up in advance for selecting the target model of VFD. When M1177 = OFF (Default), RSTEF instruction supports Delta's VFD-A inverter. When M1177 = ON, these instructions support other models of VFD inverters, e.g. VFD-B, VFD.
4. There is no limitation on the times of using RSTEF instruction, however only one instruction can be executed on single COM port at a time.
5. If rising-edge (LDP, ANDP, ORP) or falling-edge (LDF, ANDF, ORF) contacts are used before RSTEF instruction, sending request flags M1122 (COM2) / M1316 (COM3) has to be enabled in advance for obtaining correct operation.
6. For detailed information of associated flags and special registers, please refer to RS instruction.
7. M1177 = OFF, only Delta VFD-A is supported and the definition of each operand is:
   a) $\mathbf{S_1}$ = Address of VFD-A. Range of $\mathbf{S_1}$: K0 ~ K31
   b) **n** = Communication mode. Range: K1 ~ K2. **n** = 1: communicate with VFD at designated address. **n** = 2: communicate with all connected VFDs. .
   c) RSTEF is a handy communication instruction used for reset when errors occur in AC motor drive operation.
   d) The feedback data from the peripheral equipment will be stored in D1070 ~ D1080. When **n** = 2, PLC will not receive any data.
8. M1177 = ON, other Delta VFDs are supoported
   - $\mathbf{S_1}$ = Address of VFD. Range of $\mathbf{S_1}$: K0 ~ K255, when $\mathbf{S_1}$ is specified as K0, PLC will broadcast to all connected VFDs

Program Example: COM2 (RS-485)

Communication between PLC and VFD-B series AC motor drives (ASCII Mode, M1143 = OFF).

Retry when communication time-out occurs.



PLC ⇨ VFD, PLC sends: "**:01 06 2002 0002 D5** "

VFD ⇨ PLC, PLC sends: "**:01 06 2002 0002 D5** "

Data to be sent (sending messages):

Data		Descriptions	
'0'	30 H	ADR 1	AC drive address : ADR (1,0)
'1'	31 H	ADR 0	
'0'	30 H	CMD 1	Command code: CMD (1,0)
'6'	36 H	CMD 0	
'2'	32 H	Data address	
'0'	30 H		
'0'	30 H		
'2'	32 H		
'0'	30 H	Data contents	
'0'	30 H		
'0'	30 H		
'2'	32 H		
'D'	44 H	LRC CHK 1	Error checksum: LRC CHK (0,1)
'5'	35 H	LRC CHK 0	

Received data (responding messages)

Data		Descriptions
‘0’	30 H	ADR 1
‘1’	31 H	ADR 0
‘0’	30 H	CMD 1
‘6’	36 H	CMD 0
‘2’	32 H	Data address
‘0’	30 H	
‘0’	30 H	
‘2’	32 H	
‘0’	30 H	Data content
‘0’	30 H	
‘0’	30 H	
‘2’	32 H	
‘D’	44 H	LRC CHK 1
‘5’	35 H	LRC CHK 0

API	Mnemonic		Operands	Function	Controllers			
107	LRC	P	S n D	LRC checksum	ES2/EX2/EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	LRC, LRCP: 7 steps
S													*			
n					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/EC5	SS2	SA2 SE	SX2	ES2/EX2/EC5	SS2	SA2 SE	SX2	ES2/EX2/EC5	SS2	SA2 SE	SX2

Operands:

S: Starting device for ASCII mode checksum **n**: Data length for LRC operation (**n** = K1~K256)

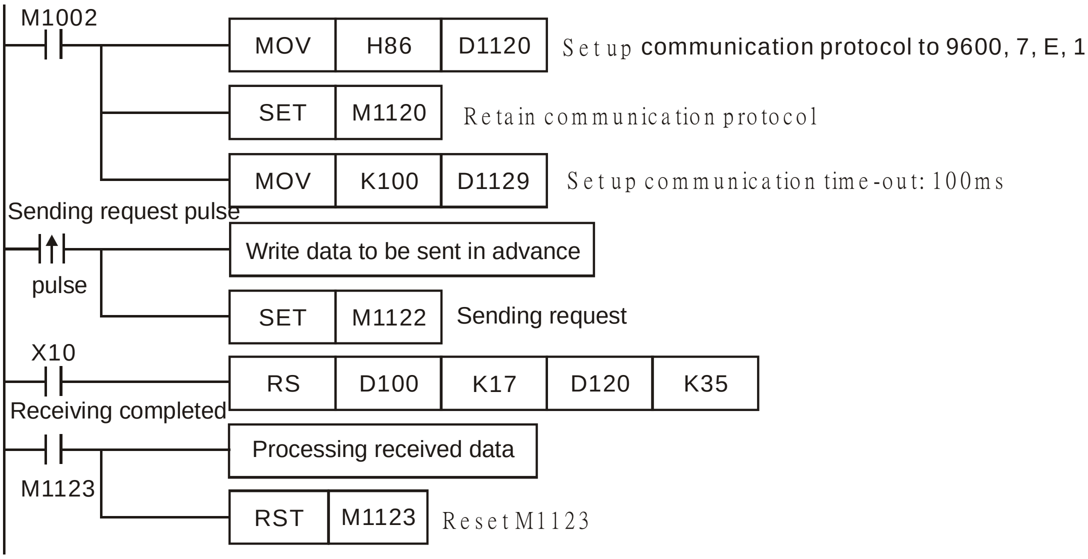
D: Starting device for storing the operation result

Explanations:

1. **n**: **n** must be an even number. If **n** is out of range, an error will occur and the instruction will not be executed. At this time, M1067 and M1068 = ON and error code H’0E1A will be recorded in D1067.
2. 16-bit mode: When LRC instruction operates with M1161 = OFF, hexadecimal data starting from **S** is divided into high byte and low byte and the checksum operation is operated on **n** number of bytes. After this, operation result will be stored in both hi-byte and low byte of **D**.
3. 8-bit mode: When LRC instruction operates with M1161 = ON, hexadecimal data starting from **S** is divided into high byte (invalid) and low byte and the checksum operation is operated on **n** number of low bytes. After this, operation result will be stored in low bytes of **D** (Consecutive 2 registers).
4. Flag: M1161 8/16-bit mode

Program Example:

Connect PLC to VFD series AC motor drive (ASCII mode, M1143 = OFF), (8-bit mode, M1161 = ON), Write the data to be sent into registers starting from D100 in advance for reading 6 data from address H0708 on VFD.

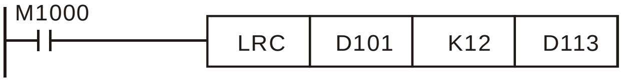


PLC ➪ VFD, PLC sends: **“: 01 03 07 08 0006 E7 CR LF ”**

Registers for sent data (sending messages)

Register	Data		Explanation	
D100 low byte	‘: ’	3A H	STX	
D101 low byte	‘0’	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D102 low byte	‘1’	31 H	ADR 0	
D103 low byte	‘0’	30 H	CMD 1	Command code: CMD (1,0)
D104 low byte	‘3’	33 H	CMD 0	
D105 low byte	‘0’	30 H	Starting data address	
D106 low byte	‘7’	37 H		
D107 low byte	‘0’	30 H		
D108 low byte	‘8’	38 H		
D109 low byte	‘0’	30 H	Number of data (words)	
D110 low byte	‘0’	30 H		
D111 low byte	‘0’	30 H		
D112 low byte	‘6’	36 H		
D113 low byte	‘E’	45 H	LRC CHK 0	Error checksum: LRC CHK (0,1)
D114 low byte	‘7’	37 H	LRC CHK 1	
D115 low byte	CR	D H	END	
D116 low byte	LF	A H		

The error checksum LRC CHK (0, 1) can be calculated by LRC instruction (8-bit mode, M1161 = ON).



LRC checksum: 01 H + 03 H + 07 H + 08 H + 00 H + 06 H = 19 H. Operate 2's complement on 19H and the result is E7H. Store 'E'(45 H) in the low byte of D113 and '7' (37 H) in the low byte of D114.

Remarks:

ASCII mode communication data:

STX	': '	Start word = ': ' (3AH)
Address Hi	' 0 '	Communication: 8-bit address consists of 2 ASCll codes
Address Lo	' 1 '	
Function Hi	' 0 '	Function code: 8-bit function consists of 2 ASCll codes
Function Lo	' 3 '	
DATA (n-1) ……. DATA 0	' 2 '	Data content: n × 8-bit data consists of 2n ASCll codes
	' 1 '	
	' 0 '	
	' 2 '	
	' 0 '	
	' 0 '	
	' 0 '	
	' 2 '	
LRC CHK Hi	' D '	LRC checksum: 8-bit checksum consists of 2 ASCll codes
LRC CHK Lo	' 7 '	
END Hi	CR	End word: END Hi = CR (0DH), END Lo = LF(0AH)
END Lo	LF	

LRC checksum: Operate 2's complement on the summed up value from communication address to the end of data, i.e. 01 H + 03 H + 21 H + 02 H + 00 H + 02 H = 29 H, the operation result of 29H is D7H.

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
108		CRC	P	S n D	CRC checksum				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	CRC, CRCP: 7 steps
S													*			
n					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Starting device for RTU mode checksum **n**: Data length for CRC operation (**n** = K1~K256)

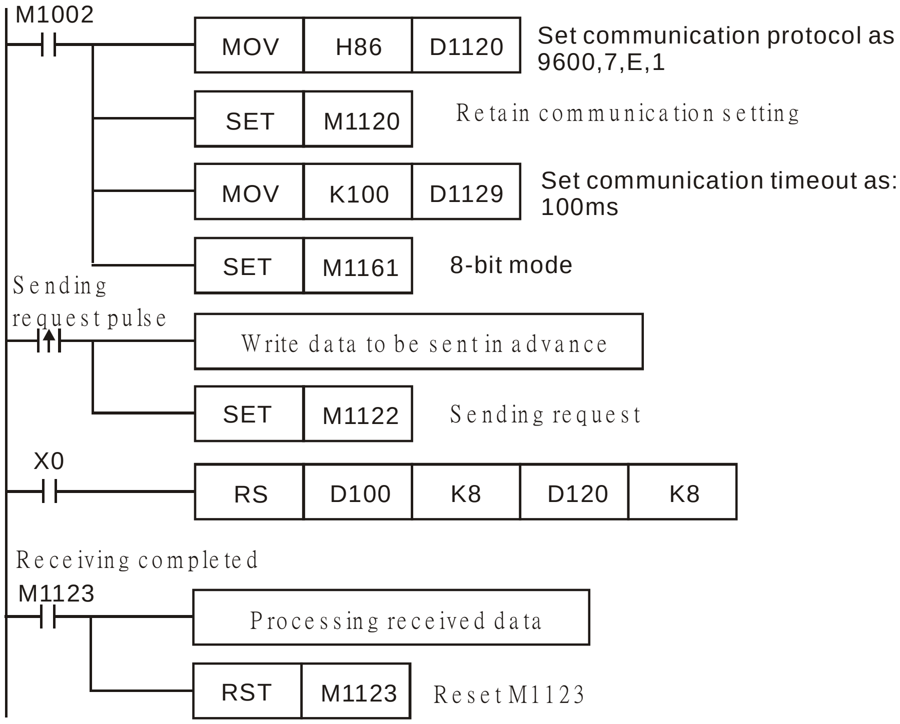
D: Starting device for storing the operation result

Explanations:

1. **n**: **n** must be an even number. If **n** is out of range, an error will occur and the instruction will not be executed. At this time, M1067 and M1068 = ON and error code H'0E1A will be recorded in D1067.
2. 16-bit mode: When CRC instruction operates with M1161 = OFF, hexadecimal data starting from **S** is divided into high byte and low byte and the checksum operation is operated on **n** number of bytes. After this, operation result will be stored in both hi-byte and low byte of **D**.
3. 8-bit mode: When CRC instruction operates with M1161 = ON, hexadecimal data starting from **S** is divided into high byte (invalid) and low byte and the checksum operation is operated on **n** number of low bytes. After this, operation result will be stored in low bytes of **D** (Consecutive 2 registers).
4. Flag: M1161 8/16-bit mode

Program Example:

Connect PLC to VFD series AC motor drive (RTU mode, M1143 = ON), (8-bit mode, M1161 = ON), Write the data to be sent (H1770) into address H0706 on VFD.

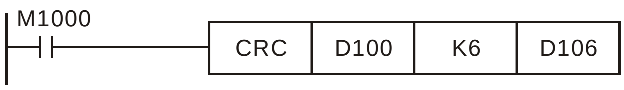


PLC ⇨ VFD, PLC sends: **01 06 0706 1770 66 AB**

Registers for sent data (sending messages)

Register	Data	Explanation
D100 low byte	01 H	Address
D101 low byte	06 H	Function
D102 low byte	07 H	Data address
D103 low byte	06 H	
D104 low byte	17 H	Data content
D105 low byte	70 H	
D106 low byte	66 H	CRC CHK 0
D107 low byte	AB H	CRC CHK 1

The error checksum CRC CHK (0,1) can be calculated by CRC instruction (8-bit mode, M1161 = ON).



CRC checksum: 66 H is stored in low byte of D106 and AB H in low byte of of D107

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
113	ETHRW	(S1) (S2) (D) (n)	Ethernet communication				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1													*			ETHRW: 9 steps
S_2					*	*							*			
D													*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2	SX2	ES2/EX2/ EC5	SS2	SE/ ES2-E	SX2/ SA2	ES2/EX2/ EC5	SS 2	SA2	SX2

Operands:

S_1: IP address, communication port number, and read/write mode **S_2**: Device address **D**: Source/Destination data register **n**: Data length; Range: K1~K96 (word), K1~K256 (bit)

Explanations:

1. **S_1**: IP address, communication port number, and read/write mode

   The operand **S_1** occupies five consecutive data registers. The functions are as follows.

   - IP address: Two data registers are occupied, that is, **S_1**+0 and **S_1**+1.

   IP address→IP3.IP2.IP1.IP0→192.168.0.2

   If **S_1** is D100, the values in D100 and D101 are H’0002 and H’C0A8 respectively.

D100 (**S_1**+0)		D101 (**S_1**+1)	
High	Low	High	Low
IP1	IP0	IP3	IP2
0	2	192	168
H’0002		H’C0A8	

   - **S_1**+2: Communication port number

     The communication port number of the Ethernet port on DVP-SE/ES2-E and that of the communication card installed in DVP-EH3 are K108. The communication ports on the left-side Ethernet modules connected to a CPU module are numbered according to their distances from the CPU module. The numbers start from K100 to K107.

   - **S_1**+3: Station address of a slave
   - **S_1**+4: Read/Write function code setting

     The definition is the same as Modbus. The function codes supported are H’01, H’02, H’03, H’04, H’05, H’06, H’0F and H’10.

2. **S_2**: Device address

   The definition is the same as Modbus.

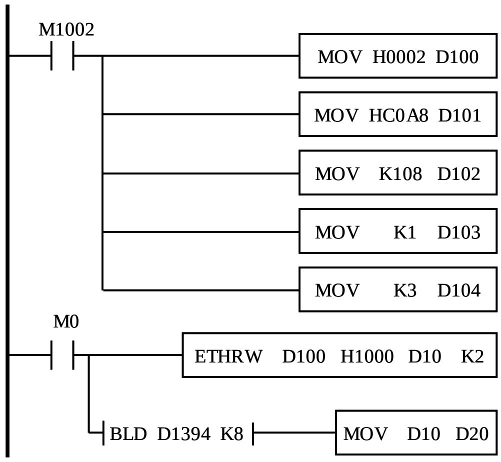
3. The operand **D** specifies a source data register or a destination data register. For example, D specifies D10 and set the function code to H’03; when it reads 2 length of data, the data will be stored in D10 and D11.

4. When setting the function code to H’05, 0 in the operand D means to Reset bit and for other values in the operand D means to Set bit.
5. **n**: Length of data (Unit: word, the setting range: K1~K96) (Unit: bit, the setting range: K1~K256) If **n** exceeds the range, it will be taken as the maximum value or the minimum value.
6. Whenever the instruction is executed, the communication command is sent. Users do not need to enable a special flag to send the communication command.
7. The instruction can be used several times. However, if an ETHRW instruction specifies a module, other ETHRW instructions cannot send communication commands to the module. The next communication command cannot be sent until the reception is complete or the module replies that an error occurs.
8. If a communication command is being received, the reception stops when the execution of the instruction stops. Besides, the flag related to the command’s having being received and the error flag are not ON.
9. The communication timeout is stored in D1349. The default timeout is 3000 milliseconds. The range of digital values is 1~32767. If the communication timeout exceeds the range, it will be taken as 3000 milliseconds.
10. The values of bit0~bit8 in D1395 indicate which communication port has received a command. For example, if the communication port built in DVP-SE has received a command, “BLD D1395 K8” is satisfied.
11. The values of bit0~bit8 in D1396 indicate which module experience an error. For example, if a reception error occurs in the first left-side DVP-EN01, “BLD D1396 K0” is satisfied.
12. D1111 is the setting of Modbus communication port; the supported range is 1 to 65535. If the value 0 is used, it will be seen as the default value 502. It is available for ES2-E V1.46 and SE V2.04.
13. When the instruction is executed, user can not use the online editing function. Otherwise, the data received will not be stored correctly.
14. SA2/SX2 v2.62, SE/ES2-E v1.00 and later versions support the function codes H’03, H’04, H’06, and H’10.
15. SE v1.86, ES2-E v1.00 and later versions support the function codes H’01, H’02, H’05, and H’0F.

Program Example 1:

(The instruction is sent and received through the Ethernet port built in DVP-SE.)

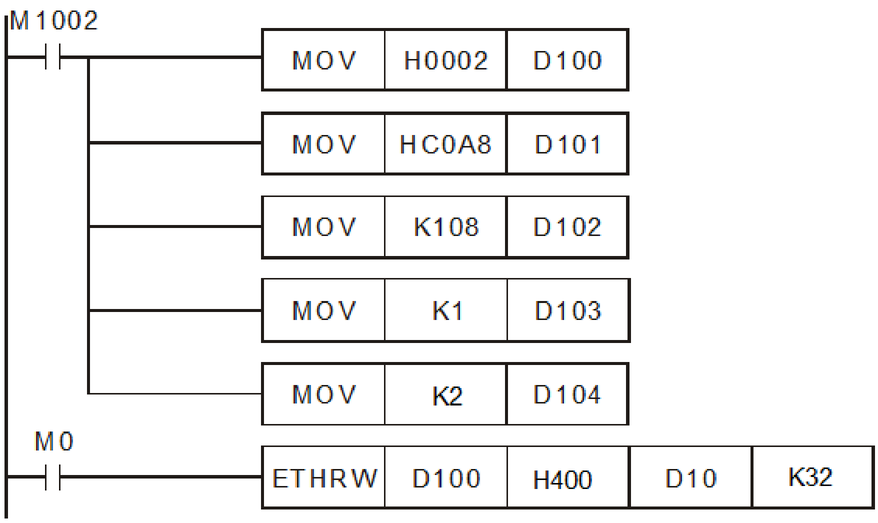
The IP address stored in D100 and D101 is 192.168.0.2, the communication port number stored in D102 is K108, the station address stored in D103 is K1, and the function code stored in D104 is H'03. The device address is H'1000, and two pieces of data are read. When M0 is ON, ETHRW is executed. After the reception of the communication command is complete, bit8 in D1394 is ON. The data received is stored in D10 and D11.



Program Example 2:

(The instruction is sent and received through the Ethernet port built in DVP-SE.)

The IP address stored in D100 and D104 is 192.168.0.2, the communication port number stored in D102 is K108, the station address stored in D103 is K1, and the function code stored in D104 is H'02. The device address is H0400 (X0), and 32 pieces of bit data (X0~X37) are read. When M0 is ON, ETHRW is executed. After the reception of the communication command is complete, bit8 in D1395 is ON. The data received is stored in D10: high byte (X0~X7) and low byte (X10~X17) and D11: low byte (X0~X27) and high byte (X30~X37).



API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
150	MODRW	(S1) (S2) (S3) (S) (n)	MODBUS Read/ Write				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			MODRW: 11 steps
S_2					*	*							*			
S_3					*	*							*			
S													*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Device address (K0~K254) **S_2**: Function code: K1(H01), K2(H02), K3(H03), K4(H04), K5(H05), K6(H06), K15(H0F), K16(H10), K23(H17) **S_3**: The function varies with the function code used. **S**: The function varies with the function code used. **n**: The function varies with the function code used.

Explanations:

1. MODRW supports COM1 (RS-232), COM2 (RS-485), COM3 (RS-485). (COM3 is only applicable to DVP-ES2/EX2/SA2/SE, and is not applicable to DVP-ES2-C.)
2. **S_1**: Address of the device to be accessed. Range: K0~K254. The address specified by the function codes K1, K2, K3, K4, and K23 can not be K0.
3. **S_2**: Function code. Only the function codes listed below are available currently; other function codes are not executable. Please refer to the program examples below for more information.

Function code	Description	Models supported
H01	Reading multiple bit devices	ES2/EX2 V3.28, SS2 V3.24, SA2/SX2 V2.82, and SE V1.64 (and above)
H02	Reading multiple bit devices	All series
H03	Reading multiple word devices	All series
H04	Reading multiple word devices	ES2/EX2 V2.6, SS2 V2.4, SA2/SX2 V2.0, and SE V1.0 (and above)
H05	Writing in a single bit device	All series
H06	Writing in a single word device	All series
H0F	Writing in multiple bit devices	All series
H10	Writing in multiple word devices	All series
H17	Reading/Writing in multiple word devices	ES2/EX2 V3.2, SS2 V3.0, SA2 V2.6, and SX2 V2.4 (and above)

4. **S_3**: Address of the data to be accessed. If the address is illegal for the designated communication device, the communication device will respond with an error message and DVP-PLC will store the error code and associated error flag will be ON. If the function code is K23, **S_3** only can specify a data register. Besides, **S_3** is a data register from which data is read, **S_3**+1 is a data register into which data is written.
   - Associated registers and flags indicating errors on PLC com ports: (For detailed information please refer to **Points to note** of API 80 RS instruction.)

PLC COM	COM1	COM2	COM3
Error flag	M1315	M1141	M1319
Error code	D1250	D1130	D1253

   - For example, if 8000H is illegal for DVP-PLC, the error will be in indicated by different set of flags and registers. For COM2, M1141 will be ON and D1130 = 2; for COM1, M1315 = ON and D1250 = 3, for COM3, M1319 = ON and D1253 = 3. Please check the user manual of DVP-PLC for error code explanations.
5. **S**: Registers for storing read/written data. Registers starting from **S** stores the data to be written into the communication device or the data read from the communication device. If the function code K23 is used, **S** is a D device index which indicates the device in which the communication data string received will be stored, and **S**+1 is a D device index which indicates the device in which the data which will be written is stored. If a reading function code (K2, K3, K4, or K23) is sent through COM2, the communication data string received will be stored in the register indicated by **S**, and the conversion data will be stored in D1296~D1311. Please refer to program example 1 and program example 3 for more information. If a reading function code (K2, K3, K4, or K23) is sent through COM1 or COM3, the conversion data will be stored in the register indicated by **S**. Please refer to program example 2 and program example 4 for more information. Users can refer to example 13 and example 14 for more information about the function code K23.
6. **n**: Data length for accessing.
   - When **S_2** (MODBUS function code) is specified as H05 which designates the PLC force ON/OFF status, **n** = 0 indicates ON and **n** = 1 indicates OFF.
   - When **S_2** is specified as H01, H02, H03, H04, H0F, H10, H17 which designate the data length for accessing, the available set range will be **K1~Km**, where **m** value should be specified according to communication modes and COM ports as the table below. (H01/H02/H0F, unit: Bit. H03/H04/H10/H17, unit: Word.) If the function code is H17, n is the number of data registers from which data is read, n+1 is the number of data registers into which data is written.

Communication mode	Communication port	Function code				
		H01/H02	H03/H04	H0F	H10	H17
RTU	COM1(RS-232)	K 64	K 16	K 64	K 16	K 16
	COM2(RS-485)	K 64	K 16	K 64	K 16	K 16
	COM3(RS-485)	K 64	K 16	K 64	K 16	K 16
ASCII	COM1(RS-232)	K 64	K 16	K 64	K 16	K 16
	COM2(RS-485)	K 64	K 8	K 64	K 8	K 16
	COM3(RS-485)	K 64	K 16	K 64	K 16	K 16

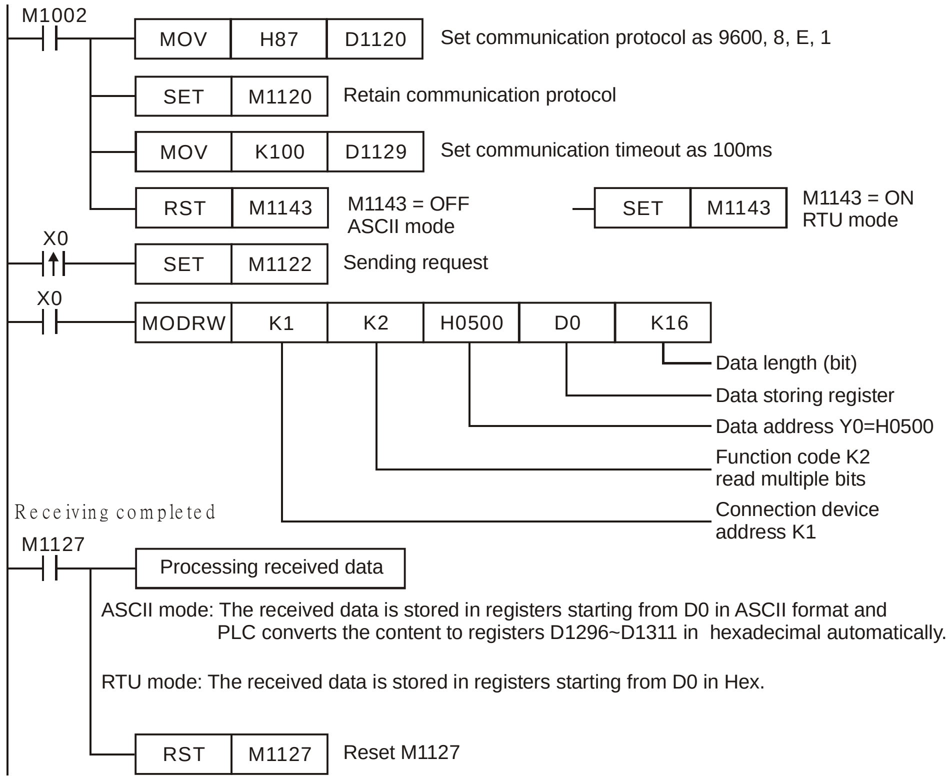
7. The functions of $\mathbf{S_3}$, **S**, and **n** vary with the function code used.

Function code	$\mathbf{S_3}$	S	n
H01	Address from which the data is read	Register in which the data read is stored	Length of data read
H02	Address from which the data is read	Register in which the data read is stored	Length of data read
H03	Address from which the data is read	Register in which the data read is stored	Length of data read
H04	Address from which the data is read	Register in which the data read is stored	Length of data read
H05	Address into which the data is written	No meaning	Status value written
H06	Address into which the data is written	Register in which the data written is stored	No meaning
H0F	Address into which the data is written	Register in which the data written is stored	Length of data written
H10	Address into which the data is written	Register in which the data written is stored	Length of data written
H17	$\mathbf{S_3}$: Address from which the data is read $\mathbf{S_3}$+1: Address into which the data is written	**S**: Register in which the data read is stored **S**+1: Register in which the data written is stored	**n**: Length of data read **n**+1: Length of data written

8. There is no limitation on the times of using this instruction, however only one instruction can be executed on the same COM port at a time.
9. Rising-edge contact (LDP, ANDP, ORP) and falling-edge contact (LDF, ANDF, ORF) can not be used as drive contact of MODRW (Function code H01, H02, H03, H04, H17) instruction, otherwise the data stored in the receiving registers will be incorrect.
10. MODRW instruction determines the COM port according to the communication request. The COM port determination is made following the order: COM1→COM3→COM2. Therefore, please insert every MODRW instruction right after the sending request instruction for avoiding errors on the target location for data access.
11. For detailed explanation of the associated flags and special registers, please refer to **Points to note** of API 80 RS instruction.

Program Example 1: COM2(RS-485), Function Code H02 (H01 is used the same as H02.)

1. Function code K2 (H02): read multiple bit devices, up to 64 bits can be read.
2. PLC1 connects to PLC2: (M1143 = OFF, ASCII mode), (M1143 = ON, RTU Mode)
3. In ASCII or RTU mode, when PLC's COM2 sends out data, the data will be stored in D1256~D1295. The feedback data will be stored in registers starting with **S** and converted into D1296~D1311 in Hex automatically.
4. Take the connection between PLC1 (PLC COM2) and PLC2(PLC COM1) for example, the tables below explains the status when PLC1 reads Y0~Y17 of PLC2.



ASCII Mode (M1143 = OFF):

When X0 = ON, MODRW instruction executes the function specified by Function Code 02.

PLC1⇨ PLC2，PLC1 sends: "**01 02 0500 0010 E8**"

PLC2 ⇨PLC1，PLC1 receives: **"01 02 02 3412 B5**"

Registers for data to be sent (sending messages)

Register	Data		Descriptions	
D1256 Low	'0'	30 H	ADR 1	Device address: ADR (1,0)
D1256 High	'1'	31 H	ADR 0	
D1257 Low	'0'	30 H	CMD 1	Control parameter: CMD (1,0)
D1257 High	'2'	32 H	CMD 0	
D1258 Low	'0'	30 H	Y0 = H0500 Starting Data Address	
D1258 High	'5'	35 H		
D1259 Low	'0'	30 H		
D1259 High	'0'	30 H		
D1260 Low	'0'	30 H	Number of Data(count by bit)	

Register	Data		Descriptions	
D1260 High	'0'	30 H		
D1261 Low	'1'	31 H		
D1261 High	'0'	30 H		
D1262 Low	'E'	45 H	LRC CHK 1	Checksum: LRC CHK (0,1)
D1262 High	'8'	38 H	LRC CHK 0	

Registers for received data (responding messages)

Register	Data		Descriptions	
D0 Low	'0'	30 H	ADR 1	
D0 High	'1'	31 H	ADR 0	
D1 Low	'0'	30 H	CMD 1	
D1 High	'2'	33 H	CMD 0	
D2 Low	'0'	30 H	Number of Data (count by Byte)	
D2 High	'2'	32 H		
D3 Low	'3'	33 H	Content of address 0500H~ 0515H	1234 H PLC automatically converts ASCII codes and store the converted value in D1296
D3 High	'4'	34 H		
D4 Low	'1'	31H		
D4 High	'2'	32H		
D5 Low	'B'	52H	LRC CHK 1	
D5 High	'5'	35 H	LRC CHK 0	

Analysis of the read status of PLC2 Y0~Y17: 1234H

Device	Status	Device	Status	Device	Status	Device	Status
Y0	OFF	Y1	OFF	Y2	ON	Y3	OFF
Y4	ON	Y5	ON	Y6	OFF	Y7	OFF
Y10	OFF	Y11	ON	Y12	OFF	Y13	OFF
Y14	ON	Y15	OFF	Y16	OFF	Y17	OFF

RTU Mode (M1143 = ON):

When X0 = ON, MODRW instruction executes the function specified by Function Code 02

PLC1⇨ PLC2，PLC1sends: "**01 02 0500 0010 79 0A**"

PLC2 ⇨ PLC1，PLC1receives: "**01 02 02 34 12 2F 75**"

Registers for data to be sent (sending messages)

Register	Data	Descriptions
D1256 Low	01 H	Address
D1257 Low	02 H	Function
D1258 Low	05 H	Y0 = H0500 Starting Data Address
D1259 Low	00 H	
D1260 Low	00 H	Number of Data (count by word)
D1261 Low	10 H	
D1262 Low	79 H	CRC CHK Low
D1263 Low	0A H	CRC CHK High

Registers for received data (responding messages)

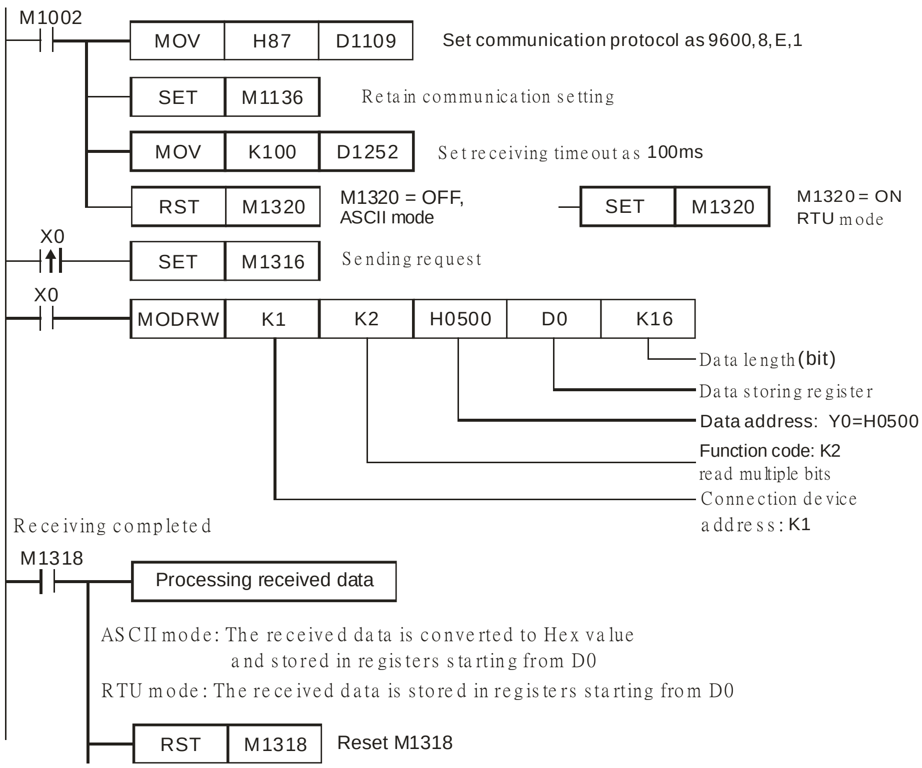
Register	Data	Descriptions
D0 Low	01 H	Address
D1 Low	02 H	Function
D2 Low	02 H	Number of Data (Byte)
D3 Low	34 H	Content of address H0500~H0515
D4 Low	12 H	
D5 Low	2F H	CRC CHK Low
D6 Low	75 H	CRC CHK High

Analysis of the read status of PLC2 Y0~Y17: 1234H

Device	Status	Device	Status	Device	Status	Device	Status
Y0	OFF	Y1	OFF	Y2	ON	Y3	OFF
Y4	ON	Y5	ON	Y6	OFF	Y7	OFF
Y10	OFF	Y11	ON	Y12	OFF	Y13	OFF
Y14	ON	Y15	OFF	Y16	OFF	Y17	OFF

Program Example 2: COM1(RS-232) / COM3(RS-485), Function Code H02 (H01 is used the same as H02.)

1. Function code K2 (H02): read multiple bit devices. Up to 64 bits can be read.
2. PLC1 connects to PLC2: (M1320 = OFF, ASCII mode), (M1320 = ON, RTU mode)
3. For both ASCII and RTU modes, PLC COM1/COM3 only stores the received data in registers starting from **S**, and will not store the data to be sent. The stored data can be transformed and moved by using DTM instruction for applications of other purposes.
4. Take the connection between PLC1 (PLC COM3) and PLC2(PLC COM1) for example, the tables below explains the status when PLC1 reads Y0~Y17 of PLC2
   - If PLC1 applies COM1 for communication, the below program can be usable by changing:
     a) D1109→D1036: communication protocol
     b) M1136→M1138: retain communication setting
     c) D1252→D1249: Set value for data receiving timeout
     d) M1320→M1139: ASCII/RTU mode selection
     e) M1316→M1312: sending request
     f) M1318→M1314: receiving completed flag



- ASCII mode (COM3: M1320 = OFF, COM1: M1139 = OFF):

  When X0 = ON, MODRW instruction executes the function specified by Function Code 02

  PLC1⇨ PLC2, PLC1 sends: "**01 02 0500 0010 E8**"

  PLC2 ⇨PLC1, PLC1 receives: "**01 02 02 3412 B5**"

  PLC1 data receiving register D0

Register	Data	Descriptions
D0	1234H	PLC converts the ASCII data in address 0500H~0515H and stores the converted data automatically.

Analysis of the read status of PLC2 Y0~Y17: 1234H

Device	Status	Device	Status	Device	Status	Device	Status
Y0	OFF	Y1	OFF	Y2	ON	Y3	OFF
Y4	ON	Y5	ON	Y6	OFF	Y7	OFF
Y10	OFF	Y11	ON	Y12	OFF	Y13	OFF
Y14	ON	Y15	OFF	Y16	OFF	Y17	OFF

- RTU mode (COM3: M1320 = ON, COM1: M1139 = ON):

  When X0 = ON, MODRW instruction executes the function specified by Function Code 02

  PLC1 ⇨ PLC2, PLC1 sends: “**01 02 0500 0010 79 0A**”

  PLC2 ⇨ PLC1, PLC1 receives: “**01 02 02 34 12 2F 75**”

  PLC data receiving register:

Register	Data	Descriptions
D0	1234 H	PLC converts the data in address 0500H ~ 0515H and stores the converted data automatically.

  Analysis of the read status of PLC2 Y0~Y17: 1234H

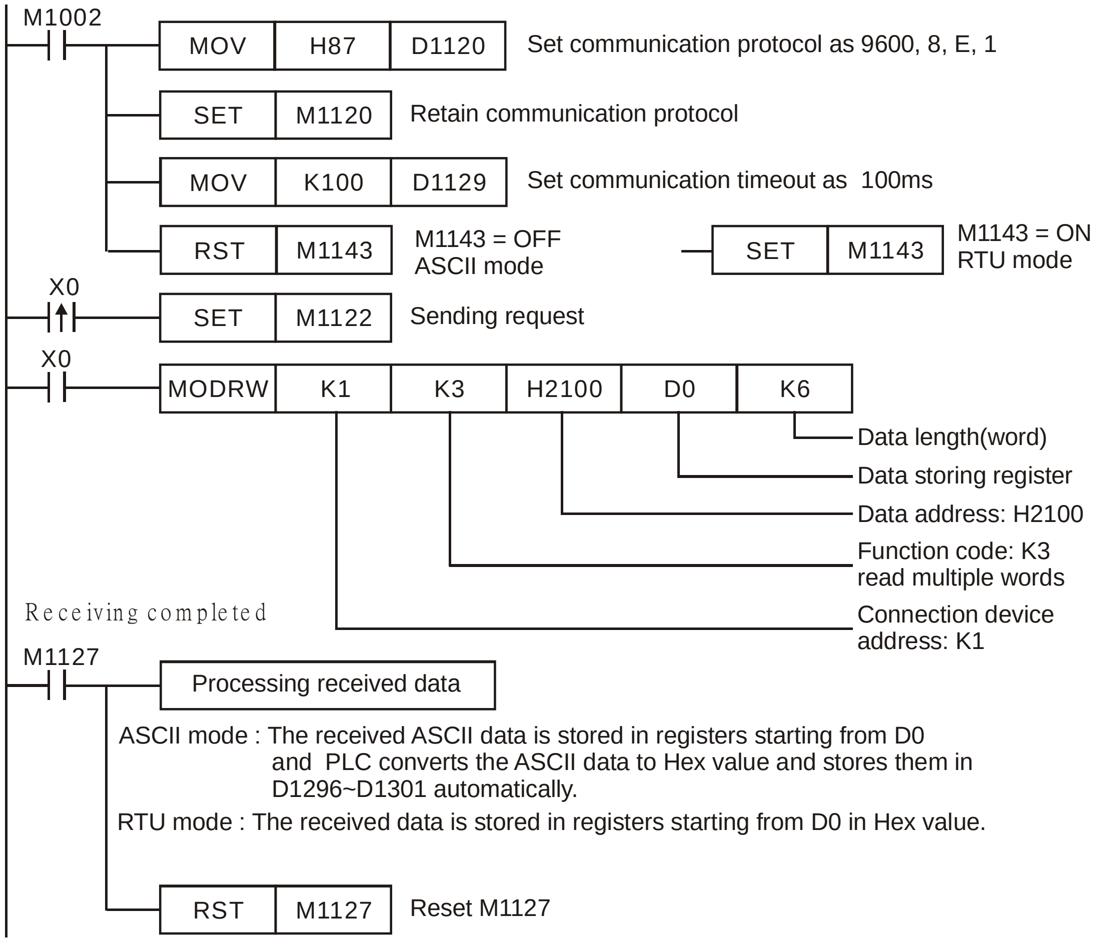
Device	Status	Device	Status	Device	Status	Device	Status
Y0	OFF	Y1	OFF	Y2	ON	Y3	OFF
Y4	ON	Y5	ON	Y6	OFF	Y7	OFF
Y10	OFF	Y11	On	Y12	OFF	Y13	OFF
Y14	ON	Y15	OFF	Y16	OFF	Y17	OFF

5. Relative flags and data registers when COM1 / COM2 / COM3 works as Master:

	COM2	COM1	COM3	Function
COM. setting	M1120	M1138	M1136	Retain communication setting
	M1143	M1139	M1320	ASCII/RTU mode selection
	D1120	D1036	D1109	Communication protocol
	D1121	D1121	D1255	PLC communication address
Sending request	M1122	M1312	M1316	Sending request
	D1129	D1249	D1252	Set value for data receiving timeout (ms)
Receiving completed	M1127	M1314	M1318	Data receiving completed
Errors	-	M1315	M1319	Data receiving error
	-	D1250	D1253	Communication error code
	M1129	-	-	Receiving timeout
	M1140	-	-	Data receiving error
	M1141	-	-	Parameter error. Exception Code is stored in D1130
	D1130	-	-	Error code (Exception code) returning from Modbus communication

Program Example 3: COM2 (RS-485), Function Code H03 (The function code H04 is the same as the function code H03.)

1. Function code K3 (H03): read multiple Word devices. Up to 16 words can be read. For COM2 ASCII mode, only 8 words can be read.
2. For ASCII or RTU mode, PLC COM2 stores the data to be sent in D1256~D1295, converts the received data in registers starting from **S**, and stores the converted 16-bit data in D1296 ~ D1311.
3. Take the connection between PLC (PLC COM2) and VFD-B for example, the tables below explains the status when PLC reads status of VFD-B. (M1143 = OFF, ASCII Mode) (M1143 = ON, RTU Mode)



ASCII mode (M1143 = OFF):

When X0 = ON, MODRW instruction executes the function specified by Function Code 03

PLC ⇨ VFD-B, PLC sends: **"01 03 2100 0006 D5"**

VFD-B ⇨ PLC, PLC receives: **"01 03 0C 0100 1766 0000 0000 0136 0000 3B"**

Registers for data to be sent (sending messages)

Register	Data		Descriptions	
D1256 Low byte	‘0’	30 H	ADR 1	Address of VFD-B: ADR (1,0)
D1256 High byte	‘1’	31 H	ADR 0	
D1257 Low byte	‘0’	30 H	CMD 1	Control parameter: CMD (1,0)
D1257 High byte	‘3’	33 H	CMD 0	
D1258 Low byte	‘2’	32 H	Data Address	
D1258 High byte	‘1’	31 H		
D1259 Low byte	‘0’	30 H		
D1259 High byte	‘0’	30 H		

Register	Data		Descriptions	
D1260 Low byte	‘0’	30 H	Number of data (count by word)	
D1260 High byte	‘0’	30 H		
D1261 Low byte	‘0’	30 H		
D1261 High byte	‘6’	36 H		
D1262 Low byte	‘D’	44 H	LRC CHK 1	Checksum: LRC CHK (0,1)
D1262 High byte	‘5’	35 H	LRC CHK 0	

Registers for received data (responding messages)

Register	Data		Descriptions	
D0 low byte	‘0’	30 H	ADR 1	
D0 high byte	‘1’	31 H	ADR 0	
D1 low byte	‘0’	30 H	CMD 1	
D1 high byte	‘3’	33 H	CMD 0	
D2 low byte	‘0’	30 H	Number of data (count by byte)	
D2 high byte	‘C’	43 H		
D3 low byte	‘0’	30 H	Content of address H2100	0100 H PLC COM2 automatically converts ASCII codes to Hex and stores the converted value in D1296
D3 high byte	‘1’	31 H		
D4 low byte	‘0’	30 H		
D4 high byte	‘0’	30 H		
D5 low byte	‘1’	31 H	Content of address H2101	1766 H PLC COM2 automatically converts ASCII codes to Hex and stores the converted value in D1297
D5 high byte	‘7’	37 H		
D6 low byte	‘6’	36 H		
D6 high byte	‘6’	36 H		
D7 low byte	‘0’	30 H	Content of address H2102	0000 H PLC COM2 automatically converts ASCII codes to hex and stores the converted value in D1298
D7 high byte	‘0’	30 H		
D8 low byte	‘0’	30 H		
D8 high byte	‘0’	30 H		
D9 low byte	‘0’	30 H	Content of address H2103	0000 H PLC COM2 automatically converts ASCII codes to hex and stores the converted value in D1299
D9 high byte	‘0’	30 H		
D10 low byte	‘0’	30 H		
D10 high byte	‘0’	30 H		

Register	Data		Descriptions	
D11 low byte	‘0’	30 H	Content of address H2104	0136 H PLC COM2 automatically converts ASCII codes to hex and stores the converted value in D1300
D11 high byte	‘1’	31 H		
D12 low byte	‘3’	33 H		
D12 high byte	‘6’	36 H		

D13 low byte	'0'	30 H	Content of address H2105	0000 H PLC COM2 automatically converts ASCII codes to hex and stores the converted value in D1301
D13 high byte	'0'	30 H		
D14 low byte	'0'	30 H		
D14 high byte	'0'	30 H		
D15 low byte	'3'	33 H	LRC CHK 1	
D15 high byte	'B'	42 H	LRC CHK 0	

RTU mode (M1143 = ON):

When X0 = ON, MODRW instruction executes the function specified by Function Code 03

PLC ⇨ VFD-B, PLC sends: **" 01 03 2100 0006 CF F4"**

VFD-B ⇨ PLC, PLC receives: **"01 03 0C 0000 0503 0BB8 0BB8 0000 012D 8E C5"**

Registers for data to be sent (sending messages)

Register	Data	Descriptions
D1256 Low byte	01 H	Address
D1257 Low byte	03 H	Function
D1258 Low byte	21 H	Data Address
D1259 Low byte	00 H	
D1260 Low byte	00 H	Number of data (count by word)
D1261 Low byte	06 H	
D1262 Low byte	CF H	CRC CHK Low
D1263 Low byte	F4 H	CRC CHK High

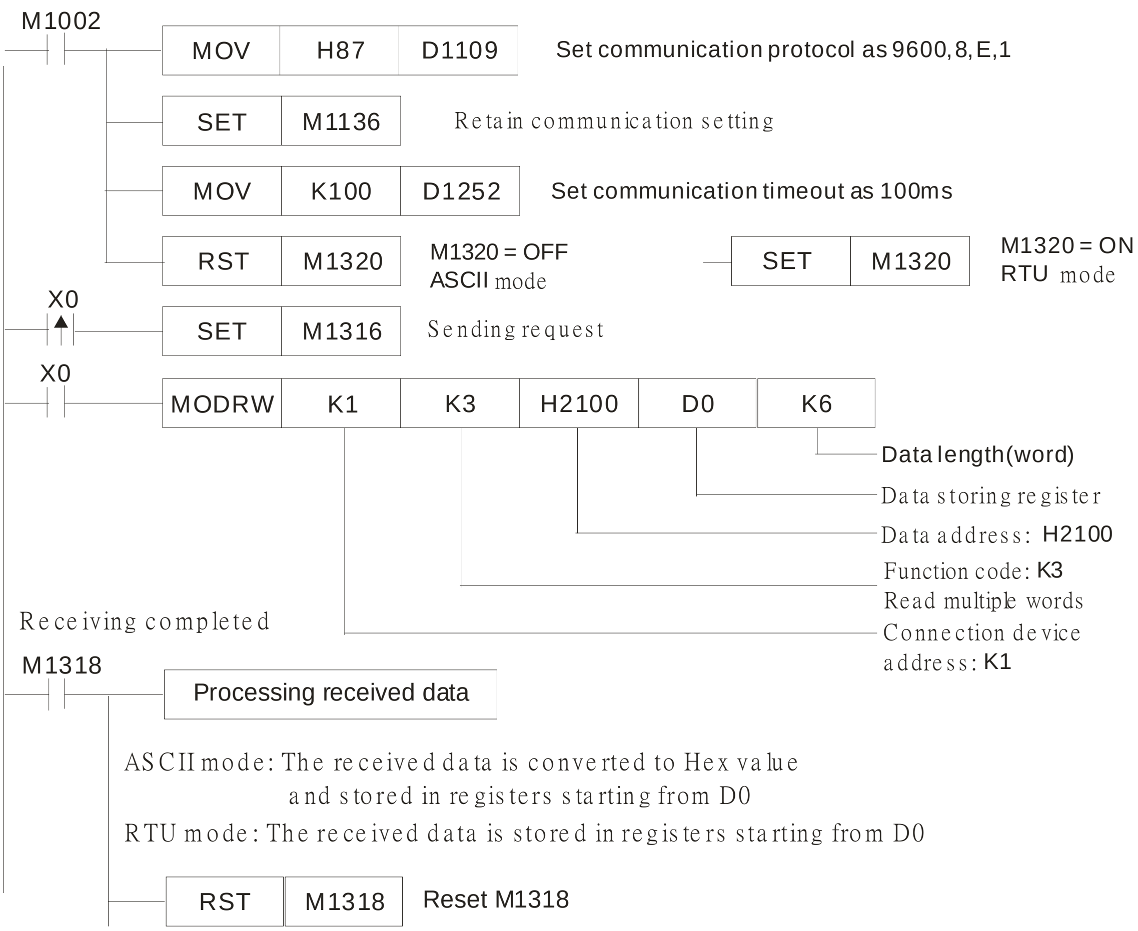
Registers for received data (responding messages)

Register	Data	Descriptions	
D0 low byte	01 H	Address	
D1 low byte	03 H	Function	
D2 low byte	0C H	Number of data (count by byte)	
D3 low byte	00 H	Content of address H2100	0000 H PLC COM2 automatically stores the value in D1296
D4 low byte	00 H		
D5 low byte	05 H	Content of address H2101	0503 H PLC COM2 automatically store the value in D1297
D6 low byte	03 H		
D7 low byte	0B H	Content of address H2102	0BB8 H PLC COM2 automatically stores the value in D1298
D8 low byte	B8 H		
D9 low byte	0B H	Content of address H2103	0BB8 H PLC COM2 automatically store the value in D1299
D10 low byte	B8 H		
D11 low byte	00 H	Content of address H2104	0000 H PLC COM2 automatically store the value in D1300
D12 low byte	00 H		
D13 low byte	01 H	Content of address H2105	012D H PLC COM2 automatically store the value in D1301
D14 low byte	2D H		
D15 low byte	8E H	CRC CHK Low	
D16 low byte	C5 H	CRC CHK High	

Program example 4: COM1(RS-232) / COM3(RS-485), Function Code H03 (The function code H04 is the same as the function code H03.)

1. Function code K3 (H03): read multiple Word devices, up to 16 words can be read. For COM2 ASCII mode, only 8 words can be read..
2. PLC COM1 / COM3 stores the received data in registers starting from **S**, and the stored data can be transformed and moved by using DTM instruction for applications of other purposes.
3. Take the connection between PLC and VFD-B for example, the tables below explains the status when PLC reads VFD-B status. (M1320 = OFF, ASCII Mode), (M1320 = ON, RTU Mode)

- If PLC applies COM1 for communication, the below program can be usable by changing:
  - a) D1109→D1036: communication protocol
  - b) M1136→M1138: retain communication setting
  - c) D1252→D1249: Set value for data receiving timeout
  - d) M1320→M1139: ASCII/RTU mode selection
  - e) M1316→M1312: sending request
  - f) M1318→M1314: receiving completed flag



ASCII mode (COM3: M1320 = OFF, COM1: M1139 = OFF):

When X0 = ON, MODRW instruction executes the function specified by Function Code 03

PLC ⇨ VFD-B, PLC sends: "**01 03 2100 0006 D5**"

VFD-B ⇨ PLC, PLC receives: "**01 03 0C 0100 1766 0000 0000 0136 0000 3B**"

Registers for received data (responding messages)

Register	Data	Descriptions
D0	0100 H	PLC converts ASCII codes in 2100 H and stores the converted data automatically.
D1	1766 H	PLC converts ASCII codes in 2101 H and stores the converted data automatically.
D2	0000 H	PLC converts ASCII codes in 2102 H and stores the converted data automatically.
D3	0000 H	PLC converts ASCII codes in 2103 H and stores the converted data automatically.
D4	0136 H	PLC converts ASCII codes in 2104 H and stores the converted data automatically.
D5	0000 H	PLC converts ASCII codes in 2105 H and stores the converted data automatically.

RTU mode (COM3: M1320 = ON COM1: M1139 = ON):

When X0 = ON, MODRW instruction executes the function specified by Function Code 03

PLC ⇨ VFD-B, PLC sends: " **01 03 2100 0006 CF F4**"

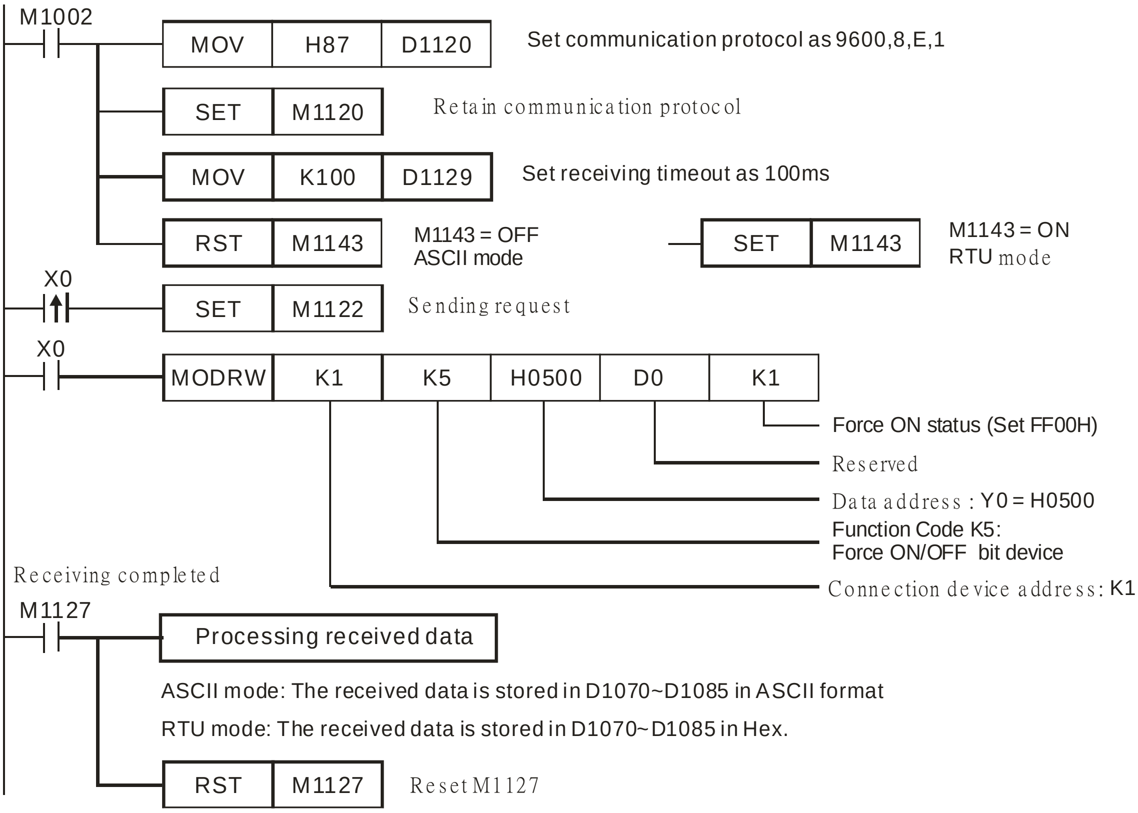
VFD-B ⇨ PLC, PLC receives: "**01 03 0C 0000 0503 0BB8 0BB8 0000 012D 8E C5**"

Registers for received data (responding messages)

Register	Data	Descriptions
D0	0000 H	PLC converts data in 2100 H and stores the converted data automatically.
D1	0503 H	PLC converts data in 2101 H and stores the converted data automatically.
D2	0BB8 H	PLC converts data in 2102 H and stores the converted data automatically.
D3	0BB8 H	PLC converts data in 2103 H and stores the converted data automatically.
D4	0136 H	PLC converts data in 2104 H and stores the converted data automatically.
D5	012D H	PLC converts data in 2105 H and stores the converted data automatically.

Program example 5: COM2(RS-485), Function Code H05

1. Function code K5(H05): Force ON/OFF bit device
2. PLC1 connects to PLC2: (M1143 = OFF, ASCII mode), (M1143 = ON, RTU Mode)
3. **n** = 1 indicates Force ON (set FF00H) and **n** = 0 indicates Force OFF (set 0000H)
4. For ASCII or RTU mode, PLC COM2 stores the data to be sent in D1256~D1295 and stores the received data in D1070~D1085
5. Take the connection between PLC1 (PLC COM2) and PLC2 (PLC COM1) for example, the tables below explain the status when PLC1 Force ON PLC2 Y0.



ASCII mode (M1143 = OFF):

When X0 = ON, MODRW instruction executes the function specified by Function Code 05

PLC1 ⇨ PLC2, PLC sends: "**01 05 0500 FF00 6F**"

PLC2 ⇨ PLC1, PLC receives: "**01 05 0500 FF00 6F**"

Registers for data to be sent (sending messages)

Register	Data		Descriptions	
D1256 low byte	'0'	30 H	ADR 1	Device address: ADR (1,0)
D1256 high byte	'1'	31 H	ADR 0	
D1257 low byte	'0'	30 H	CMD 1	CMD (1,0) Control parameter
D1257 high byte	'5'	35H	CMD 0	
D1258 low byte	'0'	30 H	Data Address	
D1258 high byte	'5'	35 H		
D1259 low byte	'0'	30 H		
D1259 high byte	'0'	30 H		
D1260 low byte	'F'	46 H	High byte to be force ON/OFF	

Register	Data		Descriptions	
D1260 high byte	'F'	46 H		
D1261 low byte	'0'	30H	Low byte to be force ON/OFF	
D1261 high byte	'0'	30 H		
D1262 low byte	'6'	36 H	LRC CHK 1	Checksum: LRC CHK (0,1)
D1262 high byte	'F'	46 H	LRC CHK 0	

Registers for received data (responding messages)

Register	Data		Descriptions
D1070 low byte	'0'	30 H	ADR 1
D1070 high byte	'1'	31 H	ADR 0
D1071 low byte	'0'	30 H	CMD 1
D1071 high byte	'5'	35H	CMD 0
D1072 low byte	'0'	30 H	Data Address
D1072 high byte	'5'	35 H	
D1073 low byte	'0'	30 H	
D1073 high byte	'0'	30 H	
D1074 low byte	'F'	46 H	High byte to be force ON/OFF
D1074 high byte	'F'	46 H	
D1075 low byte	'0'	30H	Low byte to be force ON/OFF
D1075 high byte	'0'	30 H	
D1076 low byte	'6'	36 H	LRC CHK 1
D1076 high byte	'F'	46 H	LRC CHK 0

RTU mode (M1143 = ON)

When X0 = ON, MODRW instruction executes the function specified by Function Code 05

PLC1⇨ PLC2, PLC1 sends: "**01 05 0500 FF00 8C F6**"

PLC2 ⇨PLC1, PLC1 receives: "**01 05 0500 FF00 8C F6**"

Registers for data to be sent (sending messages)

Register	Data	Descriptions
D1256 Low byte	01 H	Address
D1257 Low byte	05 H	Function
D1258 Low byte	05 H	Data Address
D1259 Low byte	00 H	
D1260 Low byte	FF H	Data content (ON = FF00H)
D1261 Low byte	00 H	
D1262 Low byte	8C H	CRC CHK Low
D1263 Low byte	F6 H	CRC CHK High

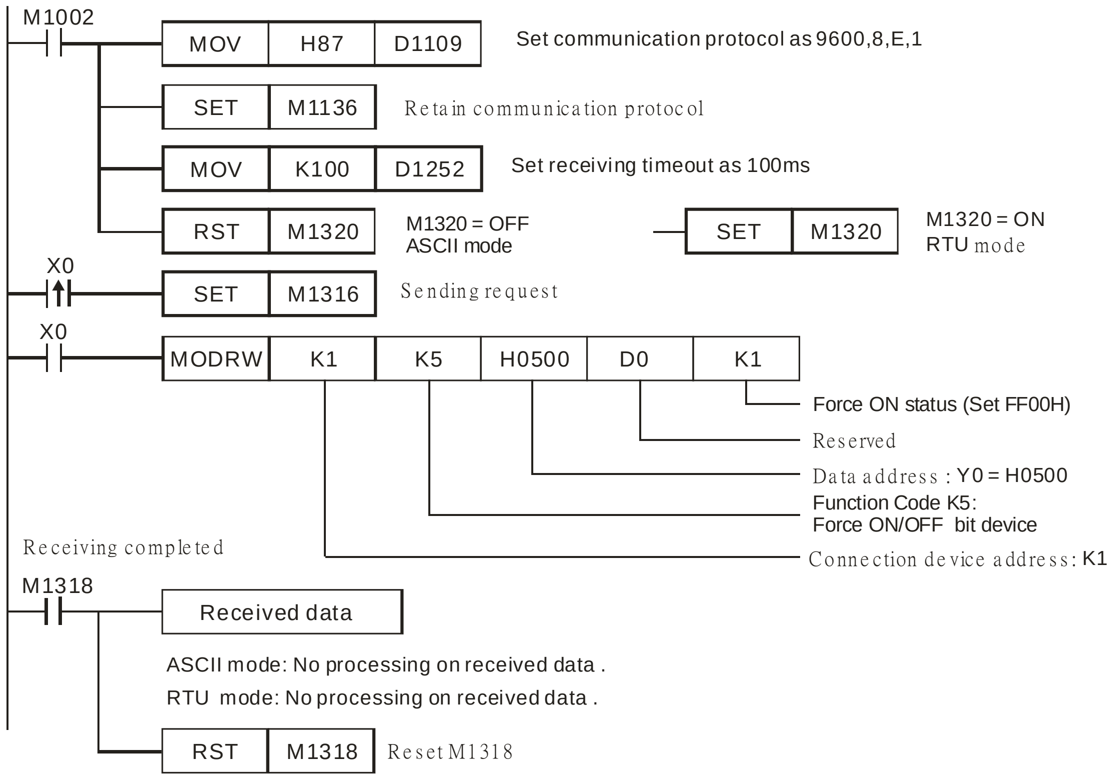
Registers for received data (responding messages)

Register	Data	Descriptions
D1070 Low byte	01 H	Address
D1071 Low byte	05 H	Function
D1072 Low byte	05 H	Data Address
D1073 Low byte	00 H	
D1074 Low byte	FF H	Data content (ON = FF00H)
D1075 Low byte	00 H	
D1076 Low byte	8C H	CRC CHK Low
D1077 Low byte	F6 H	CRC CHK High

Program example 6: COM1(RS-232) / COM3(RS-485), Function Code H05

1. Function Code K5 (H05): Force ON/OFF bit device.
2. PLC1 connects PLC2: (M1320 = OFF, ASCII Mode), (M1320 = ON, RTU Mode)
3. **n** = 1 indicates Force ON (set FF00H) and **n** = 0 indicates Force OFF (set 0000H)
4. PLC COM1/COM3 will not process the received data.
5. Take the connection between PLC1 (PLC COM3) and PLC2(PLC COM1) for example, the tables below explains the status when PLC1 reads Y0~Y17 of PLC2

- If PLC1 applies COM1 for communication, the below program can be usable by changing:
  a) D1109→D1036: communication protocol
  b) M1136→M1138: retain communication setting
  c) D1252→D1249: Set value for data receiving timeout
  d) M1320→M1139: ASCII/RTU mode selection
  e) M1316→M1312: sending request
  f) M1318→M1314: receiving completed flag



ASCII mode (COM3: M1320 = OFF, COM1: M1139 = OFF):

When X0 = ON, MODRW instruction executes the function specified by Function Code 05

PLC1 ⇨ PLC2, PLC sends: “**01 05 0500 FF00 6F**”

PLC2 ⇨ PLC1, PLC receives: “**01 05 0500 FF00 6F**”

(No data processing on received data)

RTU mode (COM3: M1320 = ON, COM1: M1139 = ON):

When X0 = ON, MODRW instruction executes the function specified by Function Code 05

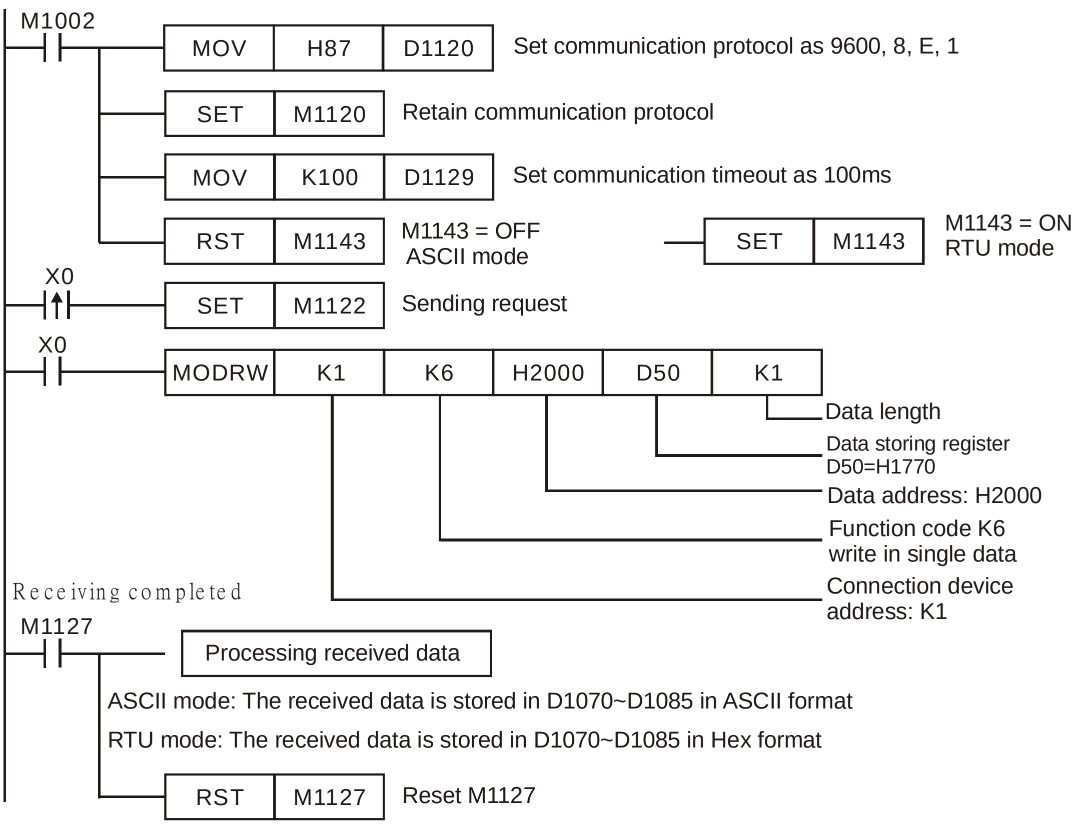
PLC1⇨ PLC2, PLC1 sends: "**01 05 0500 FF00 8C F6**"

PLC2 ⇨PLC1, PLC1 receives: "**01 05 0500 FF00 8C F6**"

(No data processing on received data)

Program Example 7: COM2(RS-485), Function Code H06

1. Function code K6 (H06): Write in single word device.
2. Set the value to be written into VFD-B in the register specified by operand **S**.
3. For ASCII or RTU mode, PLC COM2 stores the data to be sent in D1256~D1295, and received data in D1070~D1085.
4. Take the connection between PLC (PLC COM2) and VFD-B for example, the tables below explains the status when PLC reads status of VFD-B. (M1143 = OFF, ASCII Mode) (M1143 = ON, RTU Mode)



ASCII mode (M1143 = OFF)

When X0 = ON, MODRW instruction executes the function specified by Function Code 06

PLC ⇨ VFD-B, PLC sends: **"01 06 2000 1770 52"**

VFD-B ⇨ PLC, PLC receives: **"01 06 2000 1770 52"**

Registers for data to be sent (sending messages)

Register	Data		Descriptions	
D1256 Low byte	‘0’	30 H	ADR 1	Device address of VFD-B: ADR (1,0)
D1256 High byte	‘1’	31 H	ADR 0	
D1257 Low byte	‘0’	30 H	CMD 1	Control parameter: CMD (1,0)
D1257 High byte	‘6’	36 H	CMD 0	
D1258 Low byte	‘2’	32 H	Data Address	
D1258 High byte	‘0’	30 H		
D1259 Low byte	‘0’	30 H		
D1259 High byte	‘0’	30 H		
D1260 Low byte	‘1’	31 H	Data content	H1770 = K6000. The content of register D50
D1260 High byte	‘7’	37 H		
D1261 Low byte	‘7’	37 H		
D1261 High byte	‘0’	30 H		
D1262 Low byte	‘5’	35 H	LRC CHK 1	Checksum: LRC CHK (0,1)
D1262 High byte	‘2’	32 H	LRC CHK 0	

Registers for received data (responding messages)

Register	Data		Descriptions
D1070 Low byte	‘0’	30 H	ADR 1
D1070 High byte	‘1’	31 H	ADR 0
D1071 Low byte	‘0’	30 H	CMD 1
D1071 High byte	‘6’	36 H	CMD 0
D1072 Low byte	‘2’	32 H	Data Address
D1072 High byte	‘0’	30 H	
D1073 Low byte	‘0’	30 H	
D1073 High byte	‘0’	30 H	
D1074 Low byte	‘1’	31 H	Data content
D1074 High byte	‘7’	37 H	
D1075 Low byte	‘7’	37 H	
D1075 High byte	‘0’	30 H	
D1076 Low byte	‘6’	36 H	LRC CHK 1
D1076 High byte	‘5’	35 H	LRC CHK 0

RTU mode (M1143 = ON)

When X0 = ON, MODRW instruction executes the function specified by Function Code 06

PLC ⇨ VFD-B, PLC sends: **“01 06 2000 1770 8C 1E”**

VFD-B → PLC, PLC receives: “**01 06 2000 1770 8C 1E**”

Registers for data to be sent (sending messages)

Register	Data	Descriptions	
D1256 Low byte	01 H	Address	
D1257 Low byte	06 H	Function	
D1258 Low byte	20 H	Data Address	
D1259 Low byte	00 H		
D1260 Low byte	17 H	Data content	H1770 = K6000. The content of register D50
D1261 Low byte	70 H		
D1262 Low byte	8C H	CRC CHK Low	
D1263 Low byte	1E H	CRC CHK High	

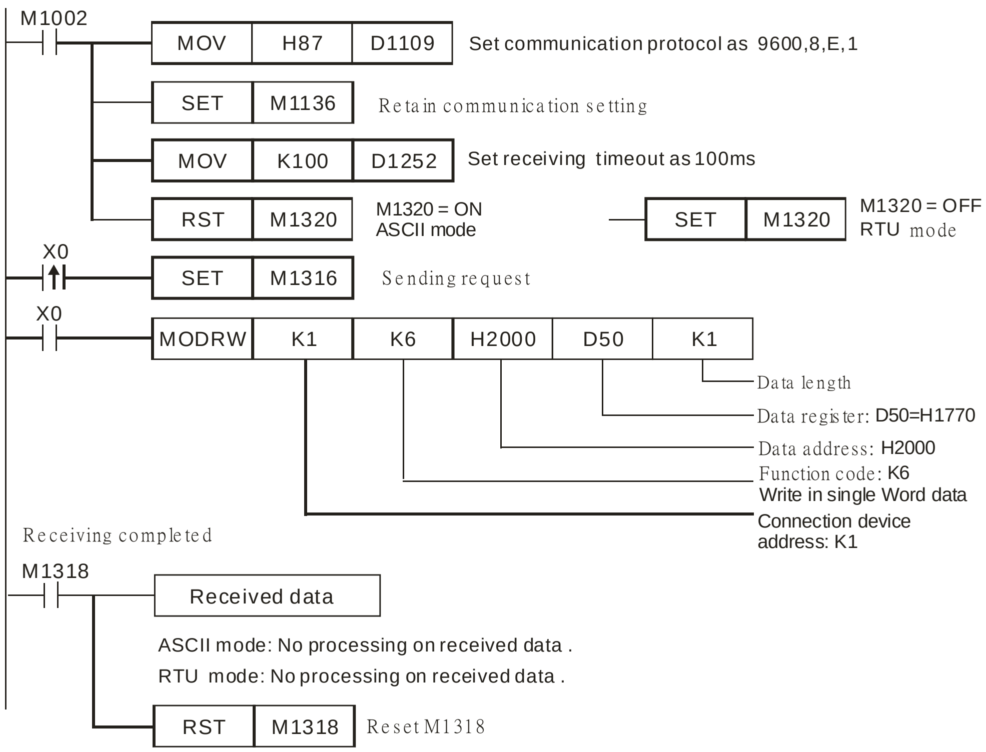
Registers for received data (responding messages)

Register	Data	Descriptions
D1070 Low byte	01 H	Address
D1071 Low byte	06 H	Function
D1072 Low byte	20 H	Data Address
D1073 Low byte	00 H	
D1074 Low byte	17 H	Data content
D1075 Low byte	70 H	
D1076 Low byte	8C H	CRC CHK Low
D1077 Low byte	1E H	CRC CHK High

Program example 8: COM1 (RS-232) / COM3 (RS-485), Function Code H06

1. Function code K6 (H06): Write in single Word device.
2. Set the value to be written into VFD-B in the register specified by operand **S**.
3. PLC COM1/COM3 will not process the received data.
4. Take the connection between PLC (PLC COM3) and VFD-B for example, the tables below explains the status when PLC COM3 writes in single Word device in VFD-B (M1320 = OFF, ASCII Mode), (M1320 = ON, RTU Mode)

- If PLC applies COM1 for communication, the below program can be usable by changing:
  a) D1109→D1036: communication protocol
  b) M1136→M1138: retain communication setting
  c) D1252→D1249: Set value for data receiving timeout
  d) M1320→M1139: ASCII/RTU mode selection
  e) M1316→M1312: sending request
  f) M1318→M1314: receiving completed flag



ASCII mode (COM3: M1320 = OFF, COM1: M1139 = OFF):

When X0 = ON, MODRW instruction executes the function specified by Function Code 06

PLC ⇨ VFD-B, PLC sends: **"01 06 2000 1770 52"**

VFD-B ⇨ PLC, PLC receives: **"01 06 2000 1770 52"**

(No data processing on received data)

RTU mode (COM3: M1320 = ON, COM1: M1139 = ON)

When X0 = ON, MODRW instruction executes the function specified by Function Code 06

PLC ⇨ VFD-B, PLC sends: **"01 06 2000 1770 8C 1E"**

VFD-B → PLC, PLC receives: "**01 06 2000 1770 8C 1E**"

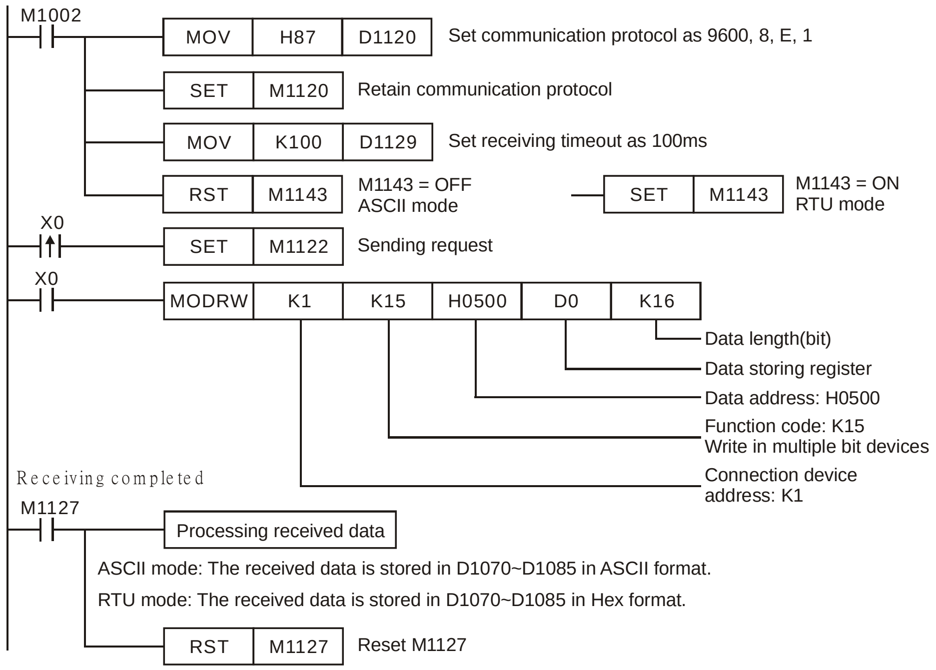
(No data processing on received data)

Program Example 9: COM2 (RS-485), Function Code H0F

1. Function code K15 (H0F): write in multiple bit devices. Up to 64bits can be written.
2. PLC1 connects to PLC2: (M1143 = OFF, ASCII Mode), (M1143 = ON, RTU Mode)
3. For ASCII or RTU mode, PLC COM2 stores the data to be sent in D1256~D1295 and the received data in D1070~D1085.
4. Take the connection between PLC1 (PLC COM2) and PLC2 (PLC COM1) for example, the tables below explain the status when PLC1 force ON/OFF Y0~Y17 of PLC2.

Set value: K4Y0=1234H

Device	Status	Device	Status	Device	Status	Device	Status
Y0	OFF	Y1	OFF	Y2	ON	Y3	OFF
Y4	ON	Y5	ON	Y6	OFF	Y7	OFF
Y10	OFF	Y11	ON	Y12	OFF	Y13	OFF
Y14	ON	Y15	OFF	Y16	OFF	Y17	OFF



ASCII mode (M1143 = OFF)

When X0 = ON, MODRW instruction executes the function specified by Function Code H0F.

PLC1 ⇨ PLC2, PLC sends: " **01 0F 0500 0010 02 3412 93** "

PLC2 ⇨ PLC1, PLC receives: " **01 0F 0500 0010 DB** "

Registers for data to be sent (sending messages)

Register	Data		Descriptions	
D1256 Low byte	'0'	30 H	ADR 1	Device address: ADR (1,0)
D1256 High byte	'1'	31 H	ADR 0	
D1257 Low byte	'0'	30 H	CMD 1	Control parameter: CMD (1,0)
D1257 High byte	'F'	46 H	CMD 0	
D1258 Low byte	'0'	30 H	Data Address	
D1258 High byte	'5'	35 H		
D1259 Low byte	'0'	30 H		
D1259 High byte	'0'	30 H		
D1260 Low byte	'0'	30 H	Number of Data (count by bit)	
D1260 High byte	'0'	30 H		
D1261 Low byte	'1'	31H		
D1261 High byte	'0'	30 H		
D1262 Low byte	'0'	30 H	Byte Count	
D1262 High byte	'2'	32 H		
D1263 Low byte	'3'	33 H	Data contents	1234H Content of register D0
D1263 High byte	'4'	46 H		
D1264 Low byte	'1'	33 H		
D1264 High byte	'2'	46 H		
D1265 Low byte	'9'	39 H	LRC CHK 1	Checksum: LRC CHK (0,1)
D1265 High byte	'3'	33 H	LRC CHK 0	

Registers for received data (responding messages)

Register	Data		Descriptions
D1070 Low byte	'0'	30 H	ADR 1
D1070 High byte	'1'	31 H	ADR 0
D1071 Low byte	'0'	31 H	CMD 1
D1071 High byte	'F'	46 H	CMD 0
D1072 Low byte	'0'	30 H	Data Address
D1072 High byte	'5'	35 H	
D1073 Low byte	'0'	30 H	
D1073 High byte	'0'	30 H	
D1074 Low byte	'0'	30 H	Number of Data(count by bit)
D1074 High byte	'0'	30 H	
D1075 Low byte	'1'	31 H	
D1075 High byte	'0'	30 H	
D1076 Low byte	'D'	44 H	LRC CHK 1
D1076 High byte	'B'	42 H	LRC CHK 0

RTU mode (M1143 = ON)

When X0 = ON, MODRW instruction executes the function specified by Function Code H0F

PLC1 ⇨ PLC2，PLC1 sends: "**01 0F 0500 0010 02 34 12 21 ED**"

PLC2 ⇨ PLC1，PLC1 receives: "**01 0F 0500 0010 54 CB**"

Registers for data to be sent (sending messages)

<table>
<tr><th>Register</th><th>Data</th><th colspan="2">Descriptions</th></tr>
<tr><td>D1256 Low byte</td><td>01 H</td><td colspan="2">Address</td></tr>
<tr><td>D1257 Low byte</td><td>0F H</td><td colspan="2">Function</td></tr>
<tr><td>D1258 Low byte</td><td>05 H</td><td colspan="2" rowspan="2">Data Address</td></tr>
<tr><td>D1259 Low byte</td><td>00 H</td></tr>
<tr><td>D1260 Low byte</td><td>00 H</td><td colspan="2" rowspan="2">Number of Data(count by bit)</td></tr>
<tr><td>D1261 Low byte</td><td>10 H</td></tr>
<tr><td>D1262 Low byte</td><td>02 H</td><td colspan="2">Byte Count</td></tr>
<tr><td>D1263 Low byte</td><td>34 H</td><td>Data content 1</td><td>Content of D0: H34</td></tr>
<tr><td>D1264 Low byte</td><td>12 H</td><td>Data content 2</td><td>Content of D1: H12</td></tr>
<tr><td>D1265 Low byte</td><td>21 H</td><td colspan="2">CRC CHK Low</td></tr>
<tr><td>D1266 Low byte</td><td>ED H</td><td colspan="2">CRC CHK High</td></tr>
</table>

Registers for received data (responding messages)

<table>
<tr><th>Register</th><th>Data</th><th>Descriptions</th></tr>
<tr><td>D1070 Low byte</td><td>01 H</td><td>Address</td></tr>
<tr><td>D1071 Low byte</td><td>0F H</td><td>Function</td></tr>
<tr><td>D1072 Low byte</td><td>05 H</td><td rowspan="2">Data Address</td></tr>
<tr><td>D1073 Low byte</td><td>00 H</td></tr>
<tr><td>D1074 Low byte</td><td>00 H</td><td rowspan="2">Number of Data(count by bit)</td></tr>
<tr><td>D1075 Low byte</td><td>10H</td></tr>
<tr><td>D1076 Low byte</td><td>54H</td><td>CRC CHK Low</td></tr>
<tr><td>D1077 Low byte</td><td>CB H</td><td>CRC CHK High</td></tr>
</table>

Program example 10: COM1 (RS-232) / COM3 (RS-485), Function Code H0F

1. Function code K15 (H0F): write in multiple bit devices. Up to 64 bits can be written
2. PLC1 connects to PLC2: (M1143 = OFF, ASCII mode), (M1143 = ON, RTU mode)
3. PLC COM1/COM3 will not process the received data.
4. Take the connection between PLC1 (PLC COM3) and PLC2 (PLC COM1) for example, the tables below explain the status when PLC1 force ON/OFF Y0~Y17 of PLC2.

Set value: K4Y0=1234H

Device	Status	Device	Status	Device	Status	Device	Status
Y0	OFF	Y1	OFF	Y2	ON	Y3	OFF
Y4	ON	Y5	ON	Y6	OFF	Y7	OFF
Y10	OFF	Y11	ON	Y12	OFF	Y13	OFF
Y14	ON	Y15	OFF	Y16	OFF	Y17	OFF

- If PLC applies COM1 for communication, the below program can be usable by changing:

a) D1109→D1036: communication protocol

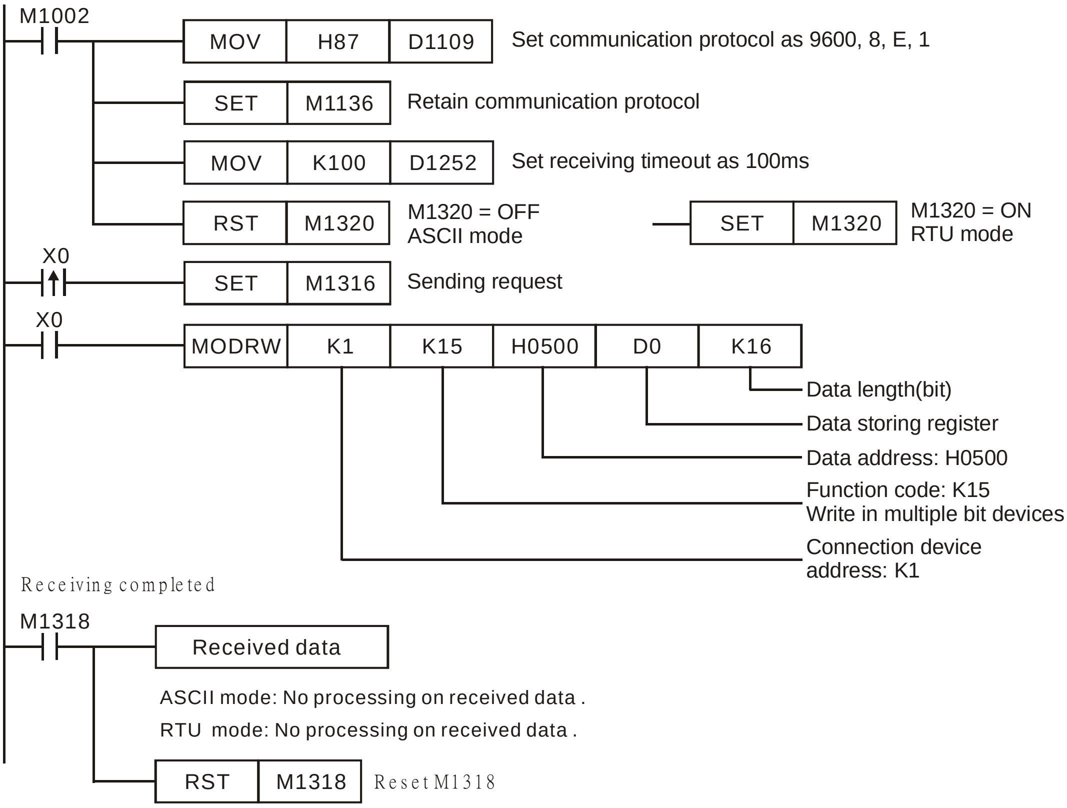
b) M1136→M1138: retain communication setting

c) D1252→D1249: Set value for data receiving timeout

d) M1320→M1139: ASCII/RTU mode selection

e) M1316→M1312: sending request

f) M1318→M1314: receiving completed flag



ASCII mode (COM3: M1320 = OFF, COM1: M1139 = OFF):

When X0 = ON, MODRW executes the function specified by Function Code H0F

PLC1 ⇨ PLC2, PLC sends: " **01 0F 0500 0010 02 3412 93** "

PLC2 ⇨ PLC1, PLC receives: " **01 0F 0500 0010 DB** "

(No data processing on received data)

RTU mode (COM3: M1320 = ON, COM1: M1139 = ON):

When X0 = ON, MODRW executes the function specified by Function Code H0F

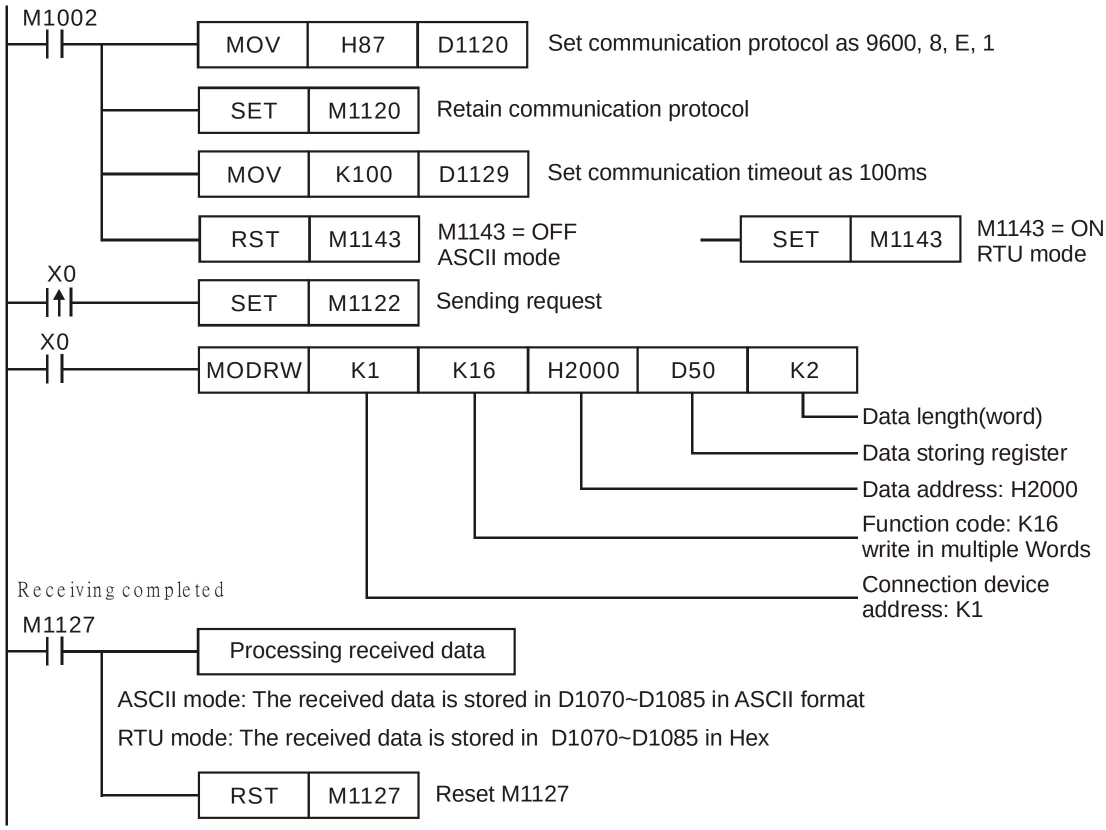
PLC1 ⇨ PLC2, PLC1 sends: "**01 0F 0500 0010 02 34 12 21 ED**"

PLC2 ⇨ PLC1, PLC1 receives: "**01 0F 0500 0010 54 CB**" ,

(No data processing on received data)

Program Example 11: COM2 (RS-485), Function Code H10

1. Function code K16 (H10): Write in multiple Word devices. Up to 16 Words can be written. For PLC COM2 ASCII mode, only 8 words can be written.
2. For ASCII or RTU mode, PLC COM2 stores the data to be sent in D1256~D1295, and the received data in D1070~D1085.
3. Take the connection between PLC COM2 and VFD-B AC motor drive for example, the tables below explain the status when PLC COM2 writes multiple word devices in VFD-B.



ASCII mode (M1143 = OFF)

When X0 = ON, MODRW instruction executes the function specified by Function Code H10

PLC ⇨VFD-B, PLC transmits: **“01 10 2000 0002 04 1770 0012 30”**

VFD⇨PLC, PLC receives: **“01 10 2000 0002 CD”**

Registers for data to be sent (sending messages)

Register	Data		Descriptions	
D1256 Low byte	‘0’	30 H	ADR 1	Address of VFD: ADR (1,0)
D1256 High byte	‘1’	31 H	ADR 0	
D1257 Low byte	‘1’	31 H	CMD 1	Control parameter: CMD (1,0)
D1257 High byte	‘0’	30 H	CMD 0	
D1258 Low byte	‘2’	32 H	Data Address	
D1258 High byte	‘0’	30 H		
D1259 Low byte	‘0’	30 H		
D1259 High byte	‘0’	30 H		
D1260 Low byte	‘0’	30 H	Number of Register	
D1260 High byte	‘0’	30 H		
D1261 Low byte	‘0’	30 H		
D1261 High byte	‘2’	32 H		
D1262 Low byte	‘0’	30 H	Byte Count	
D1262 High byte	‘4’	34 H		
D1263 Low byte	‘1’	31 H	Data contents 1	The content of register D50: H1770(K6000)
D1263 High byte	‘7’	37 H		
D1264 Low byte	‘7’	37 H		
D1264 High byte	‘0’	30 H		
D1265 Low byte	‘0’	30 H	Data contents 2	The content of register D51: H0012(K18)
D1265 High byte	‘0’	30 H		
D1266 Low byte	‘1’	31 H		
D1266 High byte	‘2’	32 H		
D1267 Low byte	‘3’	33 H	LRC CHK 1	LRC CHK (0,1) is error check
D1267 High byte	‘0’	30 H	LRC CHK 0	

Registers for received data (responding messages)

Register	Data		Descriptions
D1070 Low byte	‘0’	30 H	ADR 1
D1070 High byte	‘1’	31 H	ADR 0
D1071 Low byte	‘1’	31 H	CMD 1
D1071 High byte	‘0’	30 H	CMD 0
D1072 Low byte	‘2’	32 H	Data Address
D1072 High byte	‘0’	30 H	
D1073 Low byte	‘0’	30 H	
D1073 High byte	‘0’	30 H	
D1074 Low byte	‘0’	30 H	Number of Register
D1074 High byte	‘0’	30 H	
D1075 Low byte	‘0’	30 H	
D1075 High byte	‘2’	32 H	
D1076 Low byte	‘C’	43 H	LRC CHK 1
D1076 High byte	‘D’	44 H	LRC CHK 0

RTU mode (M1143 = ON)

When X0 = ON, MODRW instruction executes the function specified by Function Code H10

PLC ⇨VFD-B,PLC transmits: **“01 10 2000 0002 04 1770 0012 EE 0C”**

VFD-B⇨PLC, PLC receives: **”01 10 2000 0002 4A08”**

Registers for data to be sent (sending messages)

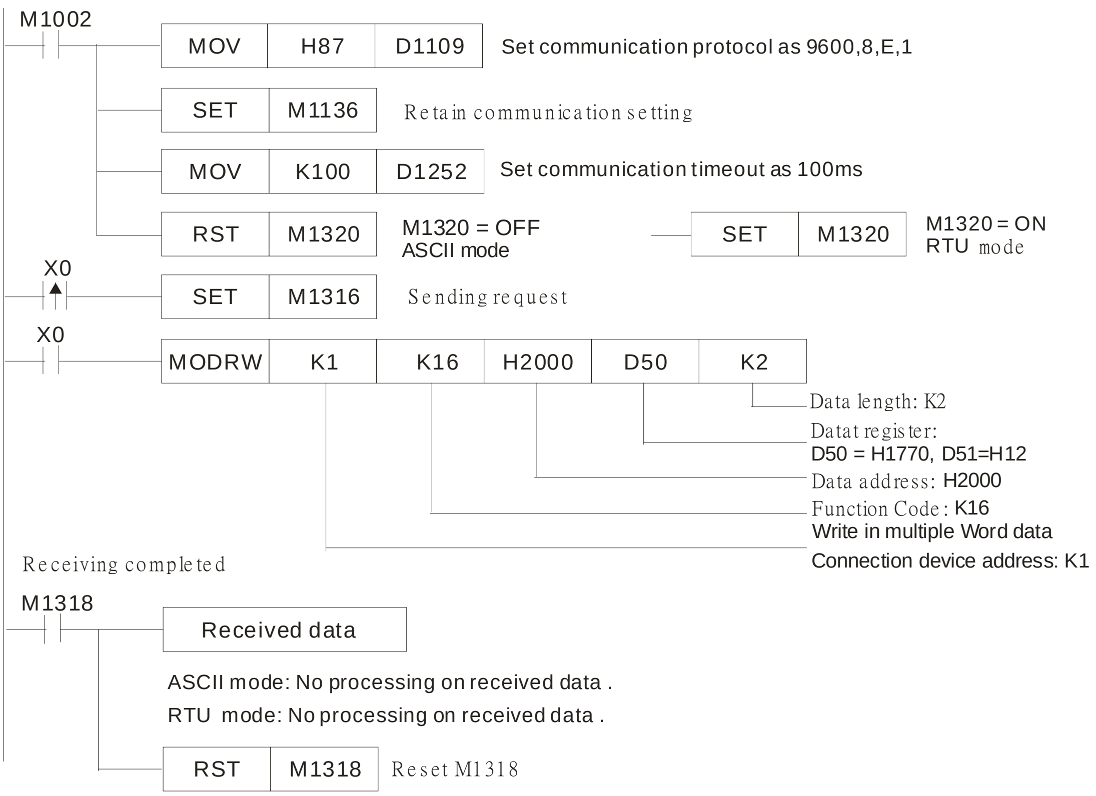
<table>
<tr><th>Register</th><th>Data</th><th colspan="2">Descriptions</th></tr>
<tr><td>D1256 Low byte</td><td>01 H</td><td colspan="2">Address</td></tr>
<tr><td>D1257 Low byte</td><td>10 H</td><td colspan="2">Function</td></tr>
<tr><td>D1258 Low byte</td><td>20 H</td><td colspan="2" rowspan="2">Data Address</td></tr>
<tr><td>D1259 Low byte</td><td>00 H</td></tr>
<tr><td>D1260 Low byte</td><td>00 H</td><td colspan="2" rowspan="2">Number of Register</td></tr>
<tr><td>D1261 Low byte</td><td>02 H</td></tr>
<tr><td>D1262 Low byte</td><td>04 H</td><td colspan="2">Byte Count</td></tr>
<tr><td>D1263 Low byte</td><td>17 H</td><td rowspan="2">Data content 1</td><td rowspan="2">The content of D50: H1770 (K6000)</td></tr>
<tr><td>D1264 Low byte</td><td>70 H</td></tr>
<tr><td>D1265 Low byte</td><td>00 H</td><td rowspan="2">Data content 2</td><td rowspan="2">The content of D51: H0012 (K18)</td></tr>
<tr><td>D1266 Low byte</td><td>12 H</td></tr>
<tr><td>D1262 Low byte</td><td>EE H</td><td colspan="2">CRC CHK Low</td></tr>
<tr><td>D1263 Low byte</td><td>0C H</td><td colspan="2">CRC CHK High</td></tr>
</table>

Registers for received data (responding messages)

<table>
<tr><th>Register</th><th>Data</th><th>Descriptions</th></tr>
<tr><td>D1070 Low byte</td><td>01 H</td><td>Address</td></tr>
<tr><td>D1071 Low byte</td><td>10 H</td><td>Function</td></tr>
<tr><td>D1072 Low byte</td><td>20 H</td><td rowspan="2">Data Address</td></tr>
<tr><td>D1073 Low byte</td><td>00 H</td></tr>
<tr><td>D1074 Low byte</td><td>00 H</td><td rowspan="2">Number of Register</td></tr>
<tr><td>D1075 Low byte</td><td>02 H</td></tr>
<tr><td>D1076 Low byte</td><td>4A H</td><td>CRC CHK Low</td></tr>
<tr><td>D1077 Low byte</td><td>08 H</td><td>CRC CHK High</td></tr>
</table>

Program example 12: COM1 (RS-232) / COM3 (RS-485), Function Code H10

1. Function code K16 (H10): Write in multiple Word devices. Up to 16 Words can be written. For PLC COM2 ASCII mode, only 8 words can be written.
2. PLC COM1/COM3 will not process the received data
3. Take the connection between PLC COM3 and VFD-B for example, the tables below explain the status when PLC COM3 writes multiple Words in VFD-B. (M1320 = OFF, ASCII mode) (M1320 = ON, RTU mode)
   - If PLC applies COM1 for communication, the below program can be usable by changing:
     a) D1109→D1036: communication protocol
     b) M1136→M1138: retain communication setting
     c) D1252→D1249: Set value for data receiving timeout
     d) M1320→M1139: ASCII/RTU mode selection
     e) M1316→M1312: sending request
     f) M1318→M1314: receiving completed flag

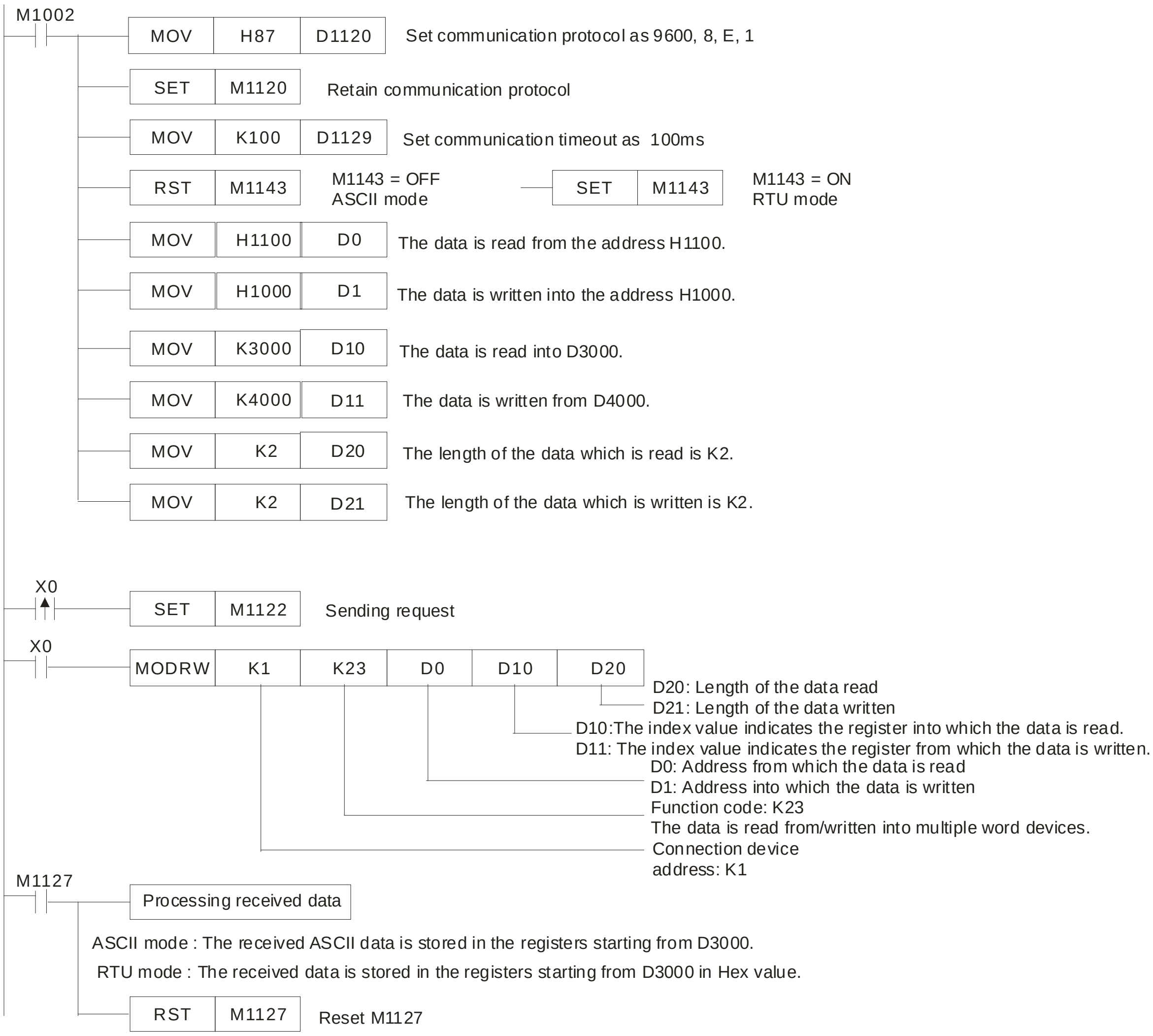


- ASCII mode (COM3: M1320 = OFF, COM1: M1139 = OFF):
  When X0 = ON, MODRW executes the function specified by Function Code H10
  PLC ⇨VFD-B, PLC sends: **“01 10 2000 0002 04 1770 0012 30”**
  VFD⇨PLC, PLC receives: **“01 10 2000 0002 CD”**
  (No processing on received data)
- RTU Mode (COM3: M1320=On, COM1: M1139=On):
  When X0 = ON, MODRW executes the function specified by Function Code H10
  PLC ⇨VFD-B,PLC sends: “0**1 10 2000 0002 04 1770 0012 EE 0C”**
  VFD-B⇨PLC, PLC receives :”01 10 2000 0002 4A08”
  (No processing on received data)

Program example 13: COM2 (RS-485)), Function Code H17

1. Function code K23 (H17): Data is read from multiple word devices and data is written into multiple word devices. Data can be read from 16 word devices at most, and data can be written into 16 word devices at most.
2. In the ASCII or RTU mode, the data received is stored in the registers starting from the register indicated by the index value in **S**.
3. The connection between PLC-A (PLC COM2) and PLC-B:

   Data is read from multiple word devices in PLC-B into PLC-A, and data is written into multiple word devices in PLC-B from PLC-A. (M1143=OFF, ASCII Mode) (M1143=ON, RTU Mode)



- ASCII Mode (M1143=OFF)

  When X0=ON, MODRW executes the function specified by the function ode H17.

  PLC-A ⇨PLC-B, PLC-A sends: **"01 17 1100 0002 1000 0002 04 1770 0012 06"**

  PLC-B⇨PLC-A, PLC-A receives: **"01 17 04 0100 1766 66"**

  Registers in PLC-A for received data (responding messages)

Register	Data		Description
D3000 Low byte	‘0’	30 H	ADR 1
D3000 High byte	‘1’	31 H	ADR 0
D3001 Low byte	‘1’	31 H	CMD 1
D3001 High byte	‘7’	37 H	CMD 0
D3002 Low byte	‘0’	30 H	Number of data (bytes)
D3002 High byte	‘4’	34 H	
D3003 Low byte	‘0’	30 H	Contents of the address 1100H
D3003 High byte	‘1’	31 H	
D3004 Low byte	‘0’	30 H	
D3004 High byte	‘0’	30 H	
D3005 Low byte	‘1’	31 H	Contents of the address 1101H
D3005 High byte	‘7’	37 H	
D3006 Low byte	‘6’	36H	
D3006 High byte	‘6’	36H	
D3007 Low byte	‘6’	36H	LRC CHK 1
D3007 High byte	‘6’	36H	LRC CHK 0

- RTU Mode (M1143=ON)

When X0=ON, MODRW executes the function specified by the function ode H17.

PLC-A ⇨PLC-B,PLC-A sends: **“01 17 1100 0002 1000 0002 04 1770 0012 A702”**

PLC-B⇨PLC-A, PLC-A receives: “**01 17 04 0100 1766 7701**”

Registers in PLC-A for received data (responding messages)

Register	Data	Description
D3000 Low byte	01 H	Address
D3001 Low byte	17 H	Function
D3002 Low byte	04 H	Number of data (bytes)
D3003 Low byte	01 H	Contents of the address 1100H
D3004 Low byte	00 H	
D3005 Low byte	17 H	Contents of the address 1101H
D3006 Low byte	66 H	
D3007 Low byte	77 H	CRC CHK Low
D3008 Low byte	01 H	CRC CHK High

Program example 14: COM1 (RS-232)/ COM3 (RS-485), Function Code H17

1. Function code K23 (H17): Data is read from multiple word devices and data is written into multiple word devices. Data can be read from 16 word devices at most, and data can be written into 16 word devices at most.
2. In the ASCII or RTU mode, the data received through COM1/COM3 on the PLC is stored in the registers starting from the register indicated by the index value in **S**+1. Users can use the instruction DTM to transform and move the data.
3. The connection between PLC-A (PLC COM3) and PLC-B:
   - Data is written into multiple word devices in PLC-B from PLC-A. (M1320=OFF, ASCII Mode) (M1320=ON, RTU Mode)
   - If COM1 on PLC-A is connected, the program can be modified as shown below.

     a) D1109→D1036: Communication protocol

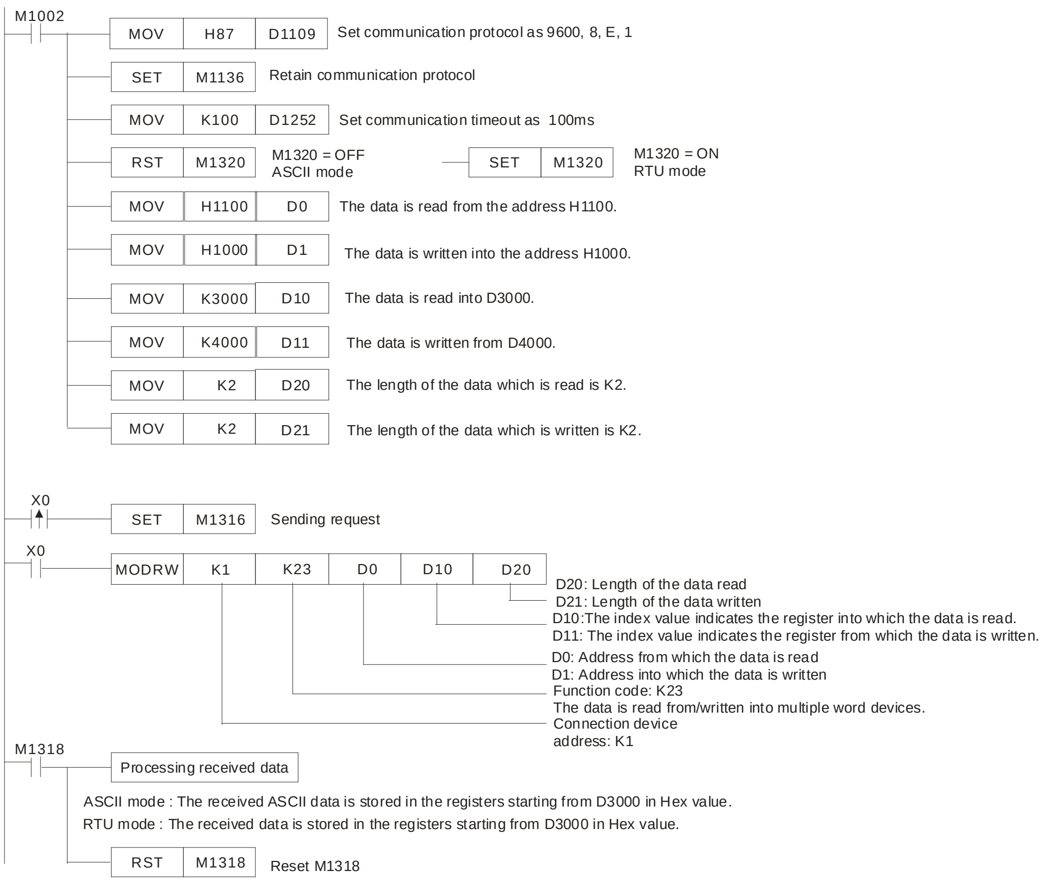
     b) M1136→M1138: The communication setting is retained.

     c) D1252→D1249: Communication timeout

     d) M1320→M1139: Choice between the ASCII mode and the RTU mode

     e) M1316→M1312: The sending of the data though the communication instruction is requested.

     f) M1318→M1314: The receiving of the data through the communication instruction is complete.



- ASCII Mode (COM3: M1320=OFF; COM1: M1139=OFF):

  When X0=ON, MODRW executes the function specified by the function ode H17.

  PLC-A ⇨PLC-B, PLC-A sends: **“01 17 1100 0002 1000 0002 04 1770 0012 06”**

  PLC-B⇨PLC-A, PLC-A receives: **“01 17 04 0100 1766 66”**

  Registers in PLC-A for received data (responding messages)

Register	Data	Description
D3000	0100H	PLC-A converts ASCII codes in 1100H and stores the converted data automatically.
D3001	1766H	PLC-A converts ASCII codes in 1101H and stores the converted data automatically.

- RTU Mode (COM3: M1320=ON; COM1: M1139=ON):

  When X0=ON, MODRW executes the function specified by the function ode H17.

  PLC-A ⇨PLC-B,PLC-A sends: **“01 17 2100 0002 2000 0002 04 1770 0012 A702”**

  PLC-B⇨PLC-A, PLC-A receives: “**01 17 04 0100 1766 7701**”

  Registers in PLC-A for received data (responding messages)

Register	Data	Description
D3000	0100 H	PLC-A converts data in 1100H and stores the converted data automatically.
D3001	1766 H	PLC-A converts data in 1101H and stores the converted data automatically.

3

API	Mnemonic	Operands	Function	Controllers			
206	ASDRW	S1 S2 S	ASDA servo drive R/W	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			ASDRW: 7 steps
S_2					*	*							*			
S													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Address of servo drive (K0~K254) **S_2**: Function code **S**: Register for read/written data

Explanations:

1. ASDRW communication instruction supports COM2 (RS-485) and COM3 (RS-485)
2. **S_1**: station number of servo drive. Range: K0~K254. K0 indicates broadcasting, i.e. PLC will not receive feedback data.
3. **S_2**: function code. Please refer to the table below.
4. **S**: Register for read/written data. Please refer to the table below for explanations.
5. Explanations of function code:

Exclusively for ASDA of A-type, AB type, A+ type, B type				
Code	Function	Parameter	Com. Addr.	Read/Write data (Settings)
K0(H0)	Status monitor	P0-04 ~ P0-08	0004H ~ 0008H	**S**+0 ~ **S**+4: Please refer to explanations in ASDA manuals.
K1(H1)	Block Data Read Register	P0-09 ~ P0-16	0009H ~ 0010H	**S**+0 ~ **S**+7: Please refer to explanations in ASDA manuals. B Type is not supported.
K2(H2)	Block Data Write Register	P0-09 ~ P0-16	0009H ~ 0010H	**S**+0 ~ **S**+7: Please refer to explanations in ASDA manuals. B Type is not supported.
K3(H3)	JOG Operation	P4-05	0405H	**S**: Range: 1~3000, 4999, 4998, 5000
K4(H4)	Servo ON/OFF	P2-30	021EH	**S**: K1 = ON, Others = OFF
K5(H5)	Speed Command (3 sets)	P1-09 ~ P1-11	0109H ~ 010BH	**S**+0 ~ **S**+2: Range: -5000~+5000
K6(H6)	Torque Command (3 sets)	P1-12 ~ P1-14	010CH ~ 010EH	**S**+0 ~ **S**+2: Range: -300~+300

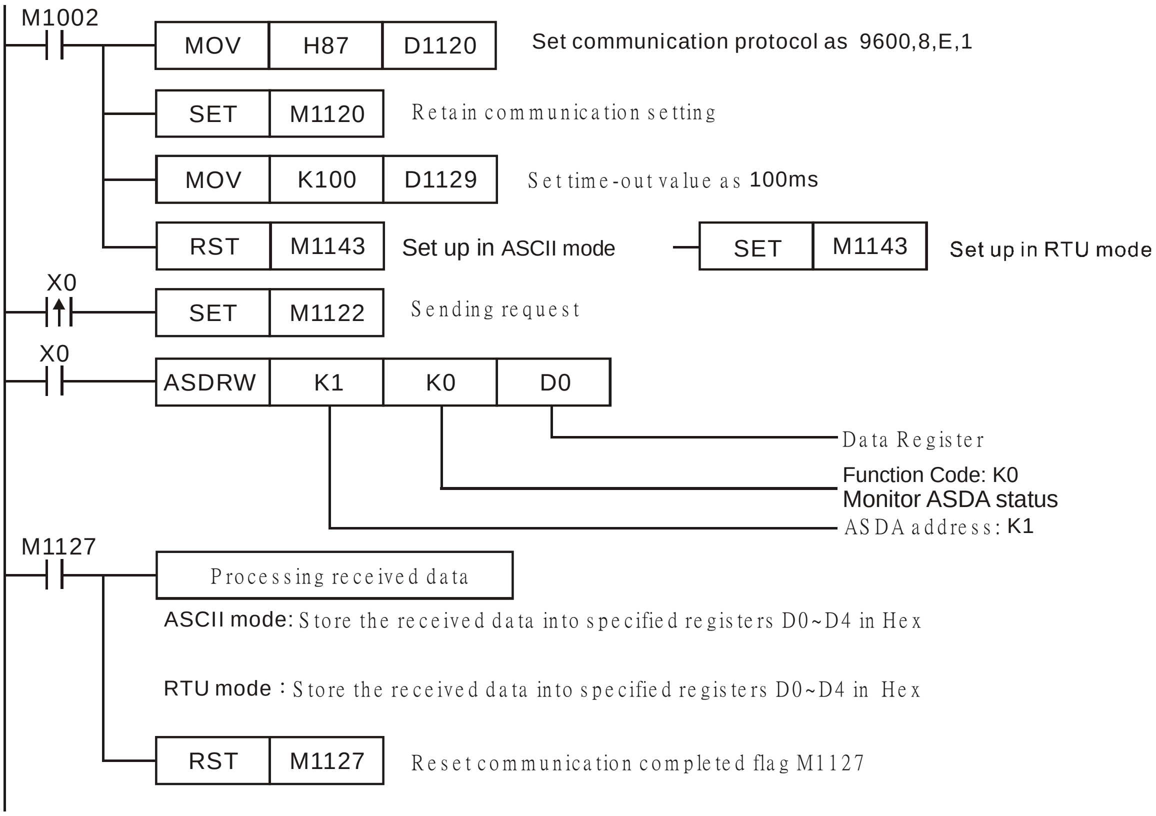
For A2-type only				
Code	Function	Parameter	Com. Addr.	Read/Write data (Settings)
K16(H10)	Status monitor (Read)	P0-09 ~ P0-13	0012H ~ 001BH	**S**+0 ~ **S**+9: Please refer to explanations in ASDA-A2 manual.
K17(H11)	Status monitor selection (Write)	P0-17 ~ P0-20	0022H ~ 0029H	**S**+0 ~ **S**+7: Please refer to explanations in ASDA-A2 manual.
K18(H12)	Mapping parameter (Write)	P0-25 ~ P0-28	0032H ~ 0039H	**S**+0 ~ **S**+7: Please refer to explanations in ASDA-A2

For A2-type only				
Code	Function	Parameter	Com. Addr.	Read/Write data (Settings)
				manual.
K19(H13)	JOG Operation	P4-05	040AH	**S**: Range: 1~5000, 4999, 4998, 0
K20(H14)	Auxiliary Function (Servo ON/OFF)	P2-30	023CH	**S**: K1 = ON, Others = OFF
K21(H15)	Speed Command (3 sets)	P1-09 ~ P1-11	0112H ~ 0117H	**S**+0 ~ **S**+5: Range: -60000~+60000
K22(H16)	Torque Command (3 sets)	P1-12 ~ P1-14	0118H ~ 011DH	**S**+0 ~ **S**+5: Range: -300~+300
K23(H17)	Block Data Read / Write Register (for mapping parameter)	P0-35 ~ P0-38	0046H~ 004DH	**S**+0 ~ **S**+7: Please refer to explanations in ASDA-A2 manual.

6. For relative M flags and special D registers, please refer to explanations of API 80 RS instruction.

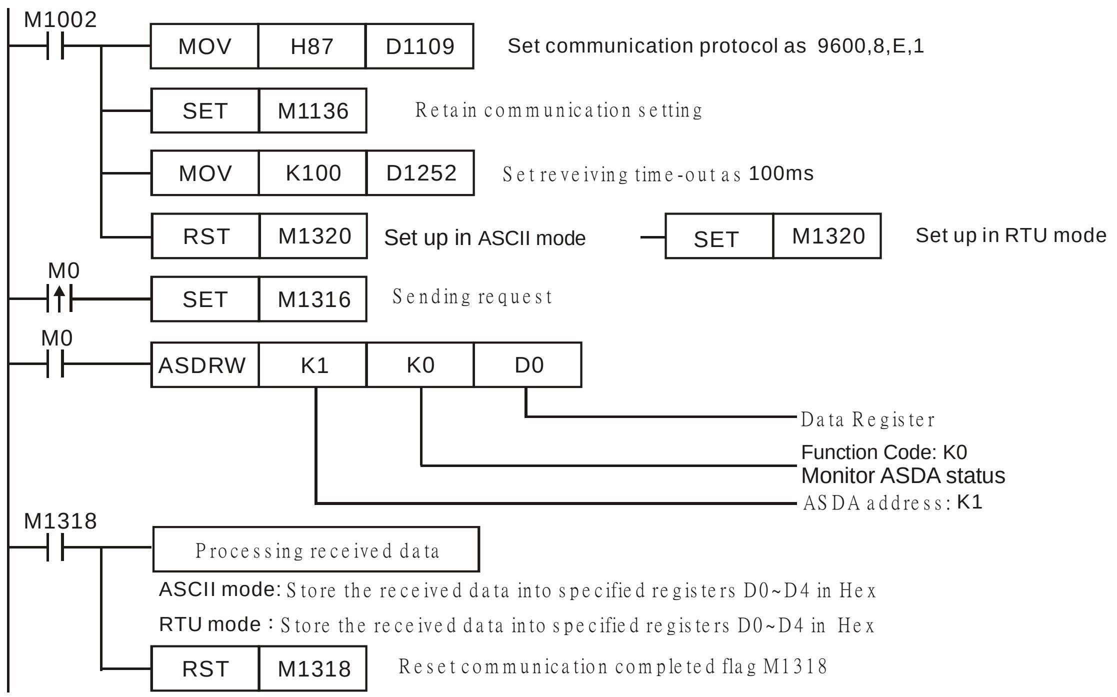
Program example 1: COM2 (RS-485)

1. When X0 = ON, PLC will send out communication commands by COM2 to read status of servo drive.
2. When PLC received the feedback data from ASDA, M1127 will be active and the read data will be stored in D0~D4.



Program example 2: COM3(RS-485)

1. When M0 = ON, PLC sends communication commands by COM3 to read servo drive status.
2. When PLC received the feedback data from ASDA, M1318 will be active and the read data will be stored in D0~D4.



Points to note:

Relative flags and special D registers of COM2/COM3 :

	COM2	COM3	Function Description
Protocol setting	M1120	M1136	Retain communication setting
	M1143	M1320	ASCII/RTU mode selection
	D1120	D1109	Communication protocol
	D1121	D1255	PLC communication address
Sending request	M1122	M1316	Sending request
	D1129	D1252	Communication timeout setting (ms)
Receiving completed	M1127	M1318	Data receiving completed
Errors	-	M1319	Data receiving error
	-	D1253	Communication error code
	M1129	-	Communication timeout setting (ms)
	M1140	-	COM2 (RS-485) MODRD/MODWR/MODRW data receiving error
	M1141	-	MODRD/MODWR/MODRW parameter error (Exception Code exists in received data) Exception Code is stored in D1130
	D1130	-	COM2 (RS-485) Error code (exception code) returning from Modbus communication

API	Mnemonic	Operands	Function	Controllers
295	DMVRW	S_1 S_2 D_1 D_2	DMV communication command	SS2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1													*			DMVRW: 9 steps
S_2													*			
D_1													*			
D_2		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/EC5	SS2	SA2/SE	SX2	ES2/EX2/EC5	SS2	SA2/SE	SX2	ES2/EX2/EC5	SS2	SA2/SE	SX2

Operands:

S_1: Communication port on a PLC **S_2**: Function of a DMV **D_1**: Source or destination device

D_2: Communication flag device

Explanations:

The models supported are SS2 V3.2 and above.

S_1 specifies a communication port on a PLC for sending/receiving data and station numbers. Only the communication ports on a PLC are supported. Please refer to the description of the PLC used for more information.

S_1+0 ~ **S_1**+3 are described below.

Number	Description	Remark
S_1+0	COM on a PLC	Please refer to the description of a PLC.
S_1+1	Station address of a DMV	Applicable to a serial communication port (RS485/RS232/RS422) K1~K254
S_1+2, **S_1**+3	Reserved	Reserved

Description of **S_1**+0:

Communication port	**S_1**+0	Numbers must be used
COM on a PLC	K1~K5 K1~K5 represent PLC COM1~PLC COM5.	**S_1**+0 ~ **S_1**+1

S_2 is used to set a communication function code. The devices that these operand occupies and the functions of the devices are described below.

Number	Description	Remark
S_2+0	Communication combination function code	Please refer to the description of the function codes below.
S_2+1	Communication address	It is only applicable to K0, and is not applicable to other codes.
S_2+2	Reading/Writing	0: Reading Other values: Writing It is only applicable to K0, and is not applicable

Number	Description	Remark
		to other codes.
S_2+3	Communication data length	It is used to set the length of the data read/written. A word is a unit of measurement for length. The maximum number of words which can be read/written is 16.

S_2+0: Communication combination function code

Function code	Attribute[#1]	Function
K0	R or W	There is no communication combination. Users can define a DMV communication command. Please refer to DMO Module Manual for more information about the registers which can be read/written. The data read/written are stored in the devices starting from **D_1**.
K1	W and R	Communication combination commands sent to a DMV[#2]: 1) DMV trigger 1 is enabled. 2) The value in **S_2**+3 indicates the number of data read from the output data area in a DMV**.** (The maximum number of words which can be read is 16.) The data read is stored in the devices starting from **D_1**.
K2	W	Communication combination commands sent to a DMV: 1) The DMV program number indicated by the value in **D_1** is used. (The value in **D_1** is in the range of 0 to 31.) 2) DMV trigger 1 is enabled.
K3	W and R	Communication combination commands sent to a DMV: 1) The DMV program number indicated by the value in **D_1** is used. (The value in **D_1** is in the range of 0 to 31.) 2) DMV trigger 1 is enabled. 3) The value in **S_2**+3 indicates the number of data read from the output data area in a DMV**.** (The maximum number of words which can be read is 16.) The data read is stored in the devices starting from **D_1**.
K4	W	Communication combination commands sent to a DMV: 1) The values in **D_1**+0 and **D_1**+1 are written into internal memory 1. 2) DMV trigger 1 is enabled.
K5	W and R	Communication combination commands sent to a DMV: 1) The values in **D_1**+0 and **D_1**+1 are written into internal memory 1. 2) DMV trigger 1 is enabled. 3) The value in **S_2**+3 indicates the number of data read from the output data area in a DMV**.** (The maximum number of words which can be read is 16.) The data read is stored in the devices starting from **D_1**.
K6	W	Communication combination commands sent to a DMV: 1) The values in **D_1**+0 and **D_1**+1 are written into internal memory 2. 2) DMV trigger 1 is enabled.
K7	W and R	Communication combination commands sent to a DMV: 1) The values in **D_1**+0 and **D_1**+1 are written into internal memory 2. 2) DMV trigger 1 is enabled. 3) The value in **S_2**+3 indicates the number of data read from the output data area in a DMV**.** (The maximum number of words which can be read is 16.) The data read is stored in the devices starting from **D_1**.

Note[#1]: W and R mean that a writing communication command is executed first, and then a reading communication command is executed. If the function code used is K3, the **D** operand functions as a source device at first, and then functions as a destination device when a reading command is executed.

Note#2: If a communication combination command is used, **S_2**+1 and **S_2**+2 will be set by the PLC according to the communication combination command.

D_1 is a source device or a destination device. Please refer to the description of the function codes above.

D_2 is a communication state flag. It occupies three consecutive devices. It is described below.

Number	On	Remark
D_2+0	The DMV is busy.	If the DMV is busy, a communication command will be resent automatically until the DMV replies that the communication is complete.
D_2+1	The communication with the DMV is complete.	
D_2+2	Communication error or timeout	The DMV does not reply after a timeout period.

Whenever the instruction is enabled, the PLC automatically reset **D_2** to Off.

Example 1: Users define a DMV communication command. COM2 on a PLC communicates with a DMV. H0888 is written into the communication address H10D0 in the DMV. The control procedure is described below.

1-1: Write K2 into D0. (COM2 on the PLC is used.) Write K1 into D1. (The station address of the DMV is K1.)

1-2: Write K0 into D4. The users define a DMV communication command by themselves, and write the command message into D5~D7.

Operand	Device	Value	Description
S_2+0	D4	K0	Communication combination function code
S_2+1	D5	H10D0	Communication address
S_2+2	D6	K1	Reading/Writing
S_2+3	D7	K1	Communication data length

1-3: When M0 is On, the PLC communicates with the DMV according to the communication data and the communication port set by the users, and H0888 in D8 is written into H10D0 in the DMV.

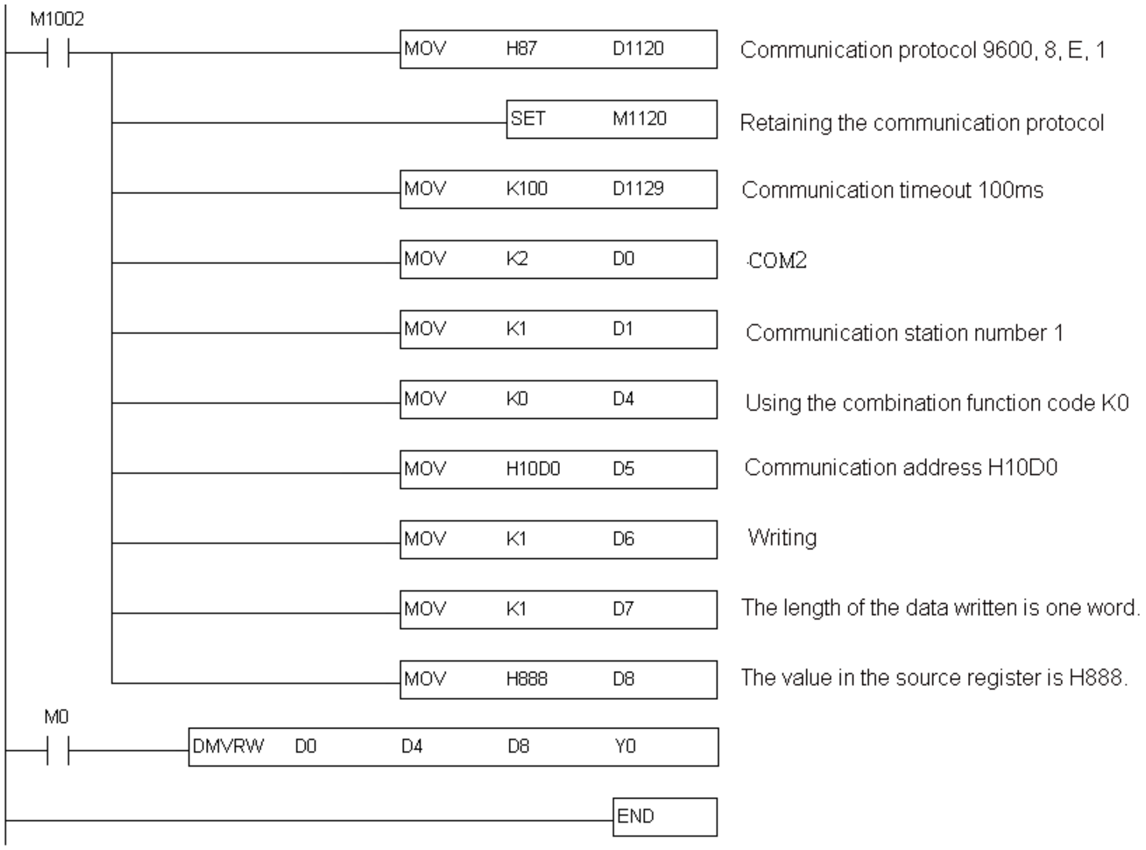
1-4: When the PLC sends the data, the operand **D_2** (Y0) is On (the DMV is busy).

1-5: When the DMV replies successfully, **D_2**+1 (Y1) in the PLC is On (the communication with the DMV is complete).

1-6: If the DMV does not reply after the timeout period 100ms, the PLC will set **D_2**+2 (Y2) to On (a communication timeout occurs).

1-7: If the DMV replies with an execption code, the PLC will resend the command to the DMV automatically, and go back to step 1-3 ~ step 1-5.

The program in the PLC and the comments are shown below.



Example 2: The combination function code K3 is used. COM2 on a PLC communicates with a DMV. The control procedure is shown below.

2-1. Write K2 into D0. (COM2 on the PLC is used.) Write K1 into D1. (The station address of the DMV is K1.)

2-2. The operand $\mathbf{S_2}$+0 specifies D4. Write K3 into D4. The function code K3 is used (There are three communication commands.)The message required are written into $\mathbf{S_2}$+3 and $\mathbf{D_1}$.

Communication command	Operand	Device	Value	Description
First	$\mathbf{D_1}$	D8	H0014	The DMV program number used is K20.
Second	-	-	-	It does not need to be set. The PLC enables DMV trigger 1 by itself.
Third	$\mathbf{S_2}$+3	D7	K2	The value in $\mathbf{S_2}$+3 indicates the number of data read from the output data area in a DMV.

2-3. When M0 is ON, the PLC sends communication data to the DMV accoding to the communication combination command order specified by the function code K3.

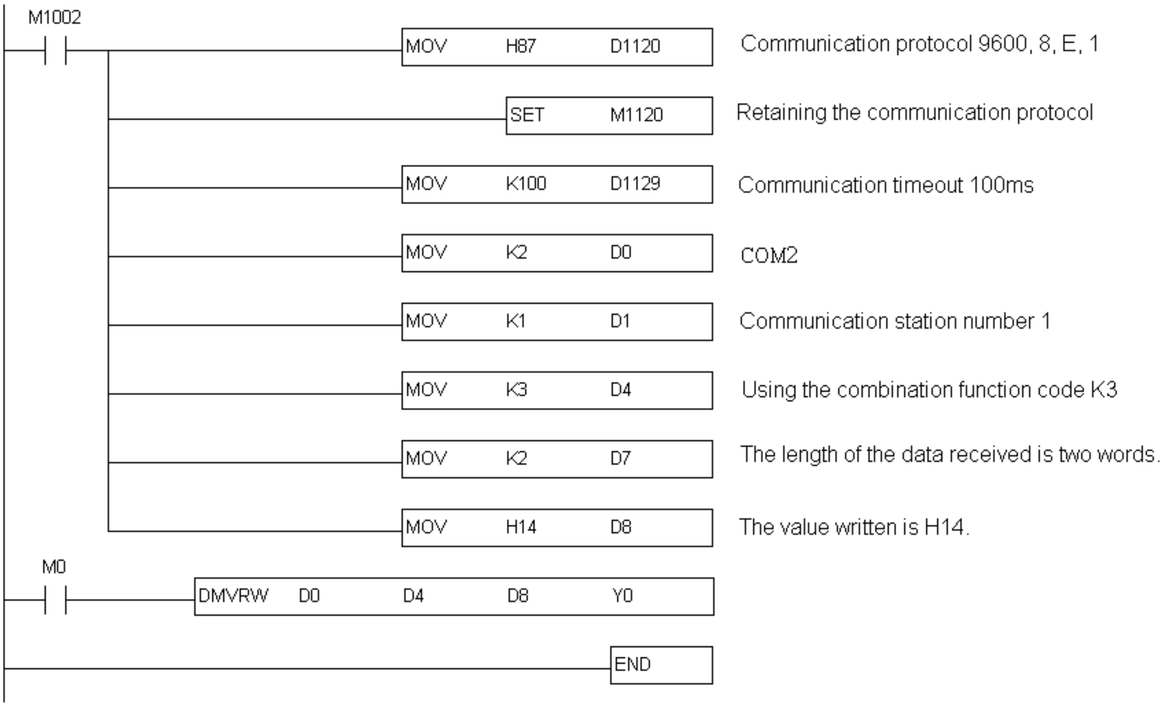
2-4. When the PLC sends the data, the operand $\mathbf{D_2}$ (Y0) is On (the DMV is busy).

2-5. When the DMV replies to the three communication commands successfully, $\mathbf{D_2}$+1 (Y1) in the PLC is On (the communication with the DMV is complete).

2-6. If the DMV does not reply after the timeout period 100ms, the PLC will set **D_2**+2 (Y2) to On (a communication timeout occurs).

2-7. If the DMV replies with an execption code, the PLC will resend the command to the DMV automatically, and go back to step 2-3 ~ step 2-5.

The program in the PLC and the comments are shown below.



Remark: D8 in the example is described below.

3-1. When the first command is sent, the value in D8 indicates a program number. In the example, program number 20 is used, and therefore H14 (or K20) is written into D8 in advance.

3-2. The the third command is sent, D8 becomes a start device in which data received from the DMV is stored. In the example, two-word data is read. When the completion flag is ON, the data read is stored in D8 and D9.

API	Mnemonic	Operands	Function	Controllers			
337	ETHRS	**S₁, S₂, S₃, S₄, D₁, D₂**	Self-defined Ethernet communication port	ES2/ EX2/ EC5	SS2	SE/ ES2-E	SA2/ SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S₁													*			ETHRS: 13 steps
S₂													*			
S₃													*			
S₄					*	*							*			
D₁													*			
D₂			*													

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2	SX2	ES2/ EX2	SS2	SE/ ES2-E	SX2/ SA2	ES2/ EX2/ EC5	SS2	SA2	SX2

Operands:

S₁: Local communication port, target IP address, communication port and UDP/TCP mode
S₂: Communication mode **S₃**: Data source **S₄**: Data length **D₁**: Receive data address
D₂: Receiving completion flag

Explanations:

1. This instruction is currently available for DVP-SE series PLC with firmware V1.83 or later.
2. **S₁** is for setups of local communication port, target IP address, communication port and UDP/TCP mode. This operand occupies 5 consecutive devices.

   IP address settings: this occupies 2 consecutive devices, **S₁**+1 and **S₁**+2 respectively

   IP definition → IP3.IP2.IP1.IP0 → 192.168.0.2

   If **S₁** is D100, the input value should be:

D100 (**S₁**+0)	D101 (**S₁**+1)		D102 (**S₁**+2)		D103 (**S₁**+3)	D104 (**S₁**+4)
Local port	High (IP1)	Low (IP0)	High (IP3)	Low (IP2)	Target port	UDP/TCP
0~65535	0	2	192	168	0~65535	0, 1
	H’0002		H’C0A8			0=UDP, 1=TCP

3. **S₂** is where you can set up modes. Client mode 0 and 1 are exchangeable and the connections are active. Server mode 2, 3 and 4 are exchangeable and the connections are active. But it is required to disconnect the connection when switching between different modes.

Value in S2	S_2 receiving mode	Description of S2+1	Remark
0	After the sending is complete, no receiving is allowed and a completion flag will be set to ON.	Unused	Client Mode 0 cannot be set in the sending data length **S_4** .
1	Complete the sending first, and then start receiving. After receiving is complete, a completion flag will be set to ON.	Receiving timeout setting; unit: 1ms; setting range: 100~32000	Client Mode A. 0 cannot be set in the sending data length **S_4** . (before firmware V1.90 for DVP12SE) B. 0 can be set in the sending data length **S_4** and that indicates not sending but start to receive data. (available for ES2-E with firmware V1.2, or later, 12SE with firmware V1.92 or later, and 26SE with firmware V1.00 or later)
2	Complete the receiving first, after the receiving is done, send the packets. After the sending is complete, a completion flag will be set to ON.	Receiving timeout setting; unit: 1ms; setting range: 100~32000; when the setting value is 0, it means no timeout.	Server Mode 0 cannot be set in the sending data length **S_4** .
3	When the receiving time is less than setting value in **S_2**+1, after receiving the communication packet, the receiving is complete.	Receiving timeout setting; unit: 1ms; setting range: 100~32000; when the setting value is 0, it means no timeout.	Server Mode **S_4** is invalid in this mode.

4	Sending specific communication data (SE: FW V1.95 or later and ES2E: FW V1.46 or later)	Invalid parameter	Server mode The value in S_4 cannot be 0 in this mode. The value exceeding the range will be seen as the maximum or the minimum value.

Mode 2: Contents for the packets to be sent should be ready. Once the receiving is done, the sending is executed immediately.

Mode 3+4: Contents for the packets to be sent can be ready before the next scan cycle. The sending is executed in the next scan cycle.

Target port descriptions: S_2 and S_1+0, S_1+1, S_1+2, S_1+3

Start Mode	Remote IP	Local communication port	Remote communication port	Description
0, 1	Specific IP address	0	0	Illegal
0, 1	Specific IP address	0	Not equal to 0	Master mode, Specifies the IP address; but not specify the local communication port.
0, 1	Specific IP address	Not equal to 0	0	Illegal
0, 1	Specific IP address	Not equal to 0	Not equal to 0	Master mode, Specifies the IP address, local communication port and remote communication port
0, 1	0.0.0.0	No limit to the value	No limit to the value	Illegal
2, 3, 4	Specific IP address	0	No limit to the value	Illegal
2, 3, 4	Specific IP address	Not equal to 0	0	Slave mode, Not specify the IP address and remote communication port
2, 3, 4	Specific IP address	Not equal to 0	Not equal to 0	Slave mode, Specify the IP address and remote communication port
2, 3, 4	0.0.0.0	0	No limit to the value	Illegal
2, 3, 4	0.0.0.0	Not equal to 0	0	Slave mode, Not specify the IP address and remote communication port
2, 3, 4	0.0.0.0	Not equal to 0	Not equal to 0	Slave mode, Not specify the IP address and remote communication port
2, 3, 4	Specific IP address	0	No limit to the value	Illegal

4. The operand **S**$_3$ and **S**$_4$ specify source data registers and data length. For example : **S**$_3$ specifies D150 and the value in **S**$_4$ is 10. The instruction ETHRS will send 10 bytes of data, starting from the low byte in D150, D151, D152 and so on. Users can use the instruction DTM to transform 16-bit data into 8-bit data when the transformation is required. The setting range for **S**$_4$ is 1~200 words. If the setting values exceed the setting range, the system will use the minimum (1) or the maximum (200) to operate.
5. The operand **D**$_1$ specifies a destination data register. For example, **D** specifies D10 and D10 is the received data length; the unit is byte. The data received will be stored starting from D11, low byte in D11, D12, D13 and so on. The maximum receiving data length is 200 words; data length exceeds this limit will not be stored in **D**. Users can use the instruction DTM to transform 16-bit data into 8-bit data when the transformation is required.
6. **D**$_2$ specifies the reception completion flag and only M device can be used. When the instruction is executed, and the transmission of packets is complete, this flag will be set to ON. Users can learn from this flag to see if the transmission is complete or not. Once it is set to ON, users need to set it to OFF. When there is any error occurred during the instruction execution or any timeout occurred, the flag will not be ON.
7. Once the instruction is executed, the communication begins. There is no need to use any special flag to trigger the sending. When the instruction is executed, there will be a special M shown to indicate the execution.
8. There is no limitation on the times of using this instruction in the program. However, only one instruction can be executed at a time.
9. When the instruction is forcedly stopped, the communication will also be stopped. And the completion flag **D**$_2$ will not be ON.
10. When this instruction is executed, do not use the Online Mode; otherwise errors may occur when receiving and storing data.
11. This instruction is available for the following models and firmware versions.

Series	ES2-E	12SE	26SE
Firmware version	V1.08	V1.88	V1.0

(For mode 4 in **S**$_2$, it is only available for SE PLC CPU with firmware V1.95 or later and ES2-E with firmware V1.6 or later.)

3

12. Relative special flags and registers for the instruction ETHRS:

Item	Function	Defaults	Stop⇔Run	Attributes
M1196	ON: the connection of the self-defined Ethernet communication port is enabled. When the instruction ETHRS stops, the connection will still be kept. ON=> OFF: the connection will be disabled. Off: use the instruction ETHRS to control the connection, when the instruction is executed, the connection is enabled.	Off	Off	R/W
M1197	ON: the instruction ETHRS is being executed.	Off	Off	R
M1198	ON: when there is a communication error or a communication timeout, the control on the connection of the self-defined Ethernet communication port is through M1196. When the communication timeout occurs, the communication instruction has to be stopped and then start the instruction again to start the communication.	Off	Off	R/W
D1176	Error code	0	0	R
D1227 D1228	During execution of the ETHRS instruction, if it is in the receiving mode, D1227 and D1228 show the sender's IP address. (available for ES2-E V1.2, 12SE V1.92, 26SE V2.00) (available for ES2-E with firmware V1.2, or later, 12SE with firmware V1.92 or later, and 26SE with firmware V1.00 or later)	0	-	R

13. If M1198 is ON, it means communication errors occur and an error code will be stored in D1176. For other error codes, please refer to the following table.

When **S_1**+4=0 (UDP mode)

Error code	Description
H2003	The value exceeds the range.
H600C	The local socket has been used.
H600D	Ethernet network is not connected.
H6209	UDP Socket illegal IP address
H620A	UDP Socket illegal communication mode
H620C	UDP Socket illegal address for sending data

Error code	Description
H620D	UDP Socket the length of sent data exceeds the range
H620E	UDP Socket the device where data are sent exceeds the range
H620F	UDP Socket illegal address for receiving data
H6210	UDP Socket the length of data actually received exceeds the range.
H6211	UDP Socket the device where data are received exceeds the range.
H6213	UDP Socket the size of data actually received is larger than the set data.
H6215	UDP Socket is not connected
H6217	UDP Socket connection has been triggered

When **S_1**+4=1 (TCP mode)

Error code	Description
H2003	The value exceeds the range.
H600C	The local socket has been used.
H600D	Ethernet network is not connected.
H6200	TCP Socket illegal IP address
H6201	Illegal TCP Socket communication mode setting
H6202	Illegal TCP Socket mode setting
H6203	TCP Socket illegal address for sending data
H6204	TCP Socket the length of sent data exceeds the range
H6205	TCP Socket the device where data are sent exceeds the range
H6206	TCP Socket illegal address for receiving data
H6207	TCP Socket the length of received data exceeds the range
H6208	TCP Socket the device for receiving data exceeds the range
H6212	TCP Socket communication timeout
H6213	TCP Socket the size of data actually received is larger than the set data.
H6214	TCP Socket connection is rejected by the remote equipment
H6215	TCP Socket has not been connected
H6217	TCP Socket connection has been triggered.

14. The already used communication ports are as below.

UDP/TCP	Communication Port	Description
TCP	502	Modbus TCP communication
TCP	44818	EtherNet/IP explicit message
UDP	67	DHCP communication
UDP	68	
UDP	2222	EtherNet/IP implicit message
UDP	44818	EtherNet/IP explicit message
UDP	20006	For internal parameter download
UDP	20008	

15. Descriptions for relevant flags during communication:

TCP MODE:

M1196=ON: Communication port is connected

➢ Master/Slave mode; communication is working fine.
- M1197 = ON, this indicates the communication is active. Make sure the TCP connection is ready or is waiting to be connected and checking its relative communication settings are set, the data length is less than 200 characters and if the slave is responding.
- After data is sent or received, M1197 stays ON and a completion flag will be set to ON. (You can reset this flag to OFF.)
- If the ETHRS instruction is stopped, the communication will be kept and the setting of Keep Alive Timeout will not affect the communication status.
- If the ETHRS instruction is executed again, the completion flag will be reset to OFF.
- When you set M1196 to OFF, the communication will be stopped.

➢ Master/Slave mode; an error occurs during communication.
- M1197 = ON, this indicates the communication is active. Make sure the TCP connection is ready or is waiting to be connected.
- When an error occurs, M1198 is ON and the error codes will be shown in D1176.
- Execute ETHRS instruction again, after the problem is fixed, and M1198 is reset to OFF.
- If receiving time out is enabled in Master mode, it starts counting after the sending is done.
- If receiving time out is enabled in Slave mode, it starts counting after the connection is established.

M1196=OFF: Use ETHRS instruction to control the connection; when it is executed, the connection is established.

- Master/Slave mode; communication is working fine.
  - M1197 = ON, this indicates the communication is active. Make sure the TCP connection is ready or is waiting to be connected and checking its relative communication settings are set, the data length is less than 200 characters and if the slave is responding.
  - After data is sent or received, M1197 stays ON and a completion flag will be set to ON. (You can reset this flag to OFF.)
  - When the connection time is exceeding the setting value in Keep Alive Timeout (default: 30ms), the connection will be switched off. M1197 is set to OFF.
  - If the ETHRS instruction is executed again, the completion flag will be reset to OFF.

- Master/Slave mode; an error occurs during communication.
  - M1197 = ON, this indicates the communication is active. Make sure the TCP connection is ready or is waiting to be connected.
  - When an error occurs, M1198 is ON and the error codes will be shown in D1176.
  - Execute ETHRS instruction again, after the problem is fixed, and M1198 is reset to OFF.
  - If receiving time out is enabled in Master mode, it starts counting after the sending is done.
  - If receiving time out is enabled in Slave mode, it starts counting after the connection is established.

UDP MODE:

- Master/Slave mode; communication is working fine.
  (Note: if M1196 is switched from ON to OFF during communication, the connection will be switched off. M1197 is reset to OFF and the completion flag will be set to ON.
  - M1197 = ON, this indicates the communication is active. Make sure the TCP connection is ready or is waiting to be connected and checking its relative communication settings are set, and the data length is less than 200 characters.
  - After data is sent or received, M1197 stays ON and a completion flag will be set to ON. (You can reset this flag to OFF.)
  - If the ETHRS instruction is executed again, the completion flag will be reset to OFF.

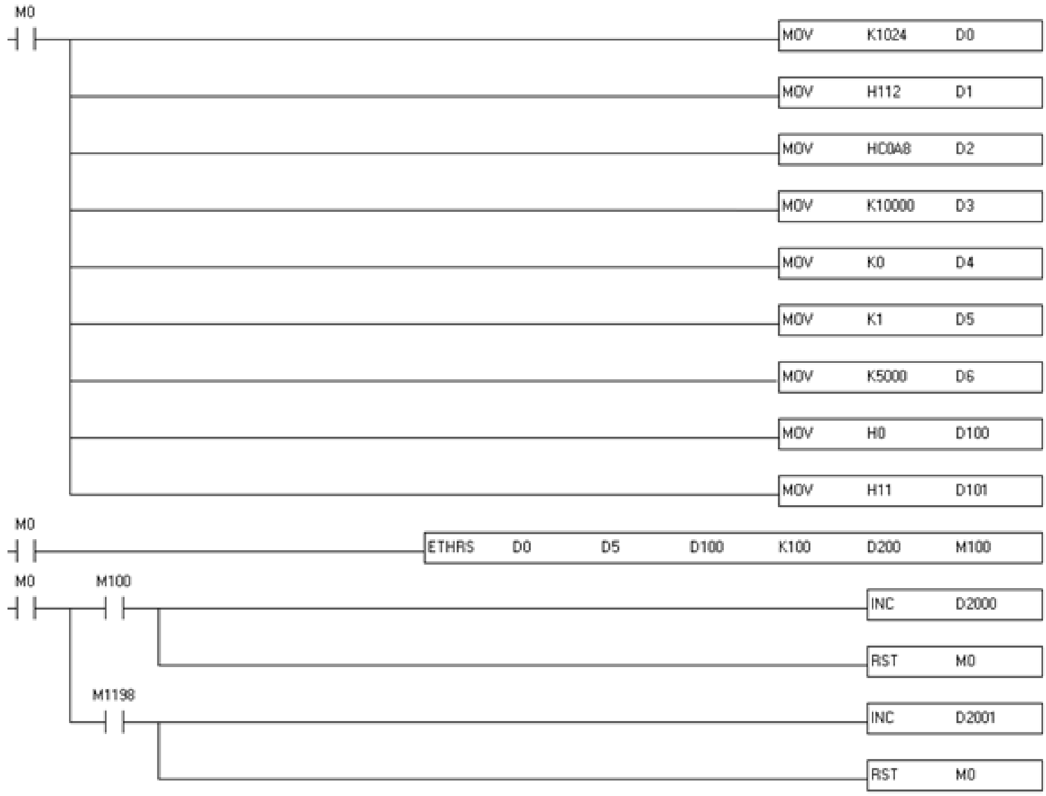
- Master/Slave mode; an error occurs during communication.
  - M1197 = ON, this indicates the communication is active. Make sure the UDP connection is ready or is waiting to be connected.
  - When an error occurs, M1198 is ON and the error codes will be shown in D1176.
  - Execute ETHRS instruction again, after the problem is fixed, and M1198 is reset to OFF.
  - If receiving time out is enabled in Master mode, it starts counting after the sending is done.
  - If receiving time out is enabled in Slave mode, it starts counting after the connection is established.

Program Example: (The command is sent and received through the Ethernet port built in DVP-SE.)

This example uses DVP-SE series as the client, M0 to activate and UDP connection mode to send and receive data. The value in **S**2 is K1. When the data is received, M100 is set to ON. The relative parameters are stated below.

TCP Socket Connection	
Remote IP	192.168.1.18
Remote port	10000
Local port	1024
Send Data Address	D100
Send Data Length	100
Receive Data Address	D200
Communication timeout (ms)	5000

1. When M0 is ON, the transmission starts and M1197 is ON. If M1198 is ON, it means communication errors occur and an error code will be stored in D1176.
2. When the data is received correctly and a response is received from the remote device, M100 will be ON. The data length and the contents will be stored in D200.

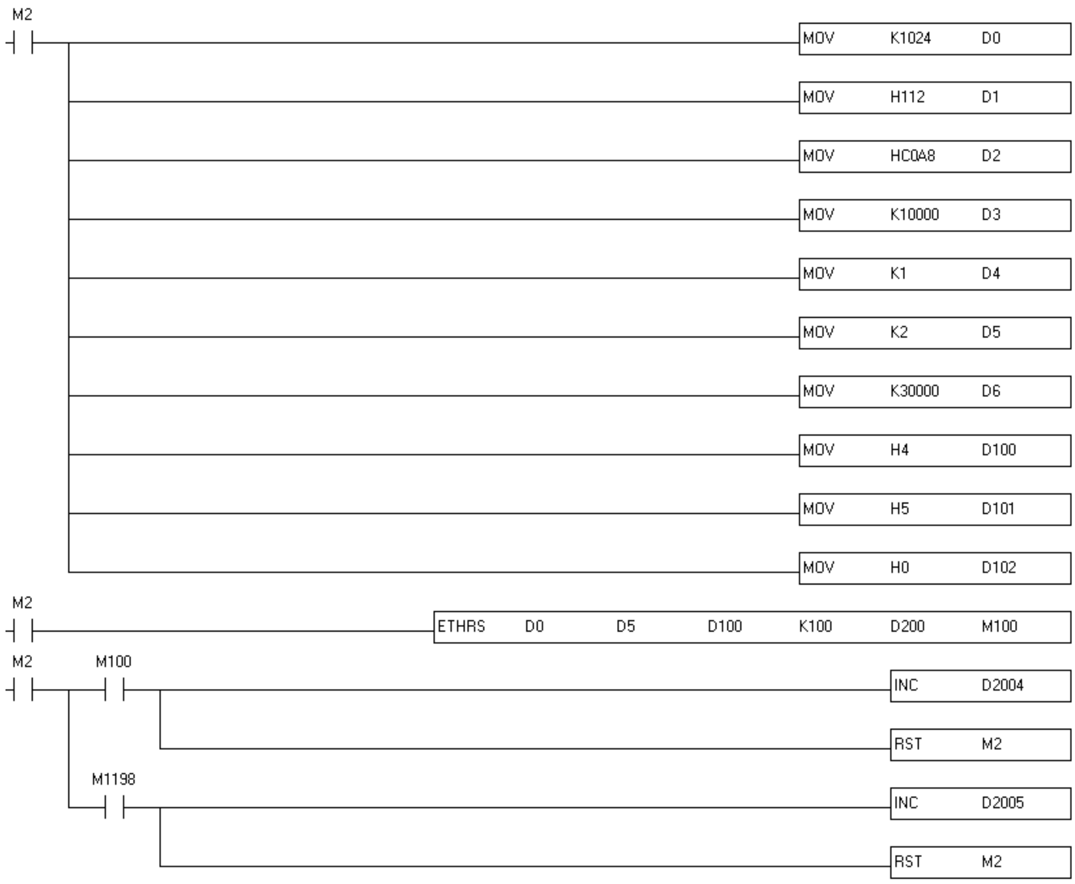


Program Example 2: (The command is sent and received through the Ethernet port built in DVP-SE.)

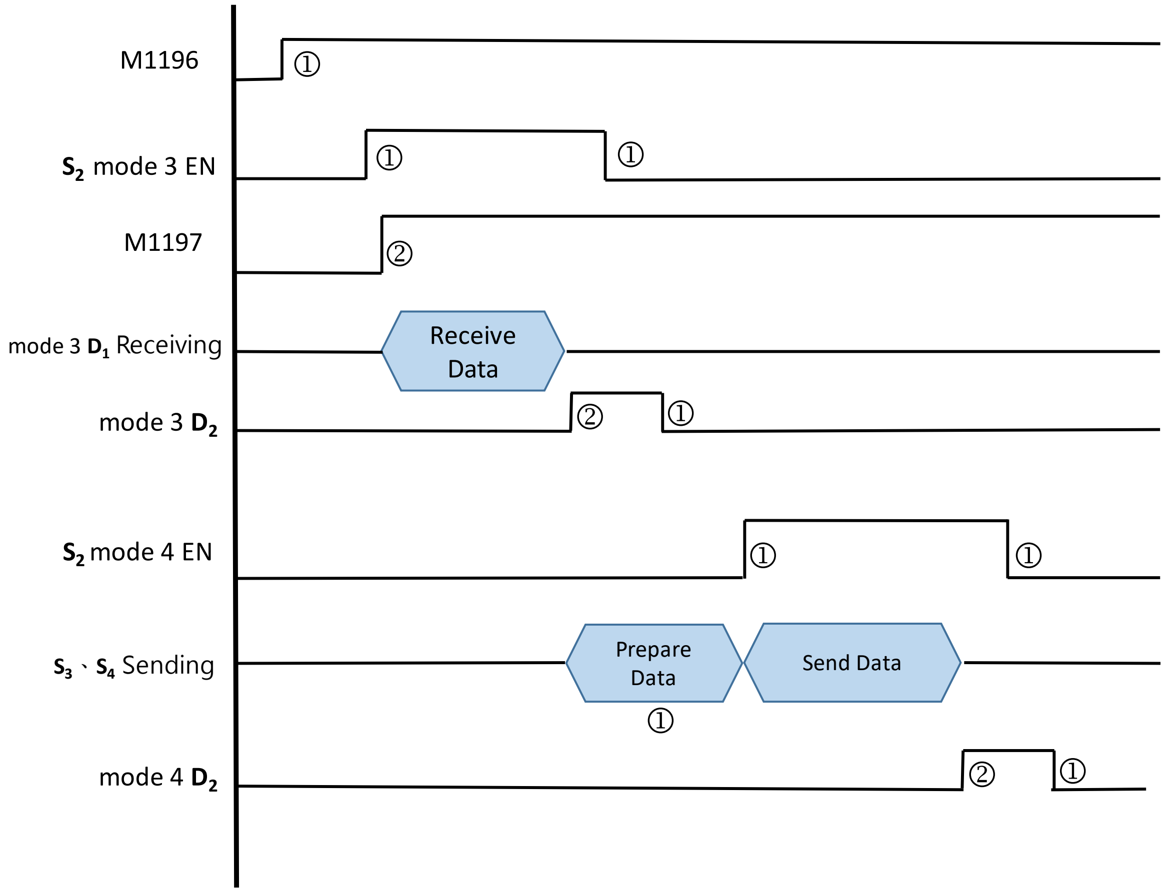
This example uses DVP-SE series as the client, M2 to activate and TCP connection mode to send and receive data. The value in **S_2** is K2. The relative parameters are stated below.

TCP Socket Connection	
Remote IP	192.168.1.31
Remote port	10000
Local port	1024
Send Data Address	D100
Send Data Length	100
Receive Data Address	D200
Communication timeout (ms)	30000

1. Set M1196 to ON. When using the TCP connection mode, it is suggested to set M1196 to ON to avoid disconnecting if a communication timeout occurs.
2. When M2 is ON, DVP-SE is waiting for the TCP connection to be established. When M100 is ON, it means the receiving is complete successfully and the data length and contents are stored in D200 and data in D100 has been sent, the data length is 100 bytes.
3. If M1198 is ON, it means communication errors occur and an error code will be stored in D1176.



Program Example 2: (The command is sent and received through the Ethernet port built in DVP-SE and mode 3 and 4 are used in S2.)



Note: ① users set and clear. ② PLC sets and clears.

Execution steps:

1. Set M1196 to ON. When using the TCP connection mode, it is suggested to set M1196 to ON to avoid disconnecting if a communication timeout occurs.
2. Activate mode 3 to receive data and wait for the completion flag to be ON, indicating the receiving is complete.
3. After receiving is complete, you need to clear the completion flag. Arrange the received data and store the data to be sent in the specified registers.
4. Activate mode 4 to send data and wait for the completion flag to be ON, indicating the sending is complete.

3.6.12 Floating Point Operation

API	Mnemonics		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
110	-	DECMP	✓	Floating point compare	✓	✓	✓	✓	-	13
111	-	DEZCP	✓	Floating point zone compare	✓	✓	✓	✓	-	17
112		DMOVR	✓	Move floating point data	✓	✓	✓	✓		9
116	-	DRAD	✓	Degree → Radian	✓	✓	✓	✓	-	9
117	-	DDEG	✓	Radian → Degree	✓	✓	✓	✓	-	9
118	-	DEBCD	✓	Float to scientific conversion	✓	✓	✓	✓	-	9
119	-	DEBIN	✓	Scientific to float conversion	✓	✓	✓	✓	-	9
120	-	DEADD	✓	Floating point addition	✓	✓	✓	✓	-	13
121	-	DESUB	✓	Floating point subtraction	✓	✓	✓	✓	-	13
122	-	DEMUL	✓	Floating point multiplication	✓	✓	✓	✓	-	13
123	-	DEDIV	✓	Floating point division	✓	✓	✓	✓	-	13
124	-	DEXP	✓	Float exponent operation	✓	✓	✓	✓	-	9
125	-	DLN	✓	Float natural logarithm operation	✓	✓	✓	✓	-	9
126	-	DLOG	✓	Float logarithm operation	✓	✓	✓	✓	-	13
127	-	DESQR	✓	Floating point square root	✓	✓	✓	✓	-	9
128	-	DPOW	✓	Floating point power operation	✓	✓	✓	✓	-	13
129	INT	DINT	✓	Float to integer	✓	✓	✓	✓	5	9
130	-	DSIN	✓	Sine	✓	✓	✓	✓	-	9
131	-	DCOS	✓	Cosine	✓	✓	✓	✓	-	9
132	-	DTAN	✓	Tangent	✓	✓	✓	✓	-	9
133	-	DASIN	✓	Arc Sine	✓	✓	✓	✓	-	9
134	-	DACOS	✓	Arc Cosine	✓	✓	✓	✓	-	9
135	-	DATAN	✓	Arc Tangent	✓	✓	✓	✓	-	9
172	-	DADDR	✓	Floating point addition	✓	✓	✓	✓	-	13
173	-	DSUBR	✓	Floating point subtraction	✓	✓	✓	✓	-	13
174	-	DMULR	✓	Floating point multiplication	✓	✓	✓	✓	-	13
175	-	DDIVR	✓	Floating point division	✓	✓	✓	✓	-	13

API	Mnemonic			Operands	Function	Controllers			
110	D	ECMP	P	(S1) (S2) (D)	Floating point compare	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DECMP, DECMPP: 13 steps
S_1					*	*							*			
S_2					*	*							*			
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

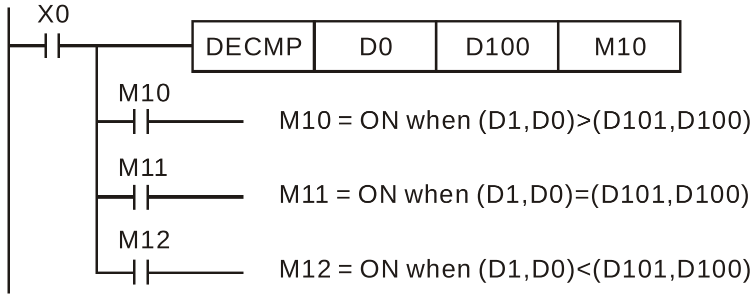
S₁: 1st comparison value **S₂**: 2nd comparison value **D**: Comparison result, 3 consecutive devices

Explanations:

1. The data of **S_1** is compared to the data of **S_2** and the result (＞, ＝, ＜) is indicated by three bit devices in **D**.
2. If the source operand **S_1** or **S_2** is specified as constant K or H, the integer value will automatically be converted to binary floating point for comparison.

Program Example:

1. If the specified device is M10, M10~M12 will automatically be used.
2. When X0 = ON, one of M10~M12 will be ON. When X0 = OFF, DECMP is not executed, M10~M12 will retain their previous state before X0 = OFF.
3. Connect M10~M12 in series or parallel for achieving the results of ≧, ≦, ≠.
4. RST or ZRST instruction is required if users need to reset the comparison result.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
111	D	EZCP	P	(S1) (S2) (S) (D)	Floating point zone compare				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DEZCP, DEZCPP: 17 steps
S_1					*	*							*			
S_2					*	*							*			
S					*	*							*			
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

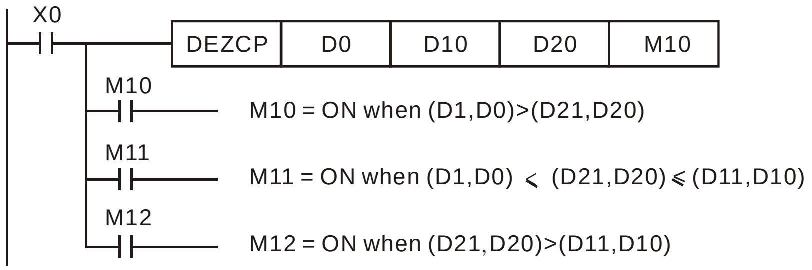
S_1: Lower bound of zone comparison S_2: Upper bound of zone comparison **S**: Comparison value **D**: Comparison result, 3 consecutive devices

Explanations:

1. The data of **S** is compared to the data range of S_1 ~ S_2 and the result (>, =, <) is indicated by three bit devices in **D**.
2. If the source operand S_1 or S_2 is specified as constant K or H, the integer value will automatically be converted to binary floating point for comparison.
3. Operand S_1 should be smaller than operand S_2. When $S_1 > S_2$, the instruction takes S_1 as the 1st comparison value and performs normal comparison similar to ECMP instruction.

Program Example:

1. If the specified device is M10, M10~M12 will automatically be used.
2. When X0 = ON, one of M10~M12 will be ON. When X0 = OFF, DEZCP instruction is not executed, M10~M12 will retain their previous state before X0 = OFF.
3. RST or ZRST instruction is required if users need to reset the comparison result.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
112	D	MOVR	P	S D	Move floating point data				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DMOVR, DMOVRP: 9 steps
S																
D								*	*	*	*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device   **D**: Destination device

Explanations:

1. **S** can only be a constant floating point value.
2. When the instruction executed, content of **S** will be moved to **D**.
3. If users want to move the floating-point value in registers, they have to use DMOV.

Program Example:

When X0 = OFF, D10 and D11 will not change. When X0 = ON, transmit F1.200E+0 (Input F1.2, and scientific notation F1.200E+0 will be displayed on ladder diagram. Users can set monitoring data format as float on the function View) to D10 and D11.

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
116	D	RAD	P	S D	Degree → Radian				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DRAD, DRADP: 9 steps
S					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC	SS2	SA2 SE	SX2

Operands:

S: Source device (degree)   **D**: Conversion result (radian)

Explanation:

1. Use the following formula to convert degree to radian:

   Radian ＝ degree × (π/180)

2. Flags: M1020 Zero flag, M1021 Borrow flag, M1022 Carry flag

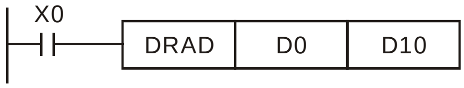
   If the absolute value of the result exceeds the max. floating point value, carry flag M1022 = ON.

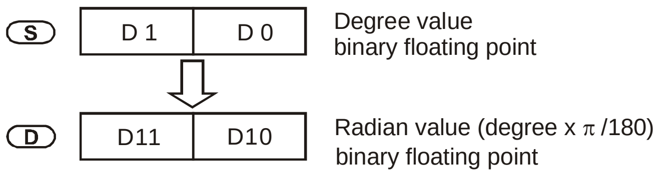
   If the absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.

   If the conversion result is 0, zero flag M1020 = ON.

Program Example:

When X0 = ON, convert degree value of the binary floating point in (D1, D0) to radian and save the binary floating point result in (D11, D10).





API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
117	D	DEG	P	(S) (D)	Radian → Degree				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DDEG, DDEGP: 9 steps
S					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device (radian) **D**: Conversion result (degree)

Explanation

1. Use the following formula to convert radian to degree:

   Degree ＝ Radian × (180/π)

   Flags: M1020 Zero flag, M1021 Borrow flag and M1022 Carry flag.

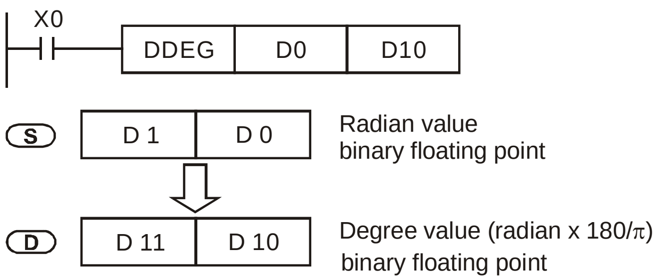
   If the absolute value of the result exceeds the max. floating point value, carry flag M1022 = ON.

   If the absolute value of the result is less than the min. floating point value, borrow flag M1021 = ON.

   If the conversion result is 0, zero flag M1020 = ON.

Program Example:

When X0 = ON, convert the radian of the binary floating point in (D1, D0) to degree and save the binary floating point result in (D11, D10).



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
118	D	EBCD	P	(S) (D)	Float to scientific conversion				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DEBCD, DEBCDP: 9 steps
S													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

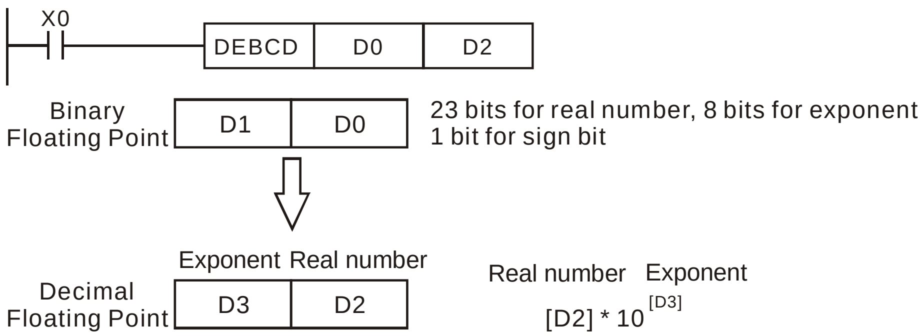
S: Source device **D**: Conversion result

Explanation

1. The instruction converts the binary floating point value in **S** to decimal floating point value and stores the results in the register specified by **D**.
2. PLC floating point is operated by the binary floating point format. DEBCD instruction is the specific instruction used to convert binary floating point to decimal floating point.
3. Flag: M1020 Zero flag, M1021 Borrow flag, M1022 Carry flag
   If absolute value of the result exceeds the max. floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than the min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.

Program Example:

When X0 = ON, the binary floating point value in D1, D0 will be converted to decimal floating point and the conversion result is stored in D3, D2.



API	Mnemonic			Operands	Function	Controllers			
119	D	EBIN	P	S D	Scientific to float conversion	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DEBIN, DEBINP: 9 steps
S													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

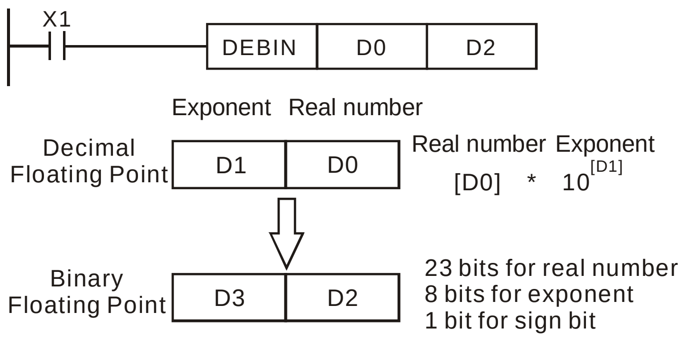
S: Source device **D**: Conversion result

Explanation:

1. The instruction converts the decimal floating point value in **S** to a binary floating point value and stores the results in the register specified by **D**.
2. For example, **S** = 1234, **S** +1 = 3. The decimal floating point value will be: 1.234 x 10^6
3. **D** must be binary floating point format. **S** and **S** +1 represent the real number and exponent of the floating point number.
4. EBIN instruction is the specific instruction used to convert decimal floating point value to binary floating point value
5. Range of real number: -9,999 ~ +9,999. Range of exponent: - 41 ~ +35. Range of PLC decimal floating point value. If the conversion result is 0, zero flag M1020 = ON.

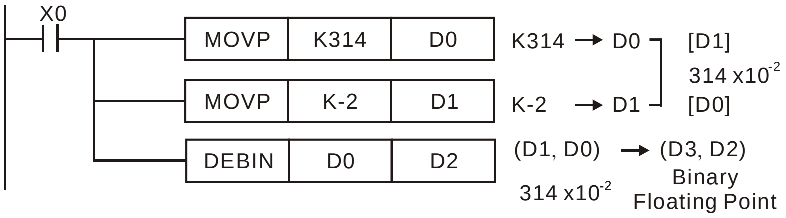
Program Example 1:

When X1 = ON, the decimal floating point value in (D1, D0) will be converted to binary floating point and the conversion result is stored in (D3, D2).



Program Example 2:

1. Use FLT instruction (API 49) to convert BIN integer into binary floating point value before performing floating point operation. The value to be converted must be BIN integer and use DEBIN instruction to convert the decimal floating point value into a binary one.
2. When X0 = ON, move K314 to D0 and K-2 to D1 to generate decimal floating point value (3.14 = 314 × 10^{-2}).



API	Mnemonic			Operands	Function	Controllers
120	D	EADD	P	S1 S2 D	Floating point addition	ES2/EX2/EC5, SS2, SA2 SE, SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			DEADD, DEADDP: 13 steps
S_2					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/EC5	SS2	SA2 SE	SX2	ES2/EX2/EC5	SS2	SA2 SE	SX2	ES2/EX2/EC5	SS2	SA2 SE	SX2

Operands:

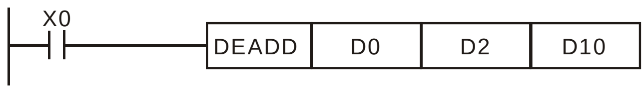
S_1: Augend **S_2**: Addend **D**: Addition result

Explanations:

1. **S_1** + **S_2** = **D**. The floating point value in **S_1** and **S_2** are added and the result is stored in **D**.
2. If the source operand **S_1** or **S_2** is specified as constant K or H, the constant will automatically be converted to binary floating point value for the addition operation.
3. **S_1** and **S_2** can designate the same register. In this case, if the instruction is specified as "continuous execution instruction" (generally DEADDP instruction) and the drive contact is ON, the register will be added once in every scan.
4. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max. floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.

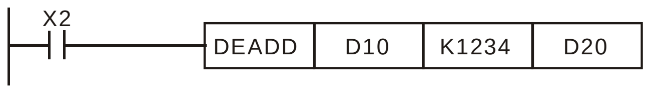
Program Example 1:

When X0 = ON, add the binary floating point value (D1, D0) with binary floating point value (D3, D2) and store the result in (D11, D10).



Program Example 2:

When X2 = ON, add the binary floating point value of (D11, D10) with K1234 (automatically converted to binary floating point value) and store the result in (D21, D20).



API	Mnemonic			Operands	Function	Controllers
121	D	ESUB	P	S1 S2 D	Floating point subtraction	ES2/EX2/EC5 SS2 SA2 SE SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			DESUB, DESUBP: 13 steps
S_2					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

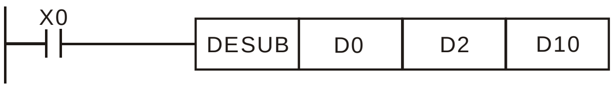
S_1: Minuend **S_2**: Subtrahend **D**: Subtraction result

Explanation:

1. **S_1 – S_2 = D**. The floating point value in **S_2** is subtracted from the floating point value in **S_1** and the result is stored in **D**. The subtraction is conducted in binary floating point format.
2. If **S_1** or **S_2** is designated as constant K or H, the instruction will convert the constant into a binary floating point value before the operation.
3. **S_1** and **S_2** can designate the same register. In this case, if the instruction is specified as "continuous execution instruction" (generally DESUBP instruction) and the drive contact is ON, the register will be subtracted once in every scan.
4. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max. floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.

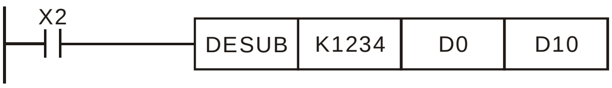
Program Example 1:

When X0 = ON, binary floating point value (D1, D0) minuses binary floating point value (D3, D2) and the result is stored in (D11, D10).



Program Example 2:

When X2 = ON, K1234 (automatically converted into binary floating point value) minuses binary floating point (D1, D0) and the result is stored in (D11, D10).



API	Mnemonic			Operands	Function	Controllers			
122	D	EMUL	P	S1 S2 D	Floating point multiplication	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			DEMUL, DEMULP: 13 steps
S_2					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

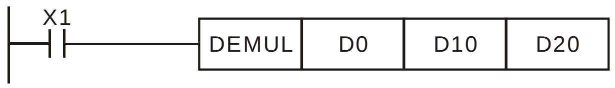
S_1: Multiplicand **S_2**: Multiplicator **D**: Multiplication result

Explanations:

1. **S_1** × **S_2** = **D**. The floating point value in **S_1** is multiplied with the floating point value in **S_2** and the result is **D**. The multiplication is conducted in binary floating point format
2. If **S_1** or **S_2** is designated as constant K or H, the instruction will convert the constant into a binary floating point value before the operation
3. **S_1** and **S_2** can designate the same register. In this case, if the instruction is specified as "continuous execution instruction" (generally DEMULP instruction) and the drive contact is ON, the register will be multiplied once in every scan.
4. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max. floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.

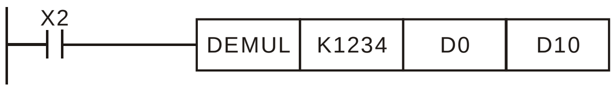
Program Example 1:

When X1 = ON, binary floating point (D1, D0) multiplies binary floating point (D11, D10) and the result is stored in (D21, D20).



Program Example 2:

When X2 = ON, K1234 (automatically converted into binary floating point value) multiplies binary floating point (D1, D0) and the result is stored in (D11, D10).



API	Mnemonic			Operands	Function	Controllers			
123	D	EDIV	P	S1 S2 D	Floating point division	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			DEADD, DEADDP: 13 steps
S_2					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

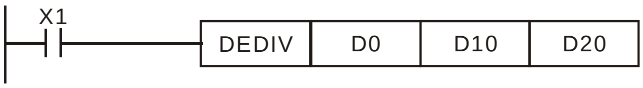
S_1: Dividend **S_2**: Divisor **D**: Quotient and Remainder

Explanation:

1. **S_1** ÷ **S_2** = **D**. The floating point value in **S_1** is divided by the floating point value in **S_2** and the result is stored in **D**. The division is conducted in binary floating point format.
2. If **S_1** or **S_2** is designated as constant K or H, the instruction will convert the constant into a binary floating point value before the operation.
3. If **S_2** = 0, operation error will occur, the instruction will not be executed
4. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max. floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.

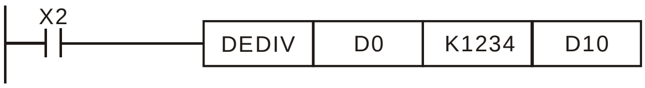
Program Example 1:

When X1 = ON, binary floating point value of (D1, D0) is divided by binary floating point (D11, D10) and the quotient and remainder is stored in (D21, D20).



Program Example 2:

When X2 = ON, binary floating point value of (D1, D0) is divided by K1234 (automatically converted to binary floating point value) and the result is stored in (D11, D10).



API	Mnemonic			Operands	Function	Controllers			
124	D	EXP	P	(S) (D)	Float exponent operation	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S					*	*							*			DEXP, DEXPP: 9 steps
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

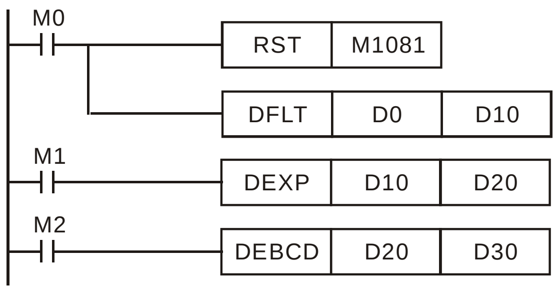
S: Exponent **D**: Operation result

Explanations:

1. The base is e = 2.71828 and exponent is **S**
2. EXP [**S +1, S**] = [**D** +1, **D**]
3. Both positive and negative values are valid for **S**. Register **D** has to be 32-bit format. Operation is conducted in floating point value, so the value in **S** needs to be converted into floating value before exponent operation.
4. The content in **D**: e ^{S}, e =2.71828 and **S** is the specified exponent..
5. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag).
   If absolute value of the result is larger than max. floating value, carry flag M1022 = ON.
   If absolute value of the result is smaller than min. floating value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.

Program Example:

1. When M0 = ON, convert (D1, D0) to binary floating value and save the result in (D11, D10).
2. When M1= ON, perform exponent operation with (D11, D10) as the exponent. The value is saved in register (D21, D20) in binary floating format.
3. When M2 = ON, convert the value in (D21, D20) into decimal floating point value and save the result in (D31, D30). (At this time, D31 indicates powers of 10 for D30)



API	Mnemonic			Operands		Function	Controllers			
							ES2/EX2/ EC5	SS2	SA2 SE	SX2
125	D	LN	P	S	D	Float natural logarithm operation				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DLN, DLNP: 9 steps
S					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

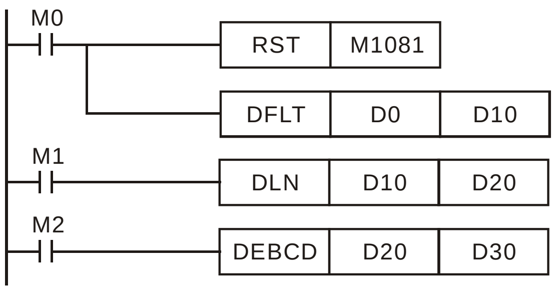
S: Source device   **D**: Operation result

Explanations:

1. Perform natural logarithm (LN) operation on operand **S**:
   LN[**S** +1, **S**]=[**D** +1, **D**]
2. Only a positive number is valid for **S**. Register **D** has to be 32-bit format. Operation is conducted in floating point value, so the value in **S** needs to be converted into floating value before natural logarithm operation.
3. e^D = **S**. The content of **D** = LN **S,** where the value in **S** is specified by users.
4. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag).
   If absolute value of the result is larger than max. floating value, carry flag M1022 = ON.
   If absolute value of the result is smaller than min. floating value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON

Program Example:

1. When M0 = ON, convert (D1, D0) to binary floating value and save the result in (D11, D10).
2. When M1= ON, perform natural logarithm operation with (D11, D10) as the antilogarithm. The value is saved in register (D21, D20) in binary floating format.
3. When M2 = ON, convert the value in (D21, D20) into decimal floating point value and save the result in (D31, D30). (At this time, D31 indicates powers of 10 for D30)



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
126	D	LOG	P	S1 S2 D	Float logarithm operation				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																DLOG, DLOGP: 13 steps
S_1					*	*							*			
S_2					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Base **S_2**: Antilogarithm **D**: Operation result

Explanations:

1. Perform logarithm operation with **S_1** as the base and **S_2** as the antilogarithm and save the result in **D**.
2. Only a positive number is valid for **S**. Register **D** has to be 32-bit format. Operation is conducted in floating point value, so the value in **S** needs to be converted into floating value before logarithm operation.
3. Logarithm operation: $S_1^D = S_2$, D = ? → $Log_{S1}^{S2} = D$

   Example: Assume $S_1 = 5$, $S_2 = 125$, $S_1^D = S_2$, D = ? → $5^D = 125$ → $D = Log_{S1}^{S2} = log_5^{125} = 3$.
4. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag).

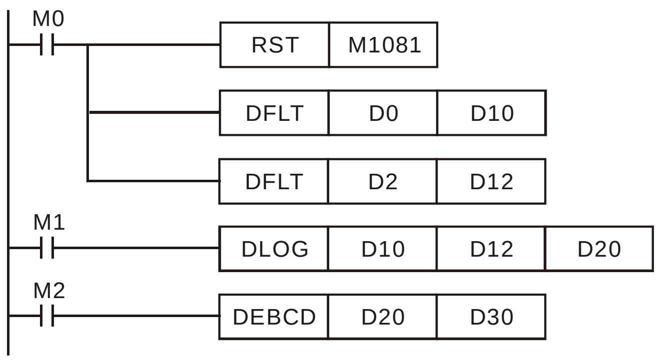
   If absolute value of the result is larger than max. floating value, carry flag M1022 = ON.

   If absolute value of the result is smaller than min. floating value, borrow flag M1021 = ON.

   If the conversion result is 0, zero flag M1020 = ON.

Program Example:

1. When M0 = ON, convert (D1, D0) and (D3, D2) to binary floating value and save the result in register (D11, D10) and (D13, D12) individually.
2. When M1= ON, perform logarithm operation with (D11, D10) as base and (D13, D12) as antilogarithm. The results are saved in register (D21, D20) in binary floating format.
3. When M2 = ON, convert the value in (D21, D20) into decimal floating point value and save the result in (D31, D30). (At this time, D31 indicates powers of 10 for D30)



API	Mnemonic			Operands	Function	Controllers			
127	D	ESQR	P	(S) (D)	Floating point square root	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices X	Y	M	S	Word devices K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S					*	*							*			DESQR, DESQRP: 9 steps
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device **D**: Operation result

Explanations:

1. This instruction performs a square root operation on the floating point value in **S** and stores the result in **D**. All data will be operated in binary floating point format and the result will also be stored in floating point format.
2. If the source device **S** is specified as constant K or H, the integer value will automatically be converted to binary floating value.
3. If operation result of **D** is 0 (zero), Zero flag M1020 = ON.
4. **S** can only be a positive value. Performing any square root operation on a negative value will result in an “operation error” and instruction will not be executed. M1067 and M1068 = ON and error code “0E1B” will be recorded in D1067.
5. Flags: M1020 (Zero flag), M1067 (Program execution error), M1068 (Execution Error Locked)

Program Example 1:

When X0 = ON, the square root of binary floating point (D1, D0) is stored in (D11, D10) after the operation of square root.

$\sqrt{(D1, D0)} \rightarrow (D11, D10)$

Binary floating point Binary floating point

Program Example 2:

When X2 = ON, the square root of K1234 (automatically converted to binary floating value) is stored in (D11, D10).

API	Mnemonic			Operands	Function	Controllers			
128	D	POW	P	S1 S2 D	Floating point power operation	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			DPOW, DPOWP: 13 steps
S_2					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

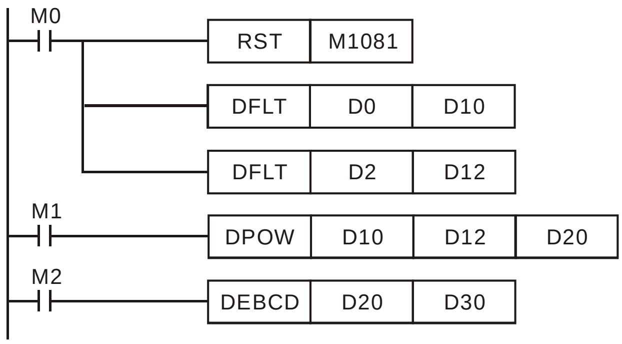
S_1: Base **S_2**: Exponent **D**: Operation result

Explanations:

1. Perform power operation on binary floating value **S_1** and **S_2** and save the result in **D**.
   POW [**S_1**+1, **S_1**]^[**S_2**+1, **S_2**] = **D**
2. Only a positive number is valid for **S**. Register **D** has to be 32-bit format. Operation is conducted in floating point value, so the value in **S_1** and **S_2** needs to be converted into floating value before power operation.
3. Example of power operation:
   When S_1^{S2} = D, D = ? Assume **S_1** = 5, **S_2** = 3, **D** = 5^3 =125
4. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag).
   If absolute value of the result is larger than max. floating value, carry flag M1022 = ON.
   If absolute value of the result is smaller than min. floating value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.

Program Example:

1. When M0 = ON, convert (D1, D0) and (D3, D2) to binary floating value and save the result in register (D11, D10) and (D13, D12) individually.
2. When M1 = ON, perform power operation with (D11, D10) as base and (D13, D12) as exponent. The value is saved in register (D21, D20) in binary floating format.
3. When M2 = ON, convert the value in (D21, D20) into decimal floating point value and save the result in (D31, D30). (At this time, D31 indicates powers of 10 for D30)



API	Mnemonic			Operands	Function	Controllers			
129	D	INT	P	S D	Float to integer	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	INT, INTP: 5 steps
S											*	*	*			DINT, DINTP: 9 steps
D											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device **D**: Operation result

Explanations:

1. The binary floating point value in the register **S** is converted to BIN integer and stored in register **D**. The decimal of the operation result will be left out.
2. This instruction is the opposite of the API 49 (FLT) instruction.
3. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag).

   If the conversion result is 0, zero flag M1020 = ON.

   If there is any decimal left out, borrow flag M1021 = ON.

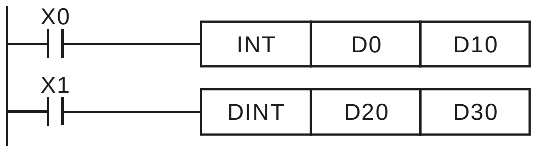
   If the conversion result is larger than the below range, carry flag M1022 = ON

   16-bit instruction: -32,768 ~ 32,767

   32-bit instruction: -2,147,483,648 ~ 2,147,483,647

Program Example:

1. When X0 = ON, the binary floating point value of (D1, D0) will be converted to BIN integer and the result is stored in D10. The decimal of the result will be left out.
2. When X1 = ON, the binary floating point value of (D21, D20) will be converted to BIN integer and the result is stored in (D31, D30). The decimal of the result will be left out.



API	Mnemonic			Operands	Function	Controllers			
130	D	SIN	P	S D	Sine	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DSIN, DSINP: 9 steps
S					*	*							*			
D													*			

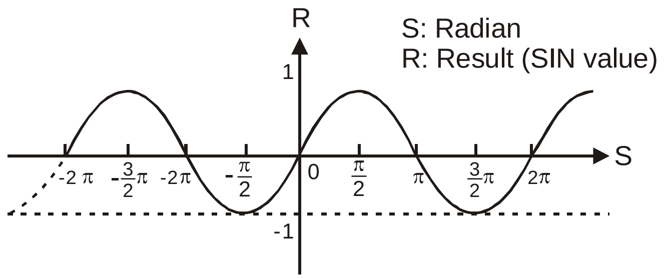
PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device (0°≦**S**＜360°) **D**: Operation result

Explanations:

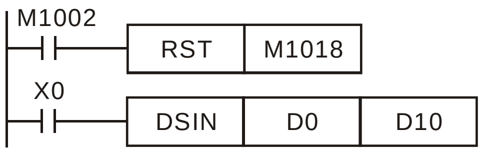
1. SIN instruction performs sine operation on **S** and stores the result in **D**.
2. The value in **S** can be set as radian or degree by flag M1018.
3. M1018 = OFF, radian mode. RAD = degree ×π/180.
4. M1018 = ON, degree mode. Degree range: 0°≦degree＜360°.
5. Flag: M1018 (Flag for Radian/Degree)
6. See the figure below for the relation between the radian and the operation result:

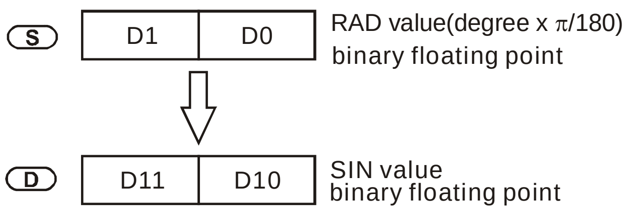


7. If operation result in **D** is 0, Zero flag M1020 = ON.

Program Example 1:

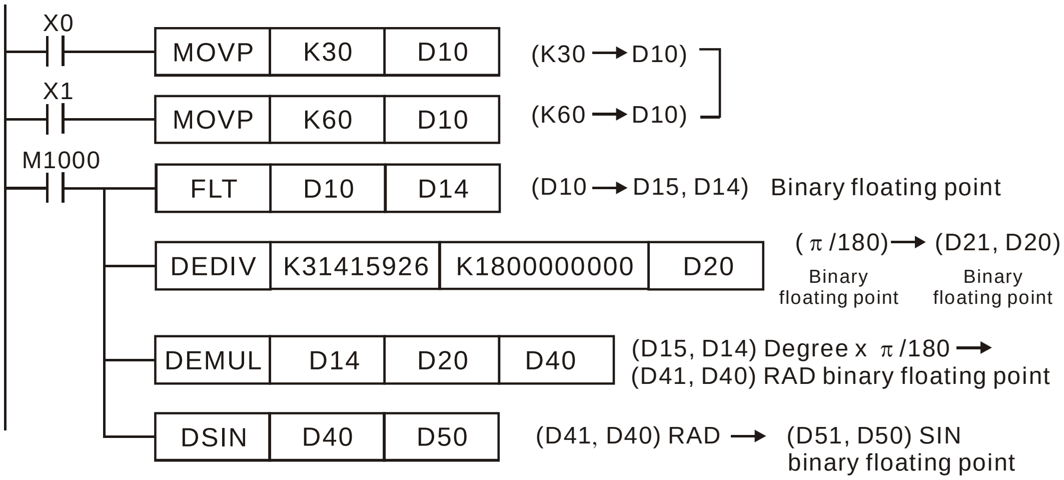
M1018 = OFF, radian mode. When X0 = ON, DSIN instruction conducts sine operation on binary floating value in (D1, D0) and stores the SIN value in (D11, D10) in binary floating format.





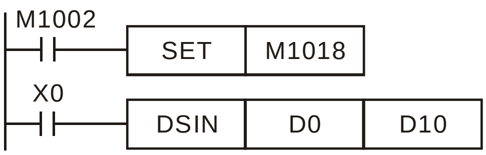
Program Example 2:

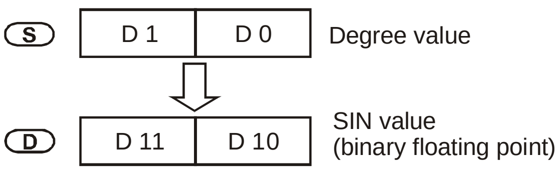
M1018 = OFF, radian mode. Select the degree value from inputs X0 and X1 and convert it to RAD value for further sine operation.



Program Example 3:

M1018 = ON, degree mode. When X0 = ON, DSIN instruction performs sine operation on the degree value (0°≦degree＜360°) in (D1, D0) and stores the SIN value in (D11, D10) in binary floating format.





API	Mnemonic			Operands	Function	Controllers
131	D	COS	P	S D	Cosine	ES2/EX2/EC5, SS2, SA2 SE, SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											DCOS, DCOSP: 9 steps
S					*	*							*			
D													*			

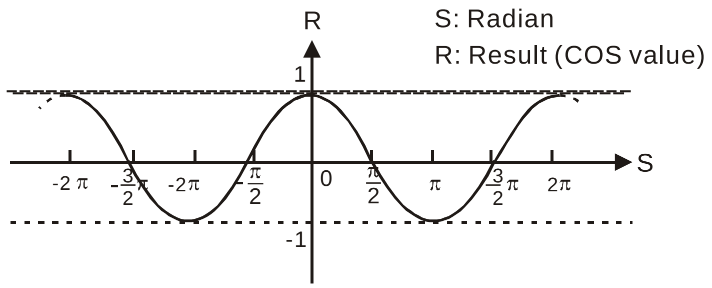
PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device (0°≦**S**＜360°) **D**: Operation result

Explanations:

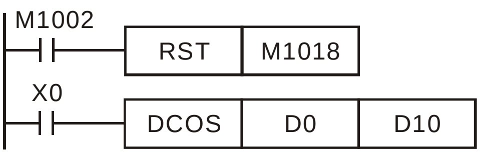
1. COS instruction performs cosine operation on **S** and stores the result in **D**.
2. The value in **S** can be set as radian or degree by flag M1018.
3. M1018 = OFF, radian mode. RAD = degree ×π/180.
4. M1018 = ON, degree mode. Degree range: 0°≦degree＜360°.
5. Flag: M1018 (Flag for Radian/Degree)
6. See the figure below for the relation between the radian and the operation result:

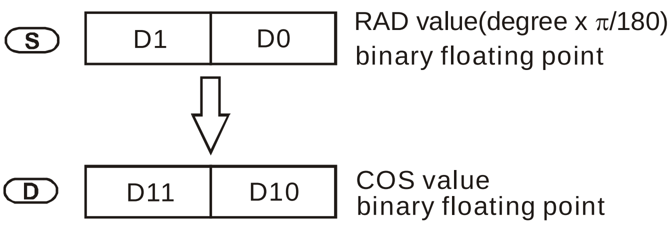


7. If operation result in **D** is 0, Zero flag M1020 = ON.

Program Example 1:

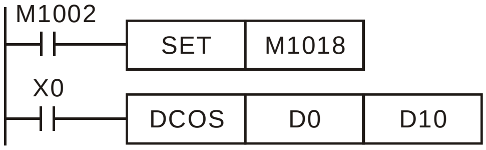
M1018 = OFF, radian mode. When X0 = ON, DCOS instruction conducts cosine operation on binary floating value in (D1, D0) and stores the COS value in (D11, D10) in binary floating format.

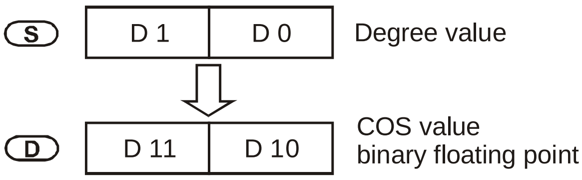




Program Example 2:

M1018 = ON, degree mode. When X0 = ON, DCOS instruction performs cosine operation on the degree value (0°≦degree＜360°) in (D1, D0) and stores the COS value in (D11, D10) in binary floating format..





API	Mnemonic			Operands	Function	Controllers			
132	D	TAN	P	S D	Tangent	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											DTAN, DTANP: 9 steps
S					*	*							*			
D													*			

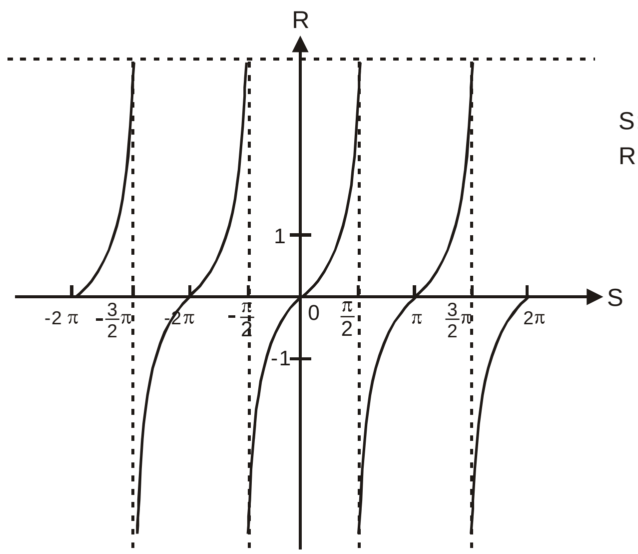
PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device (0°≦**S**＜360°) **D**: Operation result

Explanations:

1. TAN instruction performs tangent operation on **S** and stores the result in **D**.
2. The value in **S** can be set as radian or degree by flag M1018.
3. M1018 = OFF, radian mode. RAD = degree ×π/180.
4. M1018 = ON, degree mode. Degree range: 0°≦degree＜360°.
5. Flag: M1018 (Flag for Radian/Degree)
6. See the figure below for the relation between the radian and the operation result

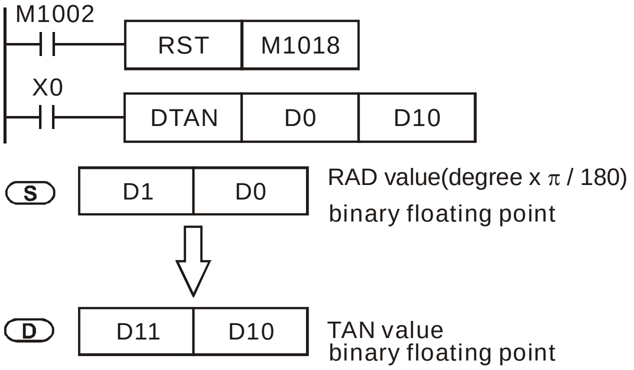


S: Radian
R: Result (TAN value)

7. If operation result in **D** is 0, Zero flag M1020 = ON.

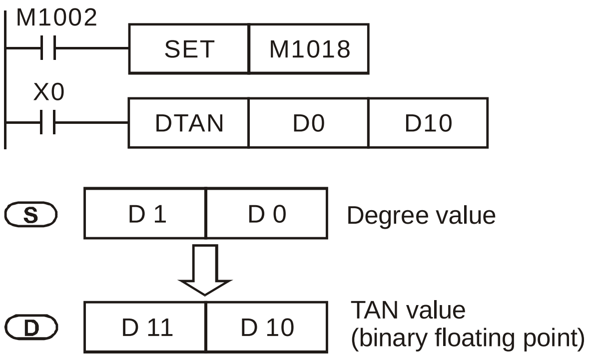
Program Example 1:

M1018 = OFF, radian mode. When X0 = ON, DTAN instruction performs tangent operation on the radian value in (D1, D0) and stores the TAN value in (D11, D10) in binary floating format.



Program Example 2:

M1018 = ON, degree mode. When X0 = ON, DTAN instruction performs tangent operation on the degree value (0°≦degree＜360°) in (D1, D0) and stores the TAN value in (D11, D10) in binary floating format.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
133	D	ASIN	P	S D	Arc Sine				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																DASIN, DASINP: 9 steps
S					*	*							*			
D													*			

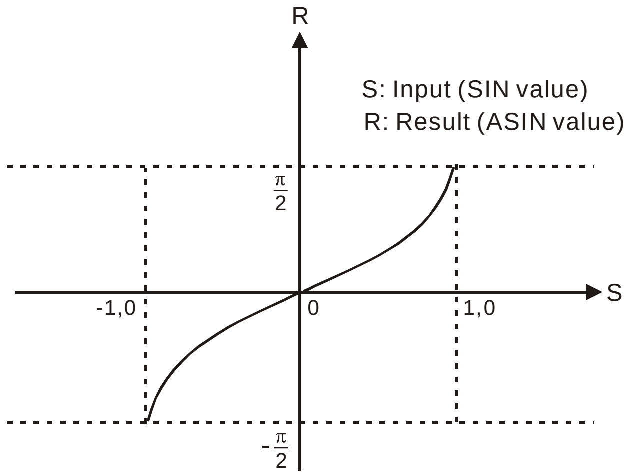
PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device (binary floating value) **D**: Operation result

Explanations:

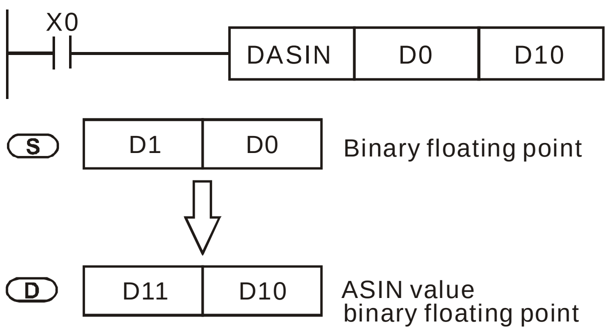
1. ASIN instruction performs arc sine operation on **S** and stores the result in **D**
2. ASIN value = SIN^{-1}
3. See the figure below for the relation between input **S** and the result:



4. If operation result in **D** is 0, Zero flag M1020 = ON.
5. The decimal value of the SIN value designated by **S** should be within -1.0 ~ +1.0. If the value exceeds the range, M1067 and M1068 will be ON and instruction will be disabled.

Program Example:

When X0 = ON, DASIN instruction performs arc sine operation on the binary floating value in (D1, D0) and stores the ASIN value in (D11, D10) in binary floating format..



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
134	D	ACOS	P	S D	Arc Cosine				

Type / OP	Bit Devices X	Y	M	S	Word devices K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S					*	*							*			DACOS, DACOSP: 9 steps
D													*			

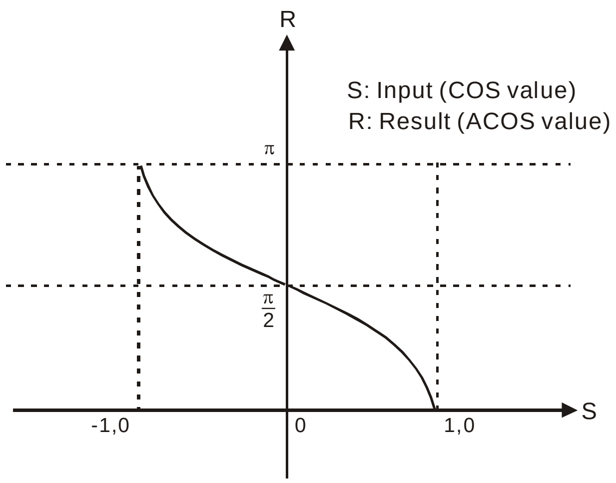
PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device (binary floating value) **D**: Operation result

Explanations:

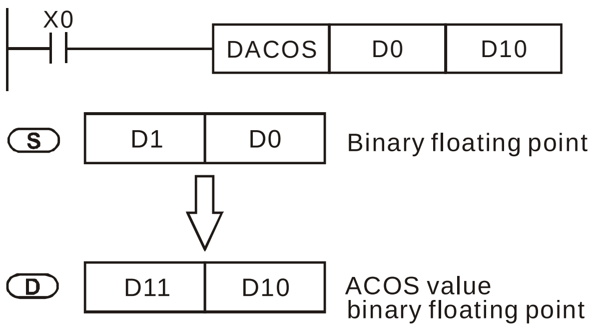
1. ACOS instruction performs arc cosine operation on **S** and stores the result in **D**
2. ACOS value = COS^{-1}
3. See the figure below for the relation between the input **S** and the result:



4. If operation result in **D** is 0, Zero flag M1020 = ON.
5. The decimal value of the COS value designated by **S** should be within -1.0 ~ +1.0. If the value exceeds the range, M1067 and M1068 will be ON and instruction will be disabled.

Program Example:

When X0 = ON, DACOS instruction performs arc cosine operation on the binary floating value in (D1, D0) and stores the ACOS value in (D11, D10) in binary floating format.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
135	D	ATAN	P	S D	Arc Tangent				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DATAN, DATANP: 9 steps
S					*	*							*			
D													*			

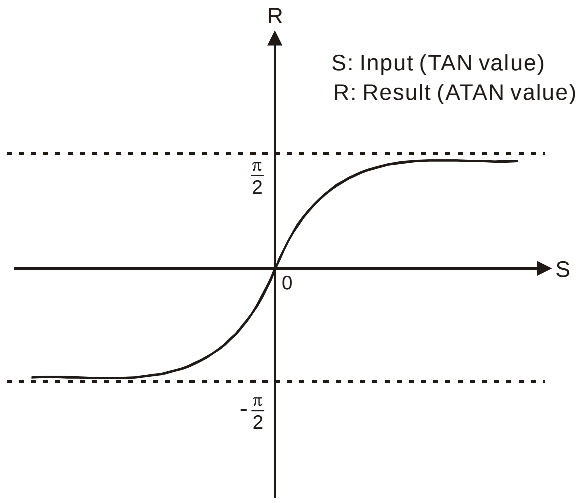
PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device (binary floating value)   **D**: Operation result

Explanations:

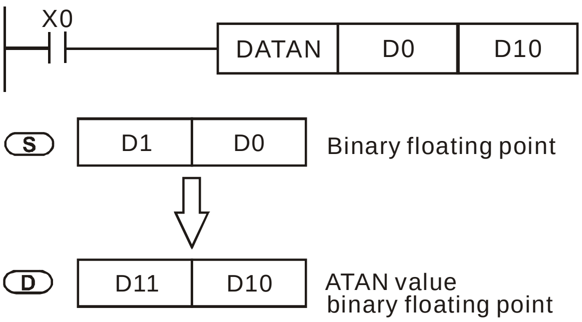
1. ATAN instruction performs arc tangent operation on **S** and stores the result in **D**
2. ATAN value=TAN^{-1}
3. See the figure below for the relation between the input and the result:



4. If operation result in **D** is 0, Zero flag M1020 = ON.

Program Example:

When X0 = ON, DATAN instruction performs arc tangent operation on the binary floating value in (D1, D0) and stores the ATAN value in (D11, D10) in binary floating format.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
172	D	ADDR	P	S1 S2 D	Floating point addition				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DADDR, DADDRP: 13 steps
S_1													*			
S_2													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

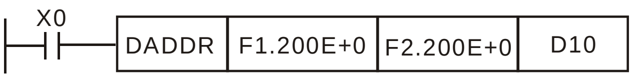
S_1: Floating point summand **S_2:** Floating point addend **D:** Sum

Explanations:

1. ADDR instruction adds the floating point summand **S_1** with floating point addend **S_2** and stores the operation result in **D**.
2. In ADDR instruction, floating point values can be directly entered into **S_1** and **S_2**.
3. In DADDR instruction, floating point values (e.g. F1.2) can be either entered directly into **S_1** and **S_2** or stored in data registers for operation.
4. When **S_1** and **S_2** is specified as data registers, the function of DADDR instruction is the same as API 120 EADD instruction.
5. **S_1** and **S_2** can designate the same register. In this case, if the instruction is specified as "continuous execution instruction" (generally DADDRP instruction) and the drive contact is ON, the register will be added once in every scan.
6. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON

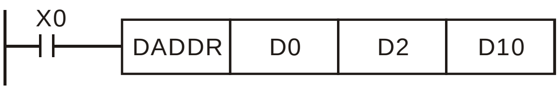
Program Example 1:

When X0 = ON, add floating point number F1.200E+0 (Input F1.2, and scientific notation F1.200E+0 will be displayed on ladder diagram. Users can set monitoring data format as float on the function View) with F2.200E+0 and store the obtained result F3.400E+0 in register D10 and D11.



Program example 2:

When X0 = ON, add floating point value (D1, D0) with (D3, D2) and store the result in (D11, D10).



API	Mnemonic			Operands	Function	Controllers			
173	D	SUBR	P	S1 S2 D	Floating point subtraction	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DSUBR: 13 steps
S_1													*			
S_2													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

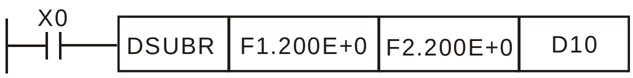
S_1: Floating point minuend **S_2:** Floating point subtrahend **D:** Remainder

Explanations:

1. SUBR instruction subtracts **S_1** with **S_2** and stores the operation result in **D**.
2. In SUBR instruction, floating point values can be directly entered into **S_1** and **S_2**..
3. In DSUBR instruction, floating point values (e.g. F1.2) can be either entered directly into **S_1** and **S_2** or stored in data registers for operation.
4. When **S_1** and **S_2** is specified as data registers, the function of DSUBR instruction is the same as API 121 ESUB instruction.
5. **S_1** and **S_2** can designate the same register. In this case, if the instruction is specified as "continuous execution instruction" (generally DSUBRP instruction) and the drive contact is ON, the register will be subtracted once in every scan.
6. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON

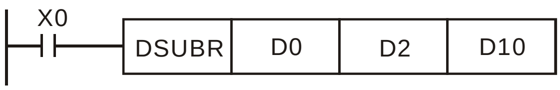
Program example 1:

When X0 = ON, subtract floating point number F1.200E+0 (Input F1.2, and scientific notation F1.200E+0 will be displayed on ladder diagram. Users can set monitoring data format as float on the function View) with F2.200E+0 and store the obtained result F-1.000E+0 in register D10 and D11.



Program example 2:

When X0 = ON, subtract the floating point value (D1, D0) with (D3, D2) and store the result in (D11, D10).



API	Mnemonic			Operands	Function	Controllers			
174	D	MULR	P	S1 S2 D	Floating point multiplication	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1													*			DMULR, DMULRP: 13 steps
S_2													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

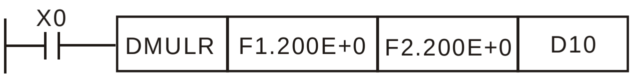
S_1: Floating point multiplicand **S_2:** Floating point multiplicator **D:** Product

Explanations:

1. MULR instruction multiplies **S_1** with **S_2** and stores the operation result in **D**.
2. In MULR instruction, floating point values can be directly entered into **S_1** and **S_2**.
3. In DMULR instruction, floating point values (e.g. F1.2) can be either entered directly into **S_1** and **S_2** or stored in data registers for operation.
4. When **S_1** and **S_2** is specified as data registers, the function of DMULR instruction is the same as API 122 EMUL instruction.
5. **S_1** and **S_2** can designate the same register. In this case, if the instruction is specified as “continuous execution instruction” (generally DMULRP instruction) and the drive contact is ON, the register will be multiplied once in every scan
6. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.

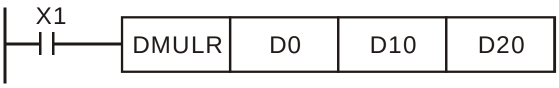
Program Example 1:

When X0= ON, multiply floating point number F1.200E+0 (Input F1.2, and scientific notation F1.200E+0 will be displayed on ladder diagram. Users can set monitoring data format as float on the function View) with F2.200E+0 and store the obtained result F2.640E+0 in register D10 and D11.



Program example 2:

When X1= ON, multiply the floating point value (D1, D0) with (D11, D10) and store the result in (D21, D20).



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
175	D	DIVR	P	S1 S2 D	Floating point division				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
S_1													*			DDIVR: 13 steps
S_2													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

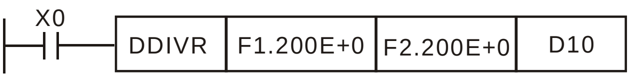
S_1: Floating point n dividend **S_2:** Floating point divisor **D:** Quotient

Explanations:

1. DIVR instruction divides **S_1** by **S_2** and stores the operation result in **D**
2. In DIVR instruction, floating point values can be directly entered into **S_1** and **S_2**.
3. In DDIVR instruction, floating point values (e.g. F1.2) can be either entered directly into **S_1** and **S_2** or stored in data registers for operation.
4. When **S_1** and **S_2** is specified as data registers, the function of DDIVR instruction is the same as API 123 EDIV instruction.
5. If **S_2** = 0, operation error occurs and M1067 = ON, M1068 = ON. D1067 stores the error code 0E19 (HEX).
6. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.

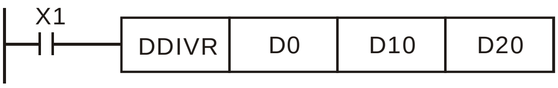
Program example 1:

When X0 = ON, divide floating point number F1.200E+0 (Input F1.2, and scientific notation F1.200E+0 will be displayed on ladder diagram. Users can set monitoring data format as float on the function View) with F2.200E+0 and store the obtained result F0.545E+0 in D10 and D11.



Program example 2:

When X1= ON, divide the floating point number value (D1, D0) by (D11, D10) and store the obtained quotient into registers (D21, D20).



3.6.13 Additional Instruction

API	Mnemonic		PULSE	Function	Applicable to					STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2	SX2	SE	16-bit	32-bit
143	DELAY	-	✓	Delay	✓	✓	✓	✓	✓	3	-
144	GPWM	-	-	General PWM output	✓	✓	✓	✓	✓	7	-
145	FTC	-		Fuzzy Temperature Control	ES2 EX2 V3.22		V2.66	V2.66		7	-
147	SWAP	DSWAP	✓	Byte swap	✓	✓	✓	✓	✓	3	5
148	MEMR	-	✓	Reading the data from the file register	ES2 EX2	-	✓	✓	-	7	-
149	MEMW	-	✓	Writing the data into the file register	ES2 EX2	-	✓	✓	-	7	-
154	RAND	DRAND	✓	Random number	✓	✓	✓	✓	✓	7	13
168	MVM	DMVM	✓	Mask and combine designated Bits	✓	✓	✓	✓	✓	7	13
176	MMOV	–	✓	16-bit→32-bit Conversion	✓	✓	✓	✓	✓	5	–
177	GPS	-	-	GPS data receiving	ES2 EX2	✓	✓	✓	-	5	-
178	-	DSPA	-	Solar cell positioning	ES2 EX2	✓	✓	✓	-	–	9
179	WSUM	DWSUM	✓	Sum of multiple devices	✓	✓	✓	✓	✓	7	13
202	SCAL	-	✓	Proportional value calculation	✓	✓	✓	✓	✓	9	-
203	SCLP	DSCLP	✓	Parameter proportional value calculation	✓	✓	✓	✓	✓	9	13
205	CMPT	DCMPT	✓	Compare table	ES2 EX2	✓	✓	✓	✓	9	17
207	CSFO	-	-	Catch speed and proportional output	ES2 EX2	✓	✓	✓	-	7	-

API	Mnemonic		Operands	Function	Controllers			
143	DELAY	P	S	Delay	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DELAY, DELAYP: 3 steps
S					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Delay time, unit: 0.1ms (K1~K1000)

Please see the explanation below for more information about the unit of a delay.

Explanations: (The instruction can be used in DVP-ES2/EX2 series PLCs whose version is 3.00; DVP-SS2 series PLCs whose version is 2.80; DVP-SA2 series PLCs whose version is 2.40; DVP-SX2 series PLCs whose version is 2.20; DVP-SE series PLCs whose version is 1.20 (and below).)

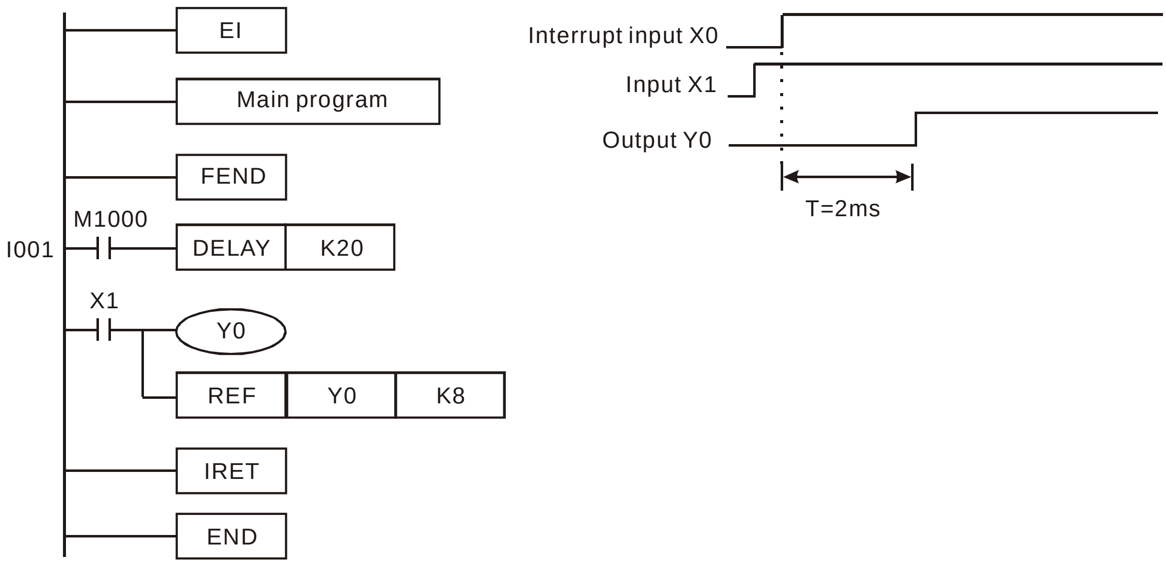
1. The unit of a delay is 100us.
2. When DELAY instruction executes, in every scan cycle, the execution of the program after DELAY instruction will be delayed according to the delay time.

Explanations: (The instruction can be used in DVP-ES2/EX2 series PLCs whose version is 3.20; EC5: V1.00；DVP-SS2 series PLCs whose version is 3.00/DVP-SA2 series PLCs whose version is 2.60/DVP-SX2 series PLCs whose version is 2.40/DVP-SE series PLCs whose version is 1.40 (and above).)

1. The unit of a delay depends on M1148. If M1148 is Off, the unit of a delay is 100us. If N1148 is On, the unit of a delay is 5us.
2. When the instruction DELAY is executed, the unit of a delay will be 5us if M1148 is On. After the instruction is executed, M1148 will be set to Off.
3. After the instruction is executed, the execution of the program following DELAY will be delayed for a period of time set by users.

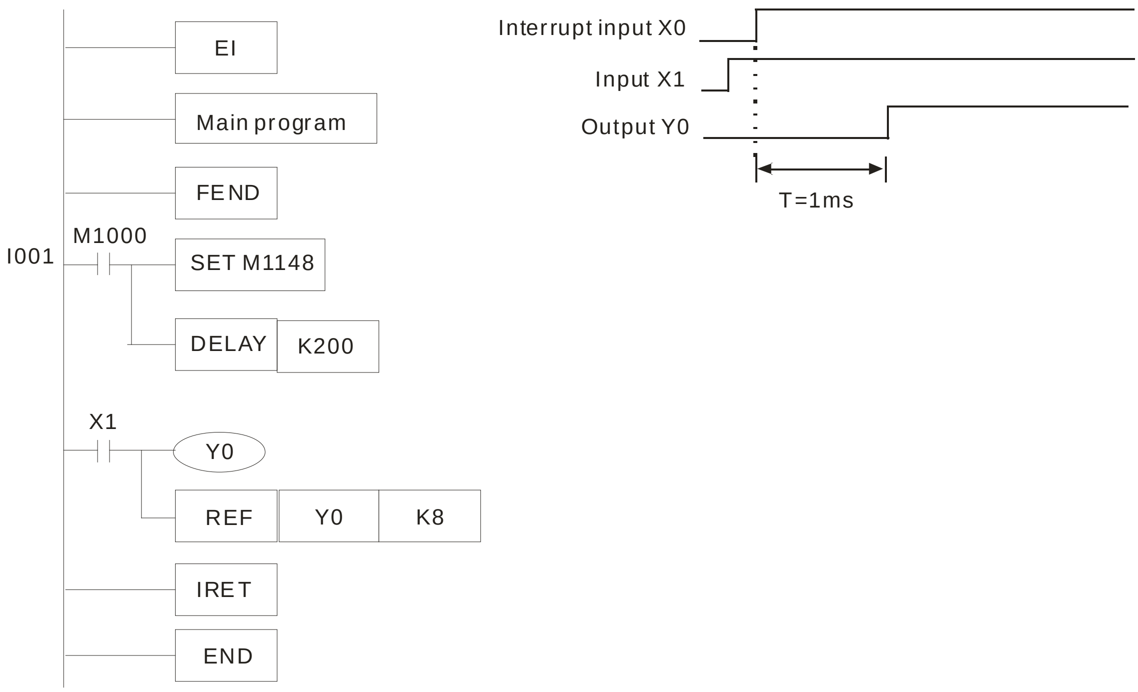
Program Example: (for DVP-ES2/EX2 series PLCs whose version is 3.00; DVP-SS2 series PLCs whose version is 2.80; DVP-SA2 series PLCs whose version is 2.40; DVP-SX2 series PLCs whose veresion is 2.20; DVP-SE series PLCs whose version is 1.20 (and below))

When interrupt input X0 is triggered from OFF to ON, interrupt subroutine executes DELAY instruction first, therefore the program after DELAY instruction (X1 = ON, Y0 = ON…) will be delayed for 2ms.



Program Example: (for DVP-ES2/EX2 series PLCs whose version is 3.20; EC5: V1.00; DVP-SS2 series PLCs whose version is 3.00; DVP-SA2 series PLCs whose version is 2.60; DVP-SX2 series PLCs whose veresion is 2.40; DVP-SE series PLCs whose version is 1.40 (and above))

When interrupt input X0 is triggered from OFF to ON, interrupt subroutine executes DELAY instruction first, therefore the program after DELAY instruction (X1 = ON, Y0 = ON…) will be delayed for 1ms.



Points to note:

1. User can adjust the delay time according to the actual needs.
2. The delay time of DELAY instruction could be increased due to the execution of communication, high-speed counter and high-speed pulse output instructions.
3. The delay time of DELAY instruction could be increased due to the delay of transistor or relay when external output (transistor or relay) is specified.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
144	GPWM	S1 S2 D	General PWM output				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1													*			GPWM: 7 steps
S_2													*			
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

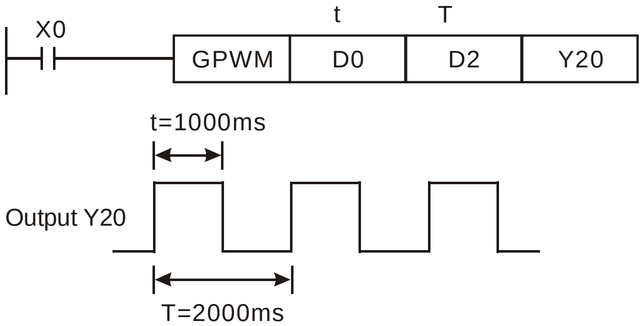
S_1: Width of output pulse **S_2**: Pulse output cycle (occupies 3 devices) **D**: Pulse output device

Explanations:

1. When GPWM instruction executes, pulse output will be executes on device specified by **D** according to pulse output width **S_1** and pulse output cycle **S_2**.
2. **S_1**: pulse output width. Range: t = 0~32,767ms.
3. **S_2**: pulse output cycle. Range: T = 1~32,767ms, **$S_1 \leqq S_2$**.
4. **S_2** +1 and **S_2** +2 are system-defined parameters, please don't use them.
5. **D:** pulse output device: Y, M and S.
6. When **S_1** $\leqq$ 0, no pulse output will be performed. When **S_1** $\geqq$ **S_2**, the pulse output device remains ON.
7. **S_1** and **S_2** can be modified when GPWM instruction is being executed

Program Example:

Assume D0 = K1000, D2 = K2000. When X0 = ON, Y20 will output pulses as the following diagram. When X0 = OFF, Y20 output will be OFF.



Points to note:

1. The instruction operates by the scan cycle; therefore the maximum error will be one PLC scan cycle. **S_1**, **S_2** and (**S_2 - S_1**) should be bigger than PLC scan cycle, otherwise malfunction will occur during GPWM outputs.
2. Please note that placing this instruction in a subroutine will cause inaccurate GPWM outputs.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2 SE	SX2
145	FTC	S_1 S_2 S_3 D	Fuzzy Temperature Control				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	FTC: 7 steps
S_1					*	*							*			
S_2					*	*							*			
S_3													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

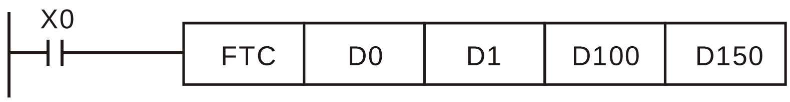
S_1: Set value (SV) **S_2**: Present value (PV) **S_3**: Parameter (sampling time) **D**: Output value (MV)

Explanations:

1. Range of **S_1** : 1 ~ 5000 (shown as 0.1°C ~ 500°C). Unit: 0.1°. If (**S_3** +1) is set as K0, the range will be 0.1°C ~ 500°C.
2. Settings of parameter **S_3** +1: bit0 = 0 ->°C; bit1 = 0 ->°F; bit1 = 0 -> no filter function; bit1 = 1 -> with filter function; bit2 ~ bit5 -> 4 kinds of heating environments; bit6 ~ bit15 -> reserved. See remarks for more information.
3. **D** is the value between 0 ~ sampling time × 100. When using this instruction, the user has to adopt other instructions according to the types of the heater. For example, FTC can be used with GPWM for output pulse control. “Sampling time × 100” is the cycle of GPWM pulse output; MV is the width of GPWM pulse. See program example 1.
4. There is no limit on the times of using FTC instruction, but Do not repeatedly use a designated operand in case an error may occur.
5. The models which are supported are DVP-ES2/EX2 v. 3.22, DVP-SA2/SX2 v. 2.66, and DVP-SE v. 1.60 (and above).

Program Example:

1. Set up the parameter before executing FTC instruction.
2. When X0 = On, the instruction will be executed and and result will be stored in D150. When X0 = Off, the instruction will not be executed and the previous data remain unchanged.

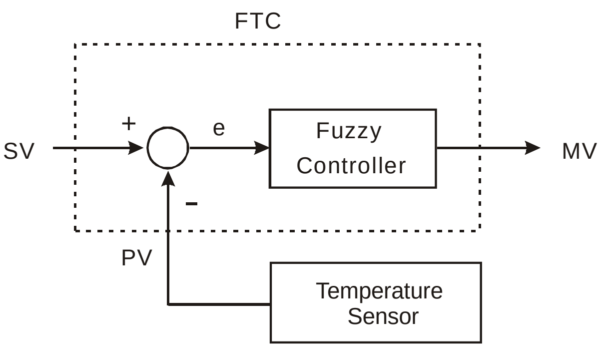


Remarks:

1. Setting of **S_3**:

Device No.	Function	Range	Explanation
S_3 :	Sampling time (T_S) (unit: 100ms)	1 ~ 200 (unit: 100ms)	If T_S is less than a scan time, PID instruction will be executed for a scan time. If T_S= 0, PID instruction will not be enabled. The minimum T_S must be greater than a scan time.
S_3 +1 :	b0: temperature unit b1: filter function b2 ~ b5: heating environnment b6 ~ b15: reserved	b0 =0 means ºC b0 =1 means ºF	When the value exceeds the upper bound, use upper bound.
		b1=0 means without fileter function b1=1 means with filter function	When without filter function, PV = currently measured value. When with filter function, PV = (currently measured value + previous PV)/2
		b2=1	Slow heating environment
		b3=1	General heating environment
		b4=1	Fast heating environment
		b5=1	High-speed heating environment
S_3 +2 : ≀ **S_3** +6 :	Parameters for system use only. Do not use them.		

2. Control diagram:



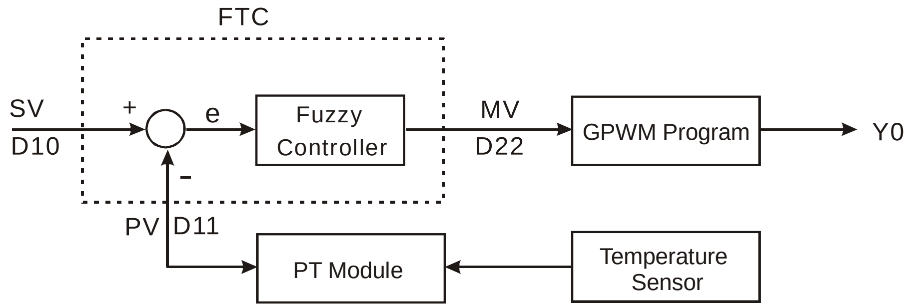
3. Notes and suggestion:

   It is recommended that the sampling time be set to 2 times more than the sampling time of the temperature sensor for better temperature control.

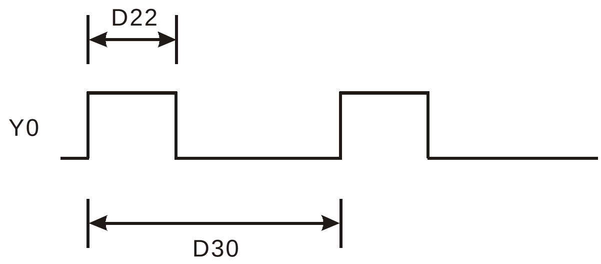
   bit2 ~ bit5 of **S_3**+1 are for the control speed. If the user does not set up the parameter, FTC will automatically activate “general heating environment”. When the user finds that the control is too slow to reach SV, select “slow heating environment” to enhance the speed to reach SV. On the contrary, when the user finds that the control is too fast or with too many fluctuations, select “fast heating environment” to slow down the control speed.

   When bit2 ~ bit5 of **S_3**+1 are all set as 1 or more than 1 environments are designated, FTC instruction will check from bit2 to bit 5 in order and enable the function that has been set as 1. The parameter can be modified during the control.

Example 1: control diagram

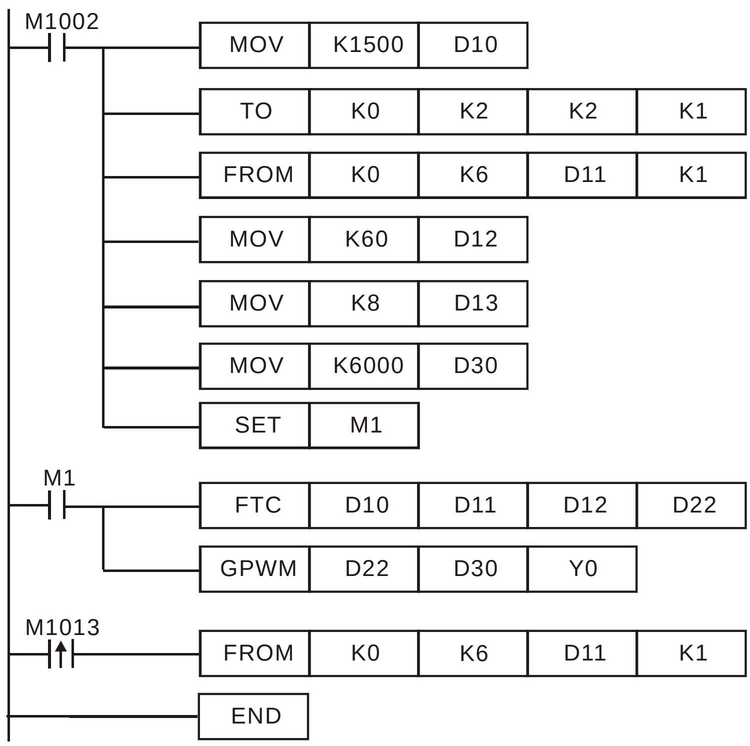


Output D22 (MV) of FTC instruction is the input D22 of GPWM instruction, as the duty cycle of adjustable pulses. D30 is the fixed cycle time of pulses. See below for the timing diagram of Y0 output.

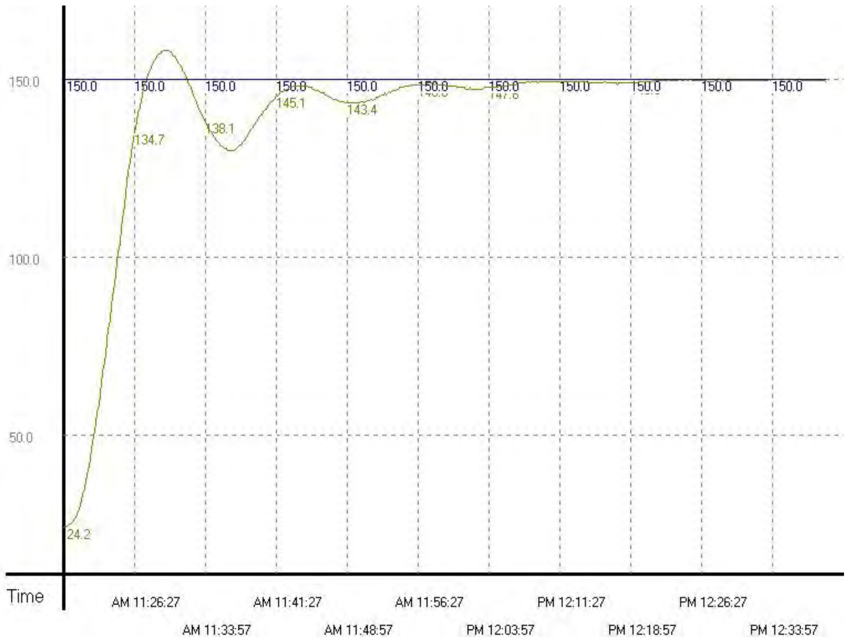


Assume parameter settings: D10 = K1,500 (target temperature), D12 = K60 (sampling time: 6 secs.), D13 = K8 (bit3=1), D30 = K6,000 (=D12*100)

The example control program is indicated as:

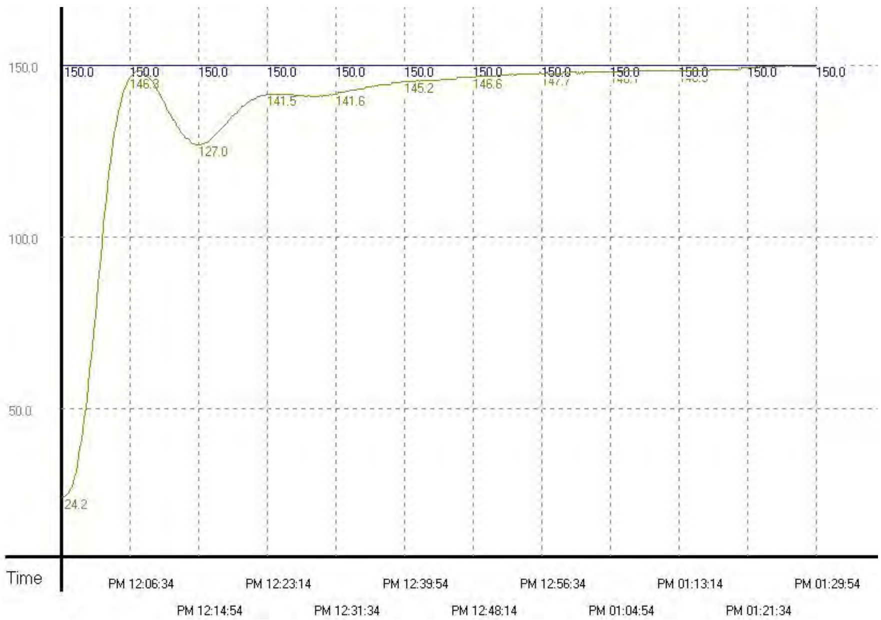


Experiment in an oven which can be heated up to 250°C. See below for the records of target and present temperatures. As shown in the diagram below, we can see that after 48 minutes, the temperature is able to reach the target temperature with ±1ºC inaccuracy and exceed approx. 10°C of the target temperature.

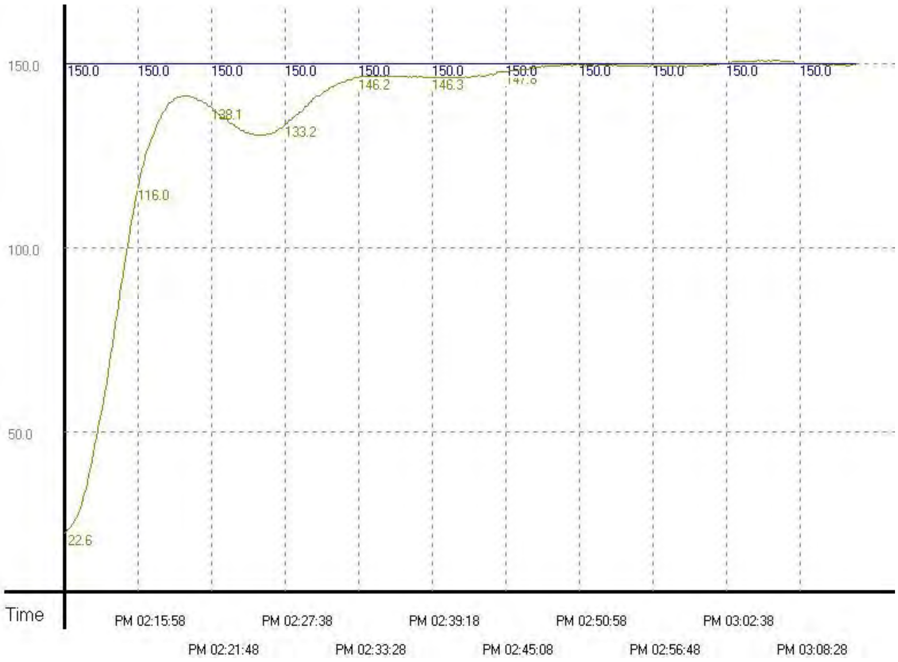


Example 2: Due to that the temperature once exceeds the target temperature, we modify the heating environment into “fast heating environment” (D13 = K16). The results are shown in the diagram below.

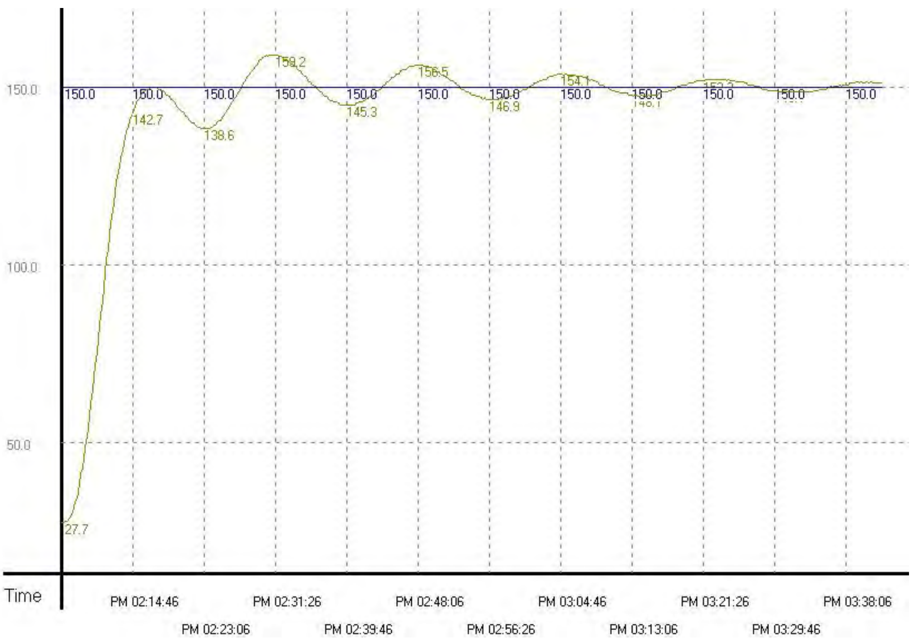
From the diagram below, we see that though the temperature no longer exceeds the target temperature, it still needs to take more than 1 hour and 15 minutes to reach the target temperature with ±1ºC inaccuracy. It seems that we have chosen the right environment, but the sampling time is too long, resulting in the extension of heating time.



Example 3: To speed up the speed to reach the target temperature, we correct the sampling time as 4 seconds (D12 = K40, D30 = K4,000). The results are shown in the diagram below.
From the diagram below, we see that the overall control time has been shortened as 37 minutes. Therefore, we find out that modifying the sampling time can speed up the time for reaching the target temperature.



Example 4: To see if we can reach the target temperature faster, we modify the sampling time frim example 3 into 2 seconds (D12 = K20, D30 = K2,000). The results are shown in the diagram below. From the diagram below, we see that the sampling time that is too short will cause the control system to become too sensitive and lead to up and down fluctuations.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
147	D	SWAP	P	S	Byte swap				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SWAP, SWAPP: 3 steps
S								*	*	*	*	*	*	*	*	DSWAP, DSWAPP: 5 steps

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

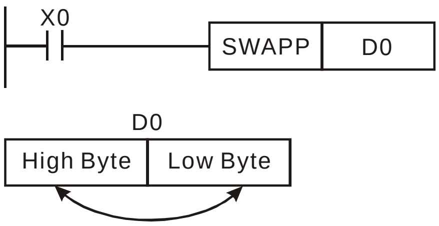
S: Device for byte swap.

Explanations:

1. For 16-bit instruction, high byte and low byte of the register will be swapped.
2. For 32-bit instruction, byte swap is conducted on the 2 registers separately.
3. This instruction adopts pulse execution instructions (SWAPP, DSWAPP)
4. If operand **D** uses device F, only 16-bit instruction is available

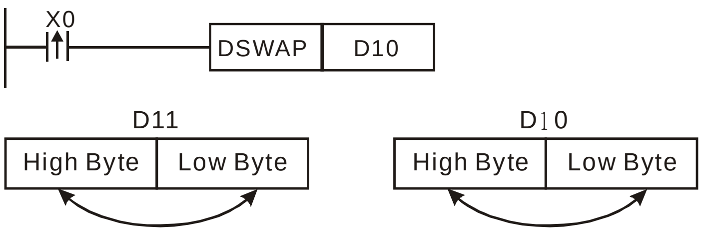
Program Example 1:

When X0 = ON, high byte and low byte of D0 will be swapped.



Program Example 2:

When X0 = ON, high byte and low byte of D11 will be swapped as well as the high byte and low byte of D10.



API	Mnemonic		Operands	Function	Controllers
148	MEMR	P	(m) (D) (n)	Reading the data from the file register	ES2/EX2, SS2, SA2, SX2, SE

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																7 steps
m					*	*							*			The 32-bit instruction and DVP-SS2 are not supported.
D													*			
n					*	*							*			

PULSE					16-bit					32-bit				
ES2/EX2	SS2	SA2	SX2	SE	ES2/EX2	SS2	SA2	SX2	SE	ES2/EX2	SS2	SA2	SX2	SE

Operands:

m: File register from which the data is read.

D: Initial data register where the data is stored (The data register is between D2000 and D9999.)

n: Number of data

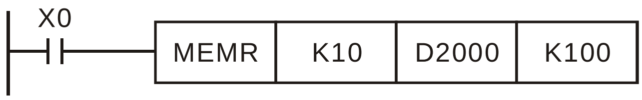
Explanations:

1. The 32-bit instruction is not supported.
2. If **m**, **D**, or **n** is not within the range, an operation error occurs, the instruction is not executed, M1067 and M1068 is ON, and the error code in D1067 is H’0E1A.
3. If no data is written into the file register, the default value which will be read from it is -1.
4. The file registers do not support M1101. If users want to read the data from the file register when the PLC runs, they can use LD M1002 and MEMR to read the data.
5. The range of supported operands **m** and **n**:

Models	Firmware version	Operand m	Operand n
SA2/SX2	V2.40~V3.00	K0 ~ K4999	K1~K5000
	V3.02 and later	K0 ~ K7999	K1~K8000
ES2/EX2	V2.80~3.44	K0 ~ K4999	K1~K5000
	V3.46 and later	K0 ~ K7999	K1~K8000
ES2-E Should work with a memory card	V1.08 adn later	K0 ~ K7999	K1~K8000
SS2/SE	NOT supported		
ES2-C	NOT supported		

Program Example:

1. Use MEMR to read the data from the 100 file registers starting from the tenth file register to the data registers starting from D2000.
2. When X0 is ON, the instruction is executed. When X0 becomes OFF, the instruction is not executed, and the data which is read previous is unchanged.



API	Mnemonic			Operands	Function	Controllers				
149		MEMW	P	S m n	Writing the data into the file register	ES2/ EX2	SS2	SA2	SX2	SE

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											7 steps
S													*			The 32-bit instruction and DVP-SS2 are not supported.
m					*	*							*			
n					*	*							*			

PULSE					16-bit					32-bit				
ES2/ EX2	SS2	SA2	SX2	SE	ES2/ EX2	SS2	SA2	SX2	SE	ES2/ EX2	SS2	SA2	SX2	SE

Operands:

S: Initial source device (The data register is between D2000 and D9999.)

m: File register into which the data is written.

n: Number of data (The number of data is between K1 and K100.)

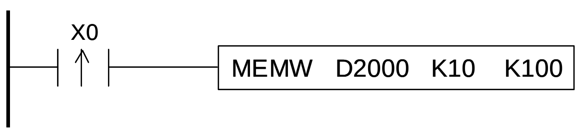
Explanations:

1. The 32-bit instruction is not supported.
2. If **m**, **D**, or **n** is not within the range, an operation error occurs, the instruction is not executed, M1067 and M1068 is ON, and the error code in D1067 is H'0E1A.
3. Owing to the fact that the file registers take flash ROM as the memories, 100 words at most can be written into the file registers, and only when the conditional contact turns from OFF to ON can the data be written into the file registers once. Note: The data only can be written into the file registers 100,000 times. Please use them with care.
4. The range of supported operands **m** and **n**:

Models	Firmware version	Operand m	Operand n
SA2/SX2	V2.40~V3.00	K0 ~ K4999	K1~K100
	V3.02 and later	K0 ~ K7999	K1~K100
ES2/EX2	V2.80~3.44	K0 ~ K4999	K1~K100
	V3.46 and later	K0 ~ K7999	K1~K100
ES2-E Should work with a memory card	V1.08 and later	K0 ~ K7999	K1~K100
SS2/SE	NOT supported		
ES2-C	NOT supported		

Program Example:

1. Use MEMW to write the data from the 100 data registers starting from D2000 to the file registers starting from the tenth file register.
2. When X0 turns from OFF to ON, the instruction is executed once.



API	Mnemonic			Operands	Function	Controllers			
154	D	RAND	P	S1 S2 D	Random number	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	RAND, RANDP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DRAND, DRANDP: 13
S_2					*	*	*	*	*	*	*	*	*	*	*	steps
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

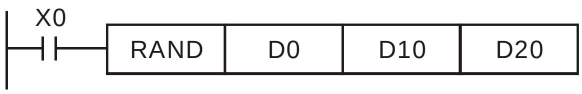
S_1: Lower bound of the random number **S_2**: Upper bound of the random number **D**: Operation result

Explanations:

1. The range of 16-bit operands **S_1**, **S_2**: K0≦**S_1** , **S_2**≦K32,767; the range of 32-bit operands **S_1**, **S_2**: K0≦**S_1** , **S_2**≦K2,147,483,647.
2. Entering **S_1** > **S_2** will result in operation error. The instruction will not be executed at this time, M1067, M1068 = ON and D1067 records the error code 0E1A (HEX)

Program Example:

When X10 = ON, RAND will produce the random number between the lower bound D0 and upper bound D10 and store the result in D20.



API	Mnemonic			Operands	Function	Controllers
168	D	MVM	P	S1 S2 D	Transfer Designated Bits	ES2/EX2/EC5 SS2 SA2/SE SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1							*	*	*	*	*	*	*	*	*	MVM, MVMP: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DMVM,DMVMP:
D							*	*	*	*	*	*	*	*	*	13 steps

PULSE				16-bit				32-bit			
ES2/EX2/EC5	SS2	SA2/SE	SX2	ES2/EX2/EC5	SS2	SA2/SE	SX2	ES2/EX2/EC5	SS2	SA2/SE	SX2

Operands:

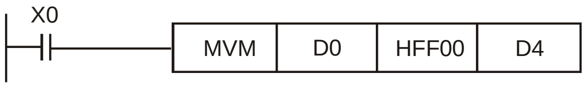
S_1: Source device 1 S_2: Bits to be masked (OFF) **D**: **D** =(S_1 & S_2) | (**D** & ~ S_2)

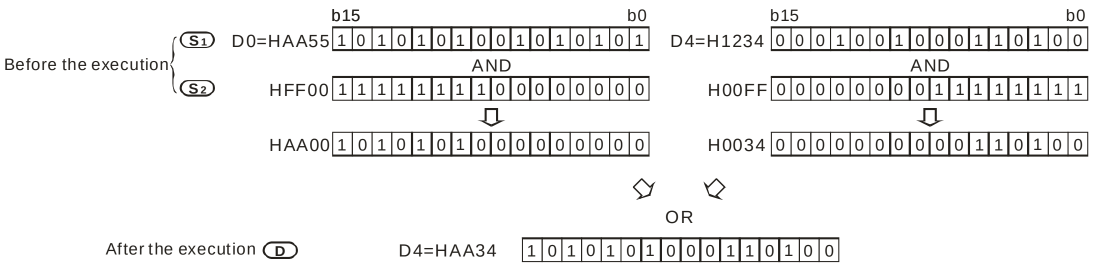
Explanations:

1. The instruction conducts logical AND operation between S_1 and S_2 first, logical AND operation between **D** and ~S_2 secondly, and combines the 1st and 2nd results in **D** by logical OR operation.
2. Rule of Logical AND operation: 0 AND 1 = 0, 1 AND 0 = 0, 0 AND 0 = 0, 1 AND 1 = 1
3. Rule of Logical OR operation: 0 OR 1= 1, 1 OR 0 = 1, 0 OR 0 = 0, 1 OR 1 = 1.

Program Example 1 :

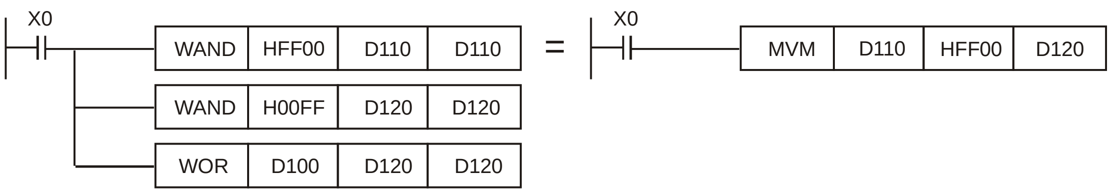
When X0 = ON, MVM instruction conducts logical AND operation between 16-bit register D0 and H'FF00 first, logical AND operation between D4 and H'00FF secondly, and combines the 1st and 2nd results in D4 by logical OR operation.





Program Example 2 :

Simplify instructions:



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
176	MMOV	P	(S) (D)	16-bit→32-bit Conversion				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MMOV, MMOVP: 5 steps
S					*	*	*	*	*	*	*	*	*			
D											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

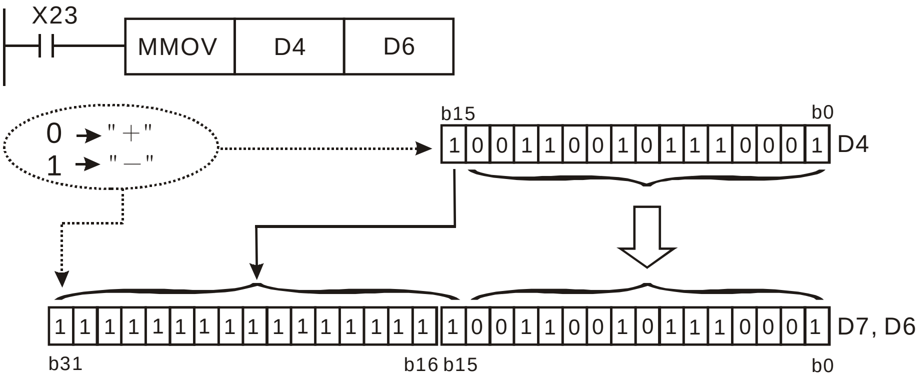
S: Source device (16-bit) **D:** Destination device (32-bit)

Explanations:

1. MMOV instruction sends the data in 16-bit device **S** to 32-bit device **D**. Sign bit (MSB) of source device will be copied to every bit in the high byte of **D**.

Program example:

When X23 = 0N, 16-bit data in D4 will be sent to D6 and D7.



In the example above, b15 in D4 will be sent to b15~b31 of D7/D6, therefore all bits in b15~b31 will be “negative.”

API	Mnemonic	Operands	Function	Controllers				
				ES2 EX2	SS2	SA2	SX2	SE
177	GPS	(S) (D)	GPS data receiving					

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	GPS: 5 steps
S					*	*							*			
D													*			

PULSE					16-bit					32-bit				
ES2 EX2	SS2	SA2	SX2	SE	ES2 EX2	SS2	SA2	SX2	SE	ES2 EX2	SS2	SA2	SX2	SE

Operands:

S: Sentence identifier for GPS data receiving **D**: Destination device for feedback data

Explanations:

1. GPS data receiving instruction is only applicable on COM1 (RS-232), with communication format: 9600,8,N,1, protocol: NMEA-0183, and communication frequency: 1Hz.
2. Operand **S** is sentence identifier for GPS data receiving. K0: $GPGGA, K1: $GPRMC.
3. Operand **D** stores the received data. Up to 17 consecutive words will be occupied and can not be used repeatedly. Please refer to the table below for the explanations of each **D** device.

- When **S** is set as K0, sentence identifier $GPGGA is specified. **D** devices refer to:

No.	Content	Range	Format	Note
D *+ 0*	*Hour*	*0 ~ 23*	*Word*	
D *+ 1*	*Minute*	*0 ~ 59*	*Word*	
D *+ 2*	*Second*	*0 ~ 59*	*Word*	
D *+ 3~4*	*Latitude*	*0 ~ 90*	*Float*	*Unit: dd.mmmmmm*
D *+ 5*	*North / South*	*0 or 1*	*Word*	*0(+)→North, 1(-)→South*
D *+ 6~7*	*Longitude*	*0 ~ 180*	*Float*	*Unit: ddd.mmmmmm*
D *+ 8*	*East / West*	*0 or 1*	*Word*	*0(+)→East, 1(-)→West*
D *+ 9*	*GPS data valid / invalid*	*0, 1, 2*	*Word*	*0 = invalid*
D *+ 10~11*	*Altitude*	*0 ~9999.9*	*Float*	*Unit: meter*
D + 12~13	*Latitude*	*-90 ~ 90*	*Float*	*Unit: ±dd.ddddd*
D + 14~15	*Longitude*	*-180 ~ 180*	*Float*	*Unit: ±ddd.ddddd*

- When **S** is set as K1, sentence identifier $GPRMC is specified. **D** devices refer to:

No.	Content	Range	Format	Note
D *+ 0*	*Hour*	*0 ~ 23*	*Word*	
D *+ 1*	*Minute*	*0 ~ 59*	*Word*	
D *+ 2*	*Second*	*0 ~ 59*	*Word*	
D *+ 3~4*	*Latitude*	*0 ~ 90*	*Float*	*Unit: dd.mmmmmm*
D *+ 5*	*North / South*	*0 or 1*	*Word*	*0(+)→North, 1(-)→South*
D *+ 6~7*	*Longitude*	*0 ~ 180*	*Float*	*Unit: ddd.mmmmmm*
D *+ 8*	*East / West*	*0 or 1*	*Word*	*0(+)→East, 1(-)→West*
D *+ 9*	*GPS data valid /*	*0, 1, 2*	*Word*	*0 = invalid*

No.	Content	Range	Format	Note
	invalid			
D *+ 10*	*Day*	*1 ~ 31*	*Word*	
D *+ 11*	*Month*	*1 ~ 12*	*Word*	
D *+ 12*	*Year*	*2000 ~*	*Word*	
D + 13~14	*Latitude*	*-90 ~ 90*	*Float*	*Unit: ±dd.ddddd*
D + 15~16	*Longitude*	*-180 ~ 180*	*Float*	*Unit: ±ddd.ddddd*

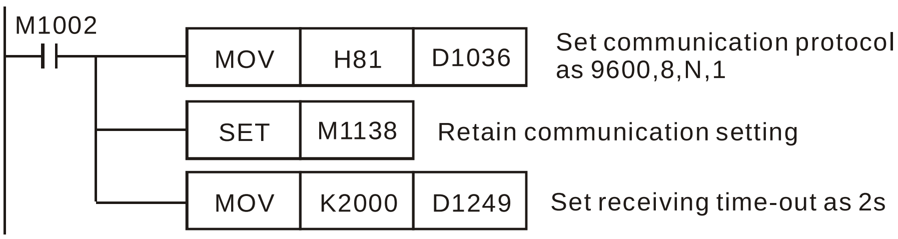
4. When applying GPS instruction, COM1 has to be applied in Master mode, i.e. M1312 has to be enabled to sending request. In addition, M1314 = ON indicates receiving completed. M1315 = ON indicates receiving error. (D1250 = K1, receiving time-out; D1250 = K2, checksum error)
5. Associated M flags and special D registers:

No.	Function
M1312	COM1 (RS-232) sending request
M1313	COM1 (RS-232) ready for data receiving
M1314	COM1 (RS-232) data receiving completed
M1315	COM1 (RS-232) data receiving error
M1138	Retaining communication setting of COM1
D1036	COM1 (RS-232) Communication protocol
D1249	COM1 (RS-232) data receiving time-out setting. (Suggested value: >1s)
D1250	COM1 (RS-232) communication error code

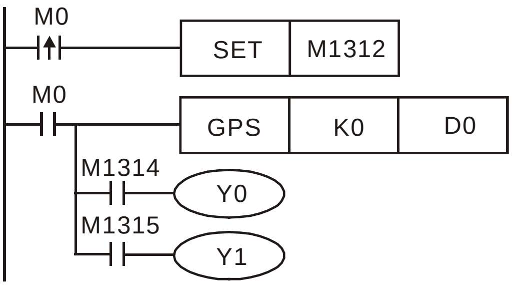
6. Before applying the received GPS data, please check the value in **D**+9. If **D**+9 = 0, the GPS data is invalid.
7. If data receiving error occurs, the previous data in **D** registers will not be cleared, i.e. the previous received data remains intact.

Program example: Sentence identifier: $GPGGA

1. Set COM1communication protocol first



2. Then enable M0 to execute GPS instruction with sentence identifier $GPGGA

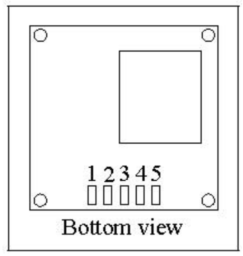


3. When receiving completed, M1314 = ON. When receiving failed, M1315 = ON. The received data will be stored in devices starting with D0.

No.	*Content*	*No.*	*Content*
D0	*Hour*	*D8*	*East / West*
D1	*Minute*	*D9*	*GPS data valid / invalid*
D2	*Second*	*D10~D11*	*Altitude*
D3~D4	*Latitude*	*D12~D13*	*Latitude. Unit: ±dd.ddddd*
D5	*North / South*	*D14~D15*	*Longitude. Unit: ±ddd.ddddd*
D6~D7	*Longitude*		

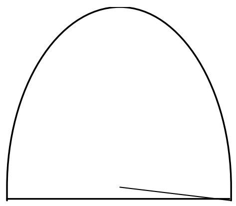
4. Pin number description on GPS module (LS20022)

Pin No. of GPS	1	2	3	4	5
Definition	VCC(+5V)	Rx	Tx	GND	GND



5. Pin number description on PLC COM1:

Pin No. of COM1	1	2	3	4	5	6	7	8
Definition	VCC(+5V)		--	Rx	Tx	--	--	GND



API	Mnemonic		Operands	Function	Controllers				
178	D	SPA	(S) (D)	Solar Panel Positioning	ES2/ EX2	SS2	SA2	SX2	SE

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DSPA: 9 steps
S					*	*							*			
D													*			

PULSE					16-bit					32-bit				
ES2/ EX2	SS2	SA2	SX2	SE	ES2/ EX2	SS2	SA2	SX2	SE	ES2/ EX2	SS2	SA2	SX2	SE

Operands:

S: Start device for input parameters **D**: Start device for output parameters

Explanations:

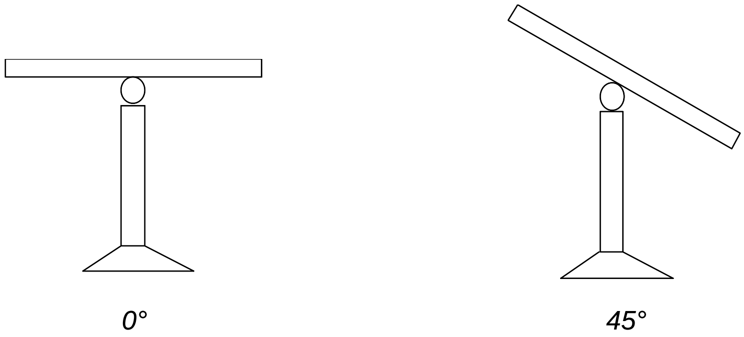
1. This instruction is a function provided for free. It is for non-commercial use only. If users want to use the instruction for a commercial purpose, they have to obtain permission from related organizations before they sell equipment.
2. Operand **S** occupies 208 consecutive word registers. The function of each device is as below:

No.	*Content*	*Range*	*Format*	*Note*
***S** + 0*	*Year*	*2000 ~*	*Word*	*Please enter the correct time of the local longitude. Please refer to DTM (parameter 11) for the conversion formula. A simple illustration is as in point 6.*
***S** + 1*	*Month*	*1 ~ 12*	*Word*	
***S** + 2*	*Day*	*1 ~ 31*	*Word*	
***S** + 3*	*Hour*	*0 ~ 23*	*Word*	
***S** + 4*	*Minute*	*0 ~ 59*	*Word*	
***S** + 5*	*Second*	*0 ~ 59*	*Word*	
***S** + 6~7*	*Time difference (Δt) (sec)*	*± 8000*	*Float*	
***S** + 8~9*	*Local time zone*	*± 12*	*Float*	*West: negative*
***S** + 10~11*	*Longitude*	*± 180*	*Float*	*West: negative Unit: degree*
***S** + 12~13*	*Latitude*	*± 90*	*Float*	*South: negative Unit: degree*
***S** + 14~15*	*Elevation*	*0~ 6500000*	*Float*	*Unit: meter*
***S** + 16~17*	*Pressure*	*0 ~ 5000*	*Float*	*Unit: millibar*
***S** + 18~19*	*Mean annual temperature (MAT)*	*-273~6000*	*Float*	*Unit: °C*
***S** + 20~21*	*Slope*	*± 360*	*Float*	
***S** + 22~23*	*Azimuth*	*± 360*	*Float*	
***S** + 24~25*	*Atmospheric refraction between sunrise and sunset*	*± 5*	*Float*	
***S** +26~207*	*Reserved for system operation*			

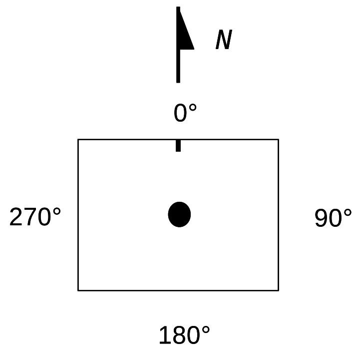
3. Operand **D** occupies 8 consecutive word registers. The function of each device is as below:

No.	Content	Range	Format	Note
D + 0~1	*Zenith*	*0 ~ 90*	*Float*	*Horizontal=0*
D + 2~3	*Azimuth*	*0 ~ 360*	*Float*	*North point=0*
D + 4~5	*Incidence*	*0 ~ 90*	*Float*	
D + 6	*Converted DA value of Zenith*	*0 ~ 2000*	*Word*	*1LSB = 0.045 degree*
D + 7	*Converted DA value of Azimuth*	*0 ~ 2000*	*Word*	*1LSB = 0.18 degree*

4. The execution time of SPA instruction costs up to 50ms, therefore we suggest users to execute this instruction with an interval not less than 1 sec, preventing the instruction from taking too much PLC operation time.
5. Definition of Zenith: 0° and 45°.



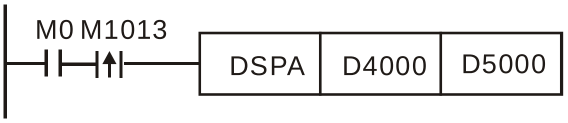
6. Definition of Azimuth:



7. The correct time of the local longitude: If we suppose that it is AM8:00:00 in Taipei, and the longitude is 121.55 degrees east, then the correct time of the local longitude in Taipei should be AM8:06:12. Please refer to API168 DTM instruction (parameter k11) for more explanation.

Program example:

1. Input parameters starting from D4000: 2009/3/23/(y/m/d),10:10:30, Δt = 0, Local time zone = +8, Longitude/Latitude = +119.192345 East, +24.593456 North, Elevation = 132.2M, Pressure = 820m, MAT = 15.0℃, Slope = 30 degree, Azimuth = -10 degree.



2. Output results: D5000: Zenith = F37.2394 degree; D5002: Azimuth = F124.7042 degree.

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
179	D	WSUM	P	S D n	Sum of multiple devices				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	WSUM, WSUMP: 7 steps
S											*	*	*			DWSUM, DWSUMP: 13
n					*	*							*			steps
D											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

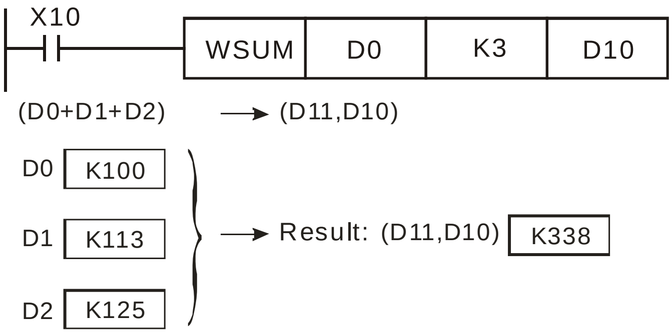
S: Source device **n**: Data length to be summed up **D**: Device for storing the result

Explanations:

1. WSUM instruction sums up **n** devices starting from **S** and store the result in **D.**
2. If the specified source devices **S** are out of valid range, only the devices in valid range will be processed.
3. Valid range for **n**: 1~64. If the specified **n** value is out of the available range (1~64), PLC will take the upper (64) or lower (1) bound value as the set value.
4. **D** used in the 16-bit/32-bit instruction is a 32-bit register.

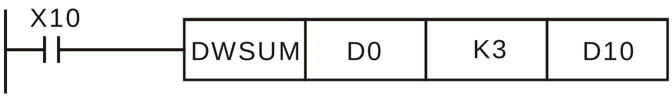
Program example 1:

When X10 = ON, the 3 consecutive devices (**n** = 3) from D0 will be summed up and the result will be stored in (D11, D10)



Program example 2:

When X10 = ON, 3 consecutive devices (**n** = 3) from (D1, D0) will be summed up and the result will be stored in (D11, D10).



(D1,D0)+(D3,D2)+(D5,D4) → (D11,D10)

(D1,D0) K100

(D3,D2) K113

(D5,D4) K125

→ Result: (D11,D10) K338

API	Mnemonic		Operands	Function	Controllers			
202	SCAL	P	(S1) (S2) (S3) (D)	Proportional calculation	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																SCAL,SCLAP: 9 steps
S_1					*	*							*			
S_2					*	*							*			
S_3					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source value **S_2**: Slope (unit: 0.001) **S_3**: Offset **D**: Operation result

Range of operands **S_1, S_2, S_3**: -32768~32767.

Explanations:

SCAL instruction performs a proportional calculation according to the internal slope equation.

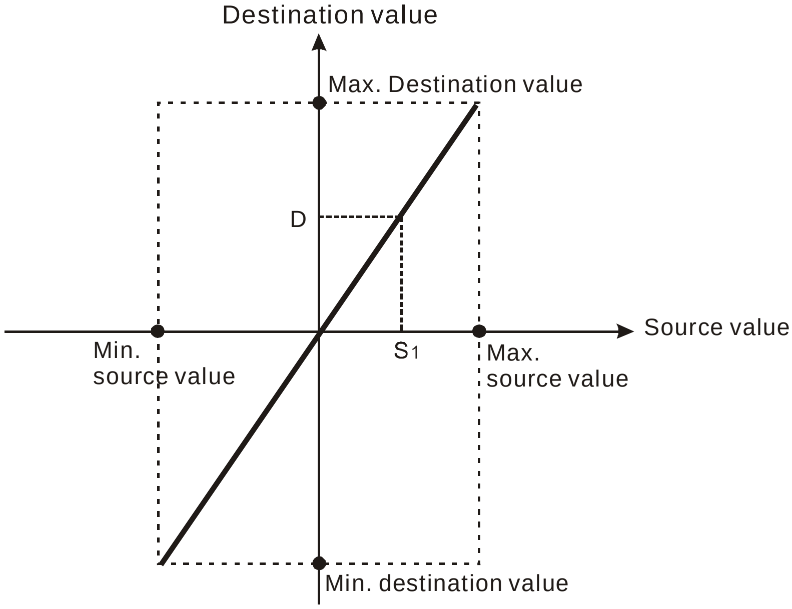
Operation equation in the instruction: **D** = (**S_1** × **S_2**) ÷ 1000 + **S_3**

Users have to obtain **S_2** and **S_3** (decimals are rounded up into 16-bit integers) by using the slope and offset equations below.

Slope equation: **S_2** = [(max. destination value – min. destination value) ÷ (max. source value – min. source value)] × 1,000

Offset equation: **S_3** = min. destination value – min. source value × **S_2** ÷ 1,000

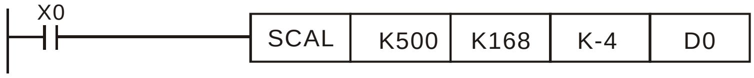
The output curve is shown as the figure:

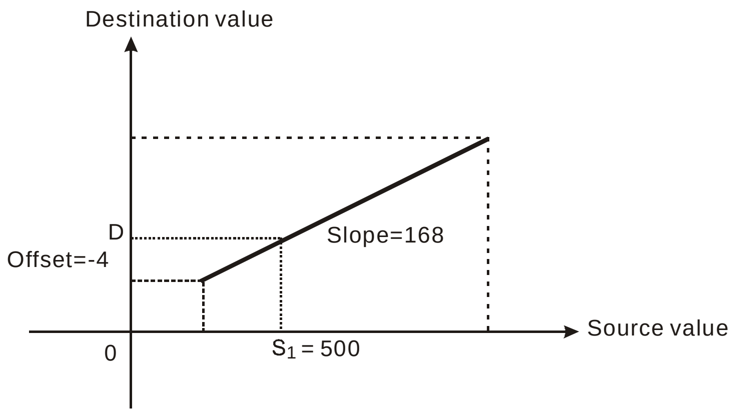


Program Example 1:

1. Assume S_1 = 500, S_2 = 168 and S_3 = -4. When X0 = ON, SCAL instruction executes and the result of proportional calculation will be stored in D0.

Equation: D0 = (500 × 168) ÷ 1000 + (-4) = 80

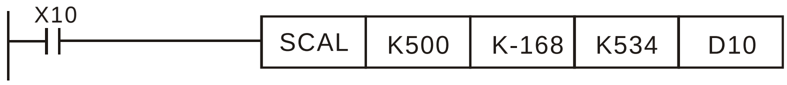


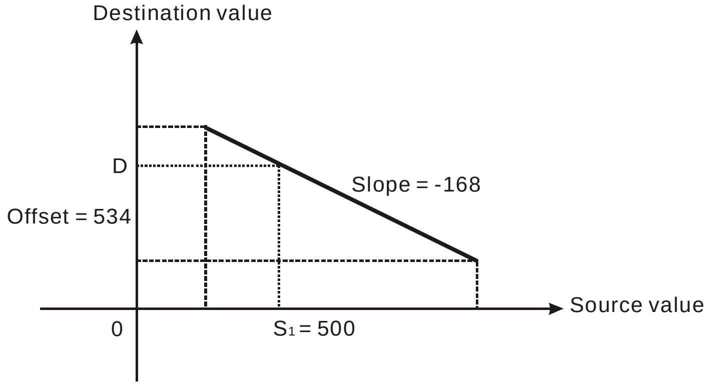


Program Example 2:

1. Assume S_1 = 500, S_2 = -168 and S_3 = 534. When X0 = ON, SCAL instruction executes and the result of proportional calculation will be stored in D10..

Equation: D10 = (500 × -168) ÷ 1000+ 534 = 450





Points to note:

1. This instruction is applicable for known slope and offset. If slope and offset are unknown, please use SCLP instruction for the calculation.
2. S_2 has to be within the range -32,768 ~ 32,767. If S_2 exceeds the applicable range, use SCLP instruction instead.
3. When adopting the slope equation, the max source value must be larger than min source value, but the max destination value does not need to be larger than min destination value.
4. If **D** > 32,767, **D** will be set as 32,767. If **D** < -32,768, **D** will be set as -32,768.

API	Mnemonic			Operands	Function	Controllers			
203	D	SCLP	P	S1 S2 D	Parameter proportional calculation	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices X	Y	M	S	Word devices K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*							*			SCLP, SCLPP: 7 steps
S_2													*			DSCLP, DSCLPP: 13
D													*			steps

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Source value **S_2**: Parameters **D**: Operation result

Explanations:

1. SCLP instruction performs a proportional calculation according to the internal slope equation as well as the parameters set in this instruction.
2. Settings of **S_2** for 16-bit instruction (occupies 4 consecutive devices):

Device No.	Parameter	Range
S_2	Max. source value	-32768~32767
S_2+1	Min. source value	-32768~32767
S_2+2	Max. destination value	-32768~32767
S_2+3	Min. destination value	-32768~32767

3. Settings of **S_2** for 32-bit instruction (occupies 8 consecutive devices).

Device No.	Parameter	Range	
		Integer	Floating point number
S_2 、**S_2**+1	Max. source value	-2,147,483,648~2,147,483,647	Range of 32-bit floating point number
S_2+2 、3	Min. source value		
S_2+4 、5	Max. destination value		
S_2+6 、7	Min. destination value		

4. Operation equation in the instruction: **D** = [(**S_1** – min. source value) × (max. destination value – min. destination value)] ÷ (max. source value – min. source value) + min. destination value
5. The equation to obtain the operation equation of the instruction:

   y = kx + b

   where

   y = Destination value (**D**)

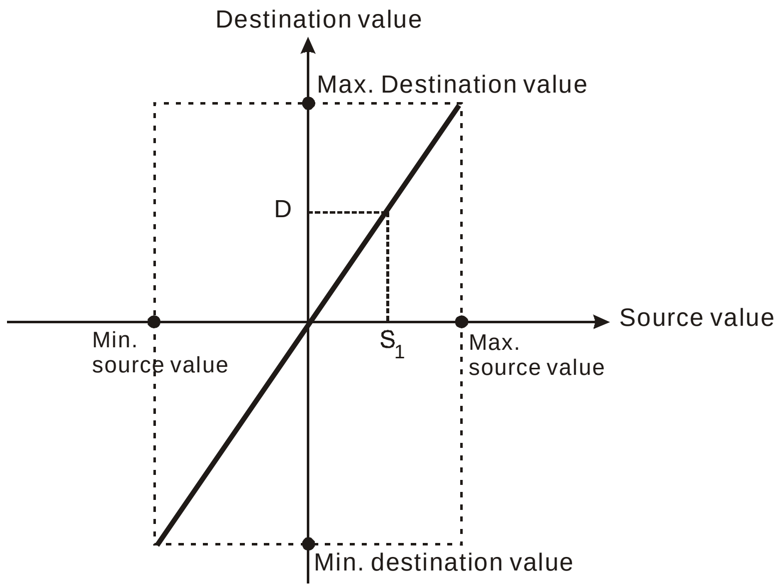
   k = Slope = (max. destination value – min. destination value) ÷ (max. source value – min. source value)

   x = Source value (**S_1**)

   b = Offset = Min. destination value – Min. source value × slope
6. Substitute the above parameters into **y = kx + b** and the operation instruction can be obtained. y = kx + b = D = k **S_1** + b = slope × **S_1** + offset = slope × **S_1** + min. destination value – min. source value × slope = slope × (**S_1** – min. source value) + min. destination value = (**S_1** – min. source

value) × (max. destination value – min. destination value) ÷ (max. source value – min. source value) + min. destination value

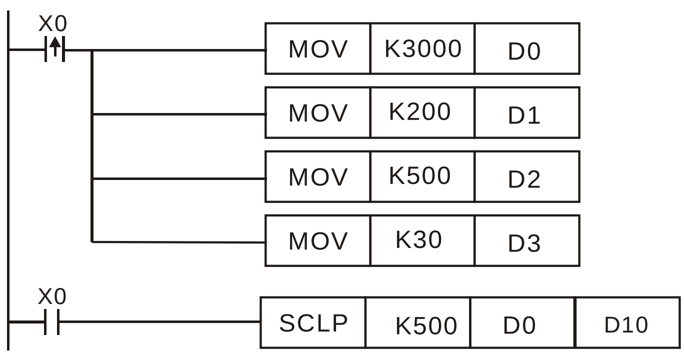
7. If **S_1** > max. source value, **S_1** will be set as max. source value. If **S_1** < min. source value, **S_1** will be set as min. source value. When the source value and parameters are set, the following output figure can be obtained:

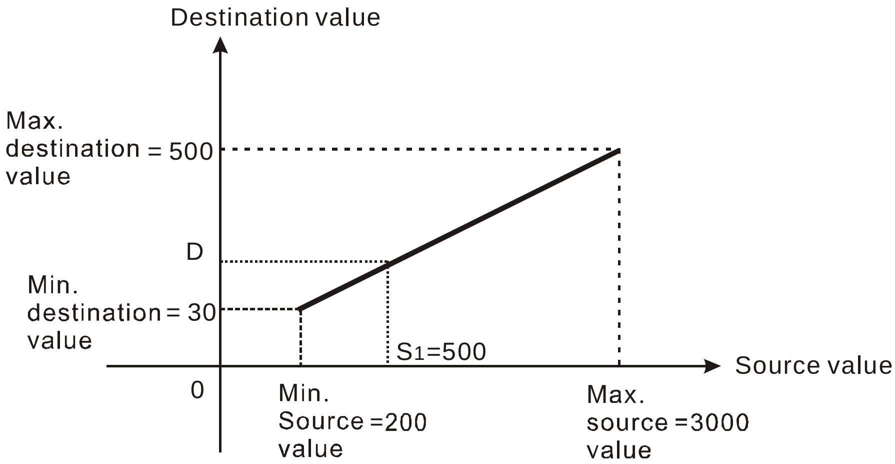


Program Example 1:

1. Assume source value **S_1** = 500, max. source value D0 = 3000, min. source value D1 = 200, max. destination value D2 = 500, and min. destination value D3 = 30. When X0 = ON, SCLP instruction executes and the result of proportional calculation will be stored in D10.

Equation: D10 = [(500 – 200) × (500 – 30)] ÷ (3000 – 200) + 30 = 80.35. Rounding off the result into an integer, D10 =80.

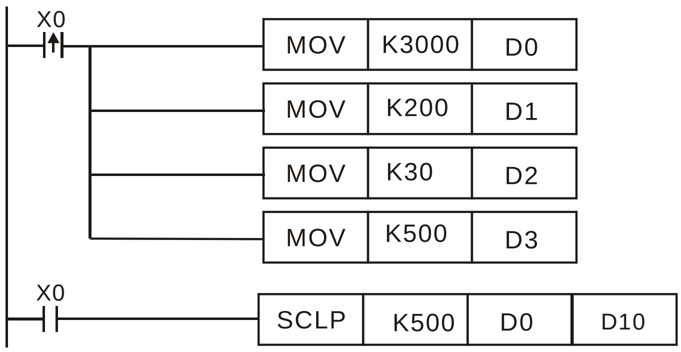


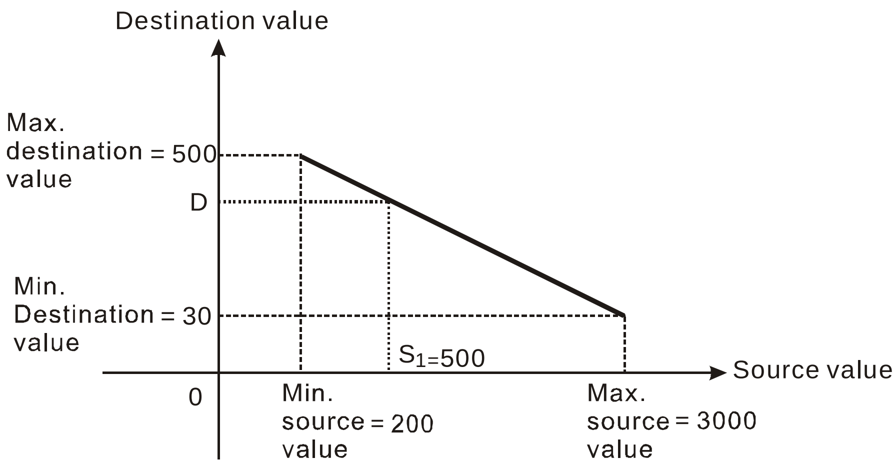


Program Example 2:

1. Assume source value S_1 = 500, max. source value D0 = 3000, min. source value D1 = 200, max. destination value D2 = 30, and min. destination value D3 = 500. When X0 = ON, SCLP instruction executes and the result of proportional calculation will be stored in D10.

Equation: D10 = [(500 – 200) × (30 – 500)] ÷(3000 – 200) + 500 = 449.64. Rounding off the result into an integer, D10 = 450.

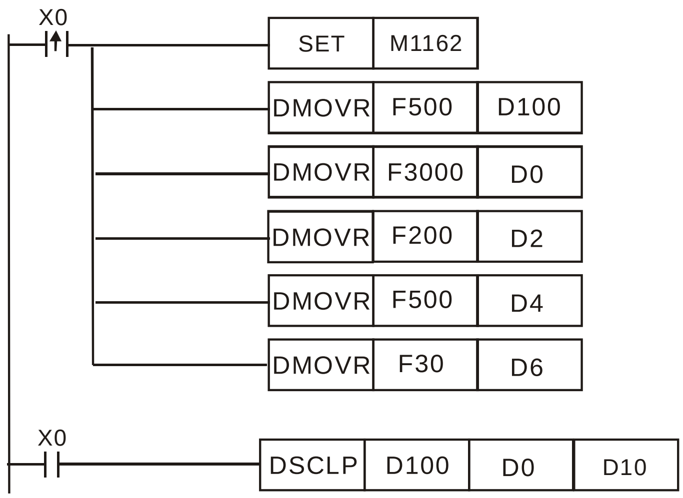




Program Example 3:

1. Assume the source value **S_1**, D100 = F500, max. source value D0 = F3000, min. source value D2 = F200, max. destination value D4 = F500, and min. destination value D6 = F30. When X0 = ON, M1162 is set up to adopt floating point operation. DSCLP instruction executes and the result of proportional calculation will be stored in D10.

Equation: D10 = [(F500 – F200) × (F500 – F30)] ÷ (F3000 – F200) + F30 = F80.35. Round off the result into an integer, D10 = F80.



Points to note:

1. Range of **S_1** for 16-bit instruction: max. source value ≥ **S_1** ≥ min. source value; -32,768 ~ 32,767. If the value exceeds the bounds, the bound value will be used for calculation.

2. Range of integer **S_1** for 32-bit instruction: max. source value ≥ **S_1** ≥ min. source value; -2,147,483,648 ~ 2,147,483,647. If the value exceeds the bounds, the bound value will be used for calculation.

3. Range of floating point **S_1** for 32-bit instruction: max. source value ≥ **S_1** ≥ min. source value; adopting the range of 32-bit floating point. If the value exceeds the bounds, the bound value will be used for calculation.

4. When adopting the slope equation, please note that the Max. source value must be larger than the min. source value. However the max. destination value does not need to be larger than the min. destination value.

API	Mnemonic			Operands	Function	Controllers			
205	D	CMPT	P	(S1) (S2) (n) (D)	Compare table	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	CMPT: 9 steps
S_1											*	*	*			CMPTP: 9 steps
S_2											*	*	*			DCMPT: 17 steps
n					*	*							*			DCMPTP: 17 steps
D								*	*	*	*	*	*			

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2 **n**: Data length (**n** = 1~16) **D**: Destination device

Explanations:

1. **S_1** and **S_2** can be T/C/D devices, for C devices only 16-bit devices are applicable (C0~C199).
2. The value in the high 16 bits of **n** used in the 32-bit instruction is an invalid value.
3. The value in the low 8 bits of **n** indicates the data length. For the 16-bit instruction, **n** is between 1 and 16. For the 32-bit instruction, **n** is between 1 and 32. If **n** is less than 1, it is count as 1. If **n** is larger than the maximum value, it is count as the maximum value.
4. The 16-bit data is written into **D**. If the data length is less than 16 bits, the bit which does not have a corresponding value is 0. For example, if **n** is K8, bit0~7 have corresponding values, and bit8~15 are 0.
5. The 32-bit instruction supports DVP-ES2/EX2 version 3.0 and above, DVP-SS2 version 2.8 and above, DVP-SA2 version 2.6 and above, DVP-SX2 version 2.4 and above, and DVP-SE.
6. The value in the high 8 bits of **n** indicates the comparison condition. The relation between the comparison conditions and the values are shown in the following table.

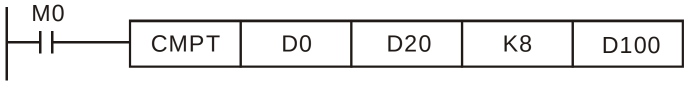
Value	K0	K1	K2	K3	K4
Comparison condition	**S_1** = **S_2**	**S_1** < **S_2**	**S_1** <= **S_2**	**S_1** > **S_2**	**S_1** >= **S_2**

7. The example of setting **n**: If **n** used in the 16-bit instruction is H0108, eight pieces of data are compared with eight pieces of data in terms of "larger than". If **n** used in the 32-bit instruction is H00000320, 32 pieces of data are compared with 32 pieces of data in terms of "less than".
8. If the setting value of the comparison condition exceeds the range, or the firmware version does not support the comparison condition, the default comparion condition "equal to" is executed. DVP-ES2/EX2 version 3.0and above, DVP-SS2 version 2.8 and above, DVP-SA2 version 2.6 and above, DVP-SX2 version 2.4 and above, and DVP-SE support the setting of the comparison condition.

9. The 16-bit comparison values used in the 16-bit instruction are signed values. The comparison values used in the 32-bit instruction are 32-bit signed values (M1162=OFF), or floating-point numbers (M1162=ON).
10. The 16-bit data or 32-bit data is written into **D**. If the data length is less than 16 bits or 32 bits, the bit which does not have a corresponding value is 0. For example, if **n** is K8, bit0~7 have corresponding values, and bit8~bit15 or bit8~bit31 are 0.
11. If the comparison result meets the comparison condition, the corresponding bit is 1. If the comparison result does not meet the comparison condition, the corresponding bit is 0.

Program example:

When M0 = ON, compare the 16-bit value in D0~D7 with D20~D27 and store the results in D100.



- Content in D0~D7:

No.	D0	D1	D2	D3	D4	D5	D6	D7
Value	K10	K20	K30	K40	K50	K60	K70	K80

- Content in D20~D27:

No.	D20	D21	D22	D23	D24	D25	D26	D27
Value	K12	K20	K33	K44	K50	K66	K70	K88

- After the comparison of CMPT instruction, the associated bit will be 1 if two devices have the same value, and other bits will all be 0. Therefore the results in D100 will be as below:

D100	Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8~15
	0	1	0	0	1	0	1	0	0…0
	H0052 (K82)								

API	Mnemonic	Operands	Function	Controllers				
207	CSFO	S S1 D	Catch speed and proportional output	ES2 EX2	SS2	SA2	SX2	SE

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	CSFO: 7 steps
S	*															
S_1													*			
D													*			

PULSE				16-bit					32-bit			
ES2/EX2	SS2	SA2/SE	SX2	ES2/EX2	SS2	SA2	SX2	SE	ES2/EX2	SS2	SA2/SE	SX2

Operands:

S: Source device of signal input (Only X0~X3 are available)   **S_1**: Sample time setting and the input speed information   **D**: Output proportion setting and output speed information

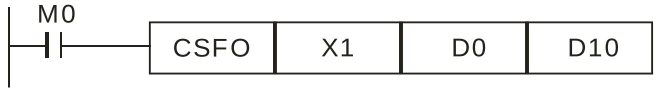
Explanations:

1. When **S** specifies X0, PLC only uses X0 input point and its associated high speed pulse output: Y0, in this case Y1 is normal output point. When **S** specifies X1, PLC uses X0 (A phase) and X1 (B phase) input points and their associated output: Y0 (Pulse) / Y1 (Dir). When **S** specifies X2, PLC only uses X2 input point and its associated high speed pulse output: Y2, in this case Y3 is normal output point. When **S** specifies X3, PLC uses X2 (A phase) and X3 (B phase) input points and their associated output: Y2 (Pulse) / Y3 (Dir).
2. The execution of CSFO requires hardware high speed counter function as well as the high speed output function. Therefore, when program scan proceeds to CSFO instruction with high speed counter input points (X0, X1) or (X2, X3) enabled by DCNT instruction, or high speed pulse outputs (Y0, Y1) or (Y2, Y3) enabled by other high speed output instructions, CSFO instruction will not be activated.
3. If **S** specifies X1 / X3 with 2-phase 2 inputs, the counting mode is fixed as one time frequency.
4. During pulse output process of Y0 or Y2, special registers (D1031, D1330 / D1337, D1336) storing the current number of output pulses will be updated when program scan proceeds to this instruction.
5. **S_1** occupies consecutive 4 16-bit registers. **S_1** +0 specifies the sampling times, i.e. when **S_1** +0 specifies K1, PLC catches the speed every time when 1 pulse is outputted. Valid range for **S_1** +0 in 1-phase 1-input mode: K1~K100, and 2-phase 2-input mode: K2~K100. If the specified value exceeds the valid range, PLC will take the lower/upper bound value as the set value. Sample time can be changed during PLC operation, however the modified value will take effect until program scan proceeds to this instruction. **S_1**+1 indicates the latest speed sampled by PLC (Read-only). Unit: 1Hz. Valid range: ±10kHz. **S_1**+2 and **S_1**+3 indicate the accumulated number of pulses in 32-bit data (Read-only).

6. **S_1** +0 specifies the sampling times. The set value of sampling times is recommended to be bigger when the input speed increases, so as to achieve a higher accuracy for speed catching. For example, set **S_1** +0 as K1 for the speed range 1Hz~1KHz, K10 for the speed range 10Hz~10KHz, K100 for the speed range 100Hz~10KHz. For single phase input, the max frequency is 10kHz; for 2-phase 2 inputs, the max frequency is 2kHz.
7. **D** occupies 3 consecutive 16-bit registers. **D** +0 specifies the output proportion value. Valid range: K1 (1%) ~ K10000 (10000%). If the specified value exceeds the valid range, PLC will take the lower/upper bound value as the set value. Output proportion can be changed during PLC operation, however the modified value will take effect until program scan proceeds to this instruction. **D**+2 and **D**+1 indicates the output speed in 32-bit data. Unit: 1Hz. Valid range: ±100kHz.
8. The speed sampled by PLC will be multiplied with the output proportion **D**+0, then PLC will generate the actual output speed. PLC will take the integer of the calculated value, i.e. if the calculated result is smaller than 1Hz, PLC will output with 0Hz. For example, input speed: 10Hz, output proportion: K5 (5%), then the calculation result will be 10 x 0.05 = 0.5Hz. Pulse output will be 0Hz; if output proportion is modified as K15 (15%), then the calculation result will be 10 x 0.15 = 1.5Hz. Pulse output will be 1Hz.

Program Example:

1. If D0 is set as K2, D10 is set as K100:
   When the sampled speed on (X0, X1) is +10Hz (D1 = K10), (Y0, Y1) will output pulses with +10Hz (D12, D11 = K10); When the sampled speed is -10Hz (D1 = K-10), (Y0, Y1) will output pulses with -10Hz (D12, D11 = K-10)
2. If D0 is set as K2, D10 is set as K1000:
   When the sampled speed on (X0, X1) is +10Hz (D1 = K10), (Y0, Y1) will output pulses with +100Hz (D12, D11 = K100); When the sampled speed is -100Hz (D1 = K-100), (Y0, Y1) will output pulses with -100Hz (D12, D11 = K-100)
3. If D0 is set as K10, D10 is set as K10:
   When the sampled speed on (X0, X1) is +10Hz (D1 = K10), (Y0, Y1) will output pulses with +1Hz (D12, D11 = K1); When the sampled speed is -10Hz (D1 = K-10), (Y0, Y1) will output pulses with -1Hz (D12, D11 = K-1)



3.6.14 Positioning Control

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
155	-	DABSR	-	Absolute position read	✓	✓	✓	✓	-	13
156	-	DZRN	-	Zero return	✓	✓	✓	✓	-	17
157	-	DPLSV		Adjustable speed pulse output	✓	✓	✓	✓	-	13
158	-	DDRVI	-	Relative position control	✓	✓	✓	✓	-	17
159	-	DDRVA	-	Absolute position control	✓	✓	✓	✓	-	17
191	-	DPPMR	-	2-Axis Relative Point to Point Motion	ES2 EX2	-	✓	✓	-	17
192	-	DPPMA	-	2-Axis Absolute Point to Point Motion	ES2 EX2	-	✓	✓	-	17
193	-	DCIMR	-	2-Axis Relative Position Arc Interpolation	ES2 EX2	-	✓	✓	-	17
194	-	DCIMA	-	2-Axis Absolute Position Arc Interpolation	ES2 EX2	-	✓	✓	-	17
195	-	DPTPO	-	Single-Axis pulse output by table	ES2 EX2	✓	✓	✓	-	13
197	-	DCLLM	-	Close loop position control	ES2 EX2	✓	✓	✓	-	17
198	-	DVSPO	-	Variable speed pulse output	ES2 EX2	✓	✓	✓	-	17
199	-	DICF	✓	Immediately change frequency	ES2 EX2	✓	✓	✓	-	13

3.6.14.1 Output Points for Positioning Instructions

API	Mnemonic		PULSE	ES2/EX2/SS2 SA2/SX2/SE	ES2/EX2/SS2 SA2/SX2/SE	EC5
	16 bits	32 bits		Y0, Y2	Y1, Y3	Y0, Y2, Y4, Y6
156	-	DZRN	-	✓	-	✓
157	-	DPLSV	-	✓	-	✓
158	-	DDRVI	-	✓	✓	✓
159	-	DDRVA	-	✓	✓	✓
191	-	DPPMR	-	✓	-	-
192	-	DPPMA	-	✓	-	-
193	-	DCIMR	-	✓	-	-
194	-	DCIMA	-	✓	-	-
195	-	DPTPO	-	✓	-	-
197	-	DCLLM	-	✓	-	-
198	-	DVSPO	-	✓	-	-
199	-	DICF	✓	✓	-	-

API	Mnemonic		Operands	Function	Controllers			
155	D	ABSR	S D1 D2	Absolute position read	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S	*	*	*	*												DABSR: 13 steps
D_1		*	*	*												
D_2								*	*	*	*	*	*	*		

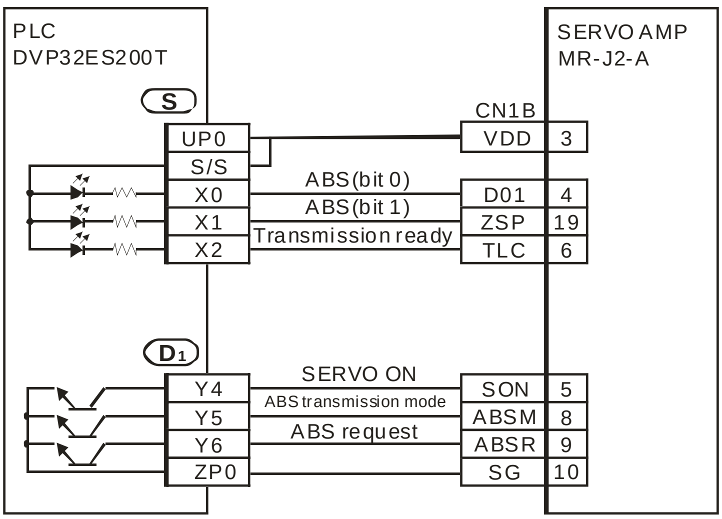
PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Input signal from servo (occupies 3 consecutive devices) **D₁**: Control signal for controlling servo (occupies 3 consecutive devices at most) **D₂**: Absolute position data (32-bit) read from servo (occupies 4 consecutive devices at most)

Explanations: (The instruction can be used in DVP-ES2/EX2 series PLCs whose version is 3.00; DVP-SA2 series PLCs whose version is 2.40; DVP-SE series PLCs whose version is 1.20; DVP-SX2 series PLCs whose version is 2.20; DVP-SS2 series PLCs (and below).)

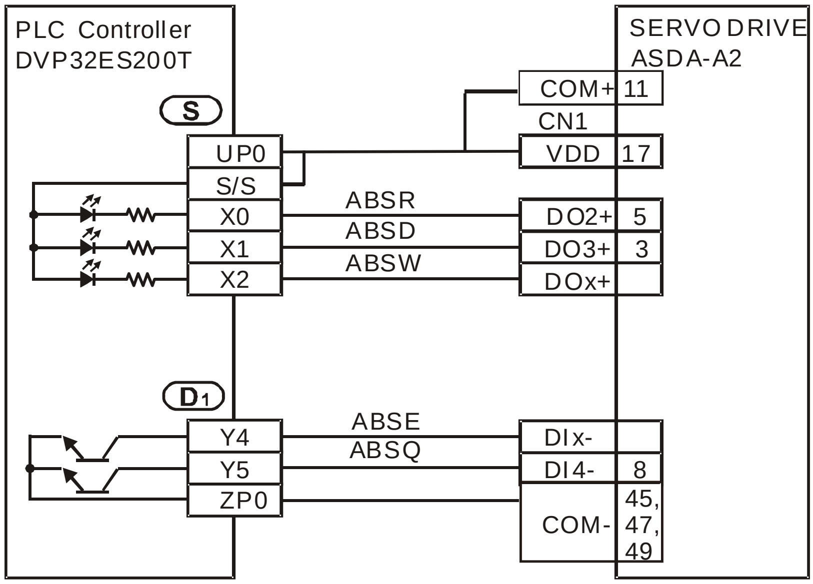
1. This instruction reads the absolute position (ABS) of servo drive with absolute position check function, e.g. MITSUBISHI MR-J2.
2. Only 32-bit instruction is applicable for ABSR instruction (DABSR) and it can only be used ONCE in the program.
3. **S**: input signal from servo. 3 consecutive devices **S**, **S** +1, **S** +2 are occupied. **S** and **S** +1 are connected to the ABS (bit0, bit1) of servo for data transmitting. **S** +2 is connected to servo for indicating transmission data being prepared.
4. **D_1**: control signal for controlling servo. 3 consecutive devices **D_1**, **D_1**+1, **D_1**+2 are occupied. **D_1** is connected to servo ON (SON) of servo, **D_1**+1 is connected to ABS transmission mode of servo and **D_1**+2 is connected to ABS request.



5. **D_2**: Absolute position data (32-bit) read from servo. 2 consecutive devices **D_2**, **D_2**+1 are occupied. **D_2** is low word and **D_2**+1 is high word.
6. When DABSR instruction is completed, M1029 will be ON. M1029 has to be reset by users.
7. Please use NO contact as the drive contact of DABSR instruction. If the drive contact is OFF during the execution of DABSR, the instruction will be stopped and errors will occur on read data.
8. If the drive contact of DABSR instruction turns OFF after the instruction is completed, the servo ON (SON) signal connected to **D_1** will also turn OFF and the operation will be disabled.

Explanations: (The instruction can be used in DVP-ES2/EX2 series PLCs whose version is 3.20; EC5: V1.00; DVP-SA2 series PLCs whose version is 2.60; DVP-SE series PLCs whose version is 1.40; DVP-SX2 series PLCs whose version is 2.40 (and above).)

1. This instruction reads the absolute position (ABS) of MITSUBISHI MR-J2 servo drive (with absolute position check function), and the absolute position (ABS) of Delta ASDA-A2 servo drive (whose firmware version is 1.045 sub12 (and above).
2. The state of M1177 determines the servo drive which is used. If M1177 is Off, MITSUBISHI MR-J2 servo drive is used. Please refer to the points above for more information about setting MITSUBISHI MR-J2 servo drive. If M1177 is On, Delta ASDA-A2 servo drive is used. Please refer to the points below for more information about settiing Delta ASDA-A2 servo drive.
3. Only 32-bit instruction is applicable for ABSR instruction (DABSR) and it can only be used ONCE in the program.
4. The input signal from a servo is stored in **S**. **S** occupies 3 consecutive devices. **S**, **S** +1, and **S** +2 are connected to ABSR, ABSD, ABSW on a servo.
5. **D_1** will occupy 2 consecutive devices, **D_1**, and **D_1** + 1. **D_1** is connected to ABSE on a servo. **D_1**+1 is connected to ABSQ on a servo. Please refer to the example below for more information about wiring.

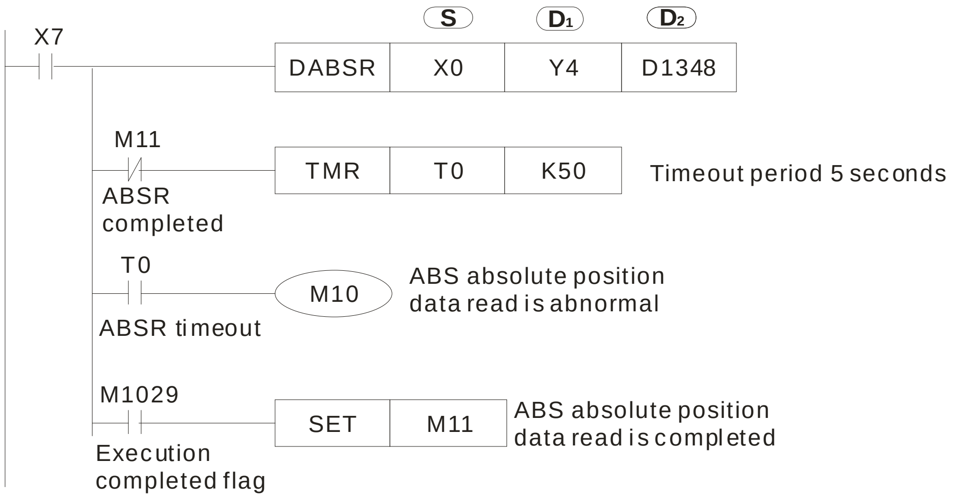


6. **D2** will occupy 4 consecutive devices **D2**, **D2** +1. **D2** +2, and **D2** +3. The absolute acoordinate system status (P0-50) is stored in **D2**, the encoder absolute position (multiturn) (P0-51) is stored in **D2** +1. The lower 16 bits of the encoder absolute position (pulse number within singleturn or PUU) (P0-52) is stored in **D2** +2. The higher 16 bits of the encoder absolute position (pulse number within singleturn or PUU) (P0-52) is stored in **D2** +3.
7. After the the reading of the absolute positio of a servo through the instruciton DABSR is complete, M1580 will be On. If an error occurs during the execution of the instruciton, M1581 will be On.
8. When driving the DABSR command, please specify normally open contact. If the drive contact of DABSR command turns Off when DABSR command read starts, the execution of absolute current value read will be interrupted and result in incorrect data. Please be careful and notice that.
9. If the input signals are from the high-speed input points X0~X7, it takes 2 seconds for the instruction to be executed. if the input signals are form the input points following X10, it takes 2.5 seconds for the instruciton to be executed. The time it takes for the instruction to be executed is affected by the scan time.

Program Example: (for DVP-ES2/EX2 series PLCs whose version is 3.00; DVP-SA2 series PLCs whose version is 2.40; DVP-SE series PLCs whose version is 1.20; DVP-SX2 series PLCs whose veresion is 2.20; DVP-SS2 series PLCs (and below))

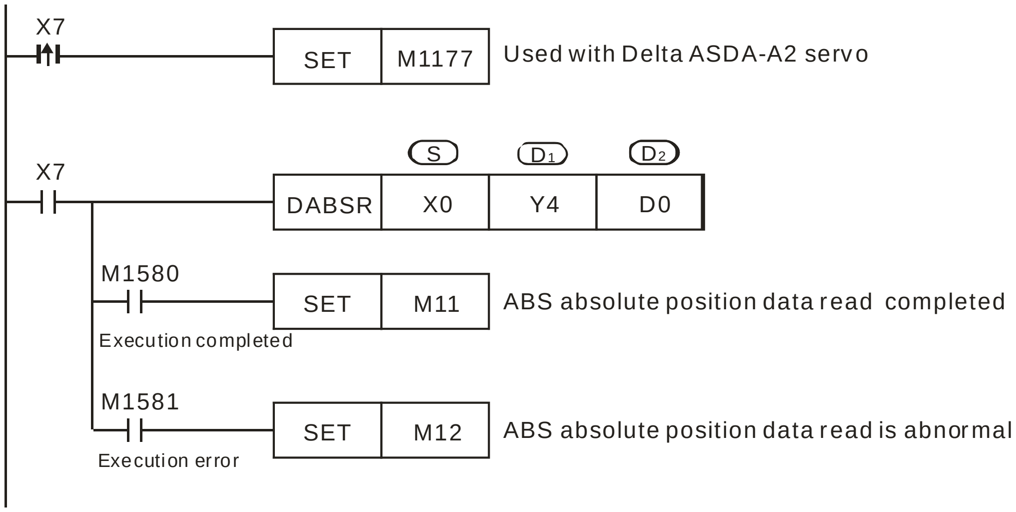
1. When X7 = ON, the 32-bit absolute position data read from Mitsubishi MR-J2 servo will be stored in the registers D0~D1. At the same time, timer T0 is enabled and starts to count for 5 seconds. If the 32-bit instruction is not completed within 5 seconds, M10 will be ON, indicating operation errors.

2. When enabling the connection to the system, please synchronize the power input of DVP-PLC and SERVO AMP or activate the power of SERVO AMP earlier than DVP-PLC.



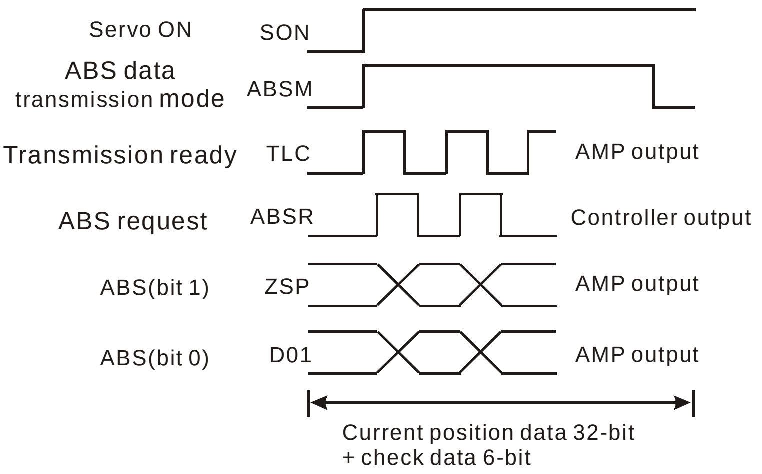
Program Example: (for DVP-ES2/EX2 series PLCs whose version is 3.20; EC5: V1.00; DVP-SA2 series PLCs whose version is 2.60/DVP-SE series PLCs whose version is 1.40; DVP-SX2 series PLCs whose version is 2.40 (and above))

1. When X7 = On, the absolute position data read from Delta ASDA-A2 servo will be stored in the registers D0~D3. The state of M1580 and the state of M1581 indicates whether the reading of the absolute position is successful.



Points to note: (Used with Mitsubishi MR-J2 Servo drive)

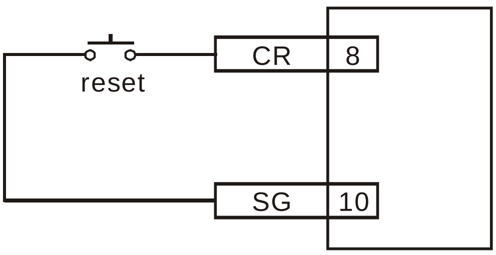
1. Timing diagram of the operation of DABSR instruction:



2. When DABSR instruction executes, servo ON (SON) and ABS data transmission mode are driven for output.
3. By “transmission ready” and “ABS request” signals, users can confirm the transmitting and receiving status of both sides as well as processing the transmission of the 32-bit ABS position data and the 6-bit check data..
4. Data is transmitted by ABS (bit0, bit1).
5. This instruction is applicable for servo drive with absolute position check function, e.g. MITSUBISHI MR-J2-A.
6. Select one of the following methods for the initial ABSR instruction:

- Execute API 156 ZRN instruction with reset function to complete zero return.
- Apply JOG function or manual adjustment to complete zero return, then input the reset signal to the servo. Please refer to the diagram below for the wiring method of reset signal. For the detailed wiring between DVP-PLC and Mitsubishi MR-J2-A, please refer to API 159 DRVA instruction.

Ex: Mitsubishi MR-J2-A



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
156	D	ZRN	S1 S2 S3 D	Zero return				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word Devices											DZRN: 17 steps
S_1					*	*	*	*	*	*	*	*	*	*		
S_2					*	*	*	*	*	*	*	*	*	*		
S_3	*															
D		*														

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Target frequency for zero return **S_2**: JOG frequency for DOG **S_3**: input device for DOG **D**: Pulse output device

Explanations (Applicable to ES2, EX2, SS2, SA2, SE, SX2 Series):

1. **S_1** (zero return speed): ranging from 6 Hz to 100k Hz; **S_2** (JOG speed for DOG) has to be lower than **S_1**. And **S_2** also refers to the start frequency.
2. **S_3** and **D** operands have to be used as an input/output set according to the table below, i.e. when **S_3** is specified as X4, **D** has to be specified as Y0; also when **S_3** is specified as X6, **D** has to be specified as Y2.
3. M1307 enables (ON) / disables (OFF) left limit switch of CH0 (Y0, Y1) and CH1 (Y2, Y3). M1307 has to be set up before the instruction executes. M1305 and M1306 can reverse the pulse output direction on Y1 and Y3 and have to be set up before instruction executes. Associated left limit switch for CH0 (Y0, Y1) is X5; associated left limit switch for CH1 (Y2, Y3) is X7. All functions, input points and output points are arranged as follows:

Input \ Channel	CH0(Y0,Y1)	CH1(Y2,Y3)	Remark
DOG point	X4	X6	
Left limit switch (M1307 = ON)	X5	X7	
The left limit switch is triggerred by a rising-edge signal or a falling-edge signal. (OFF: Rising-edge signal; ON: Falling-edge signal) (ES2/EX2/ES2-C V3.20 and above/SA2 V2.80 and above/SX2 V2.60 and above/SS2 V3.0 and above/SE V1.4 and above)	M1584	M1585	
Reverse pulse output direction	M1305	M1306	
Zero point stopping position	M1106	M1107	Please refer to point

Input \ Channel	CH0(Y0,Y1)	CH1(Y2,Y3)	Remark
			7 for the explanation.
M1346=On Start output clear signals	Y4	Y5	Please refer to point 8 for the explanation.
D1312 != 0	M1308 = Off (seeking Z-phase signal)		Please refer to point 9 for the explanation.
	X2	X3	
D1312 != 0	M1308 = On (outputting the designated number of pulses)		Please refer to point 10 for the explanation.

4. When **D** is specified as Y0, its direction signal output is Y1; when **D** is specified as Y2, its direction signal output is Y3.
5. When pulse output reaches zero point, pulse output execution completed flag M1029 (CH0), M1102 (CH1) is ON and the register indicating current position is reset to 0.
6. When DZRN instruction executes, external interrupt I400/I401(X4)) or I600/I601(X6) in program will be disabled until DZRN instruction is completed. Also. If left limit switch (X5 / X7) is enabled during instruction execution, external interrupt I500501(X5) or I700/I701(X7) will be disabled as well.
7. Zero point selection: the default position of zero point is on the left of DOG switch (the input point On→Off) (as mode 1 shows). If the user needs to change the zero point to the right of DOG switch, set ON M1106(CH0) or M1107(CH1) before DZRN instruction executes. (The function supports ES2/EX2 series, V1.20 or above.)
8. Start the pulse-clearing function of the output. When DOG leaves DOG switch and is going to stop, it will output another pulse (the width of On is about 20ms). When the pulse is On→Off, there will be a completed flag output. Please refer to state 4 for the timing diagram of this function. (The function supports ES2/EX2 series, V1.20 or above.)
9. When D1312 is not set to be 0, and M1308=Off, the function of seeking Z phase is started. When D1312 is a positive value (the maximum value is 10), it indicates that the search for Z-phase signal is toward the positive direction. When D1312 is a negative value (the minimum value is -10), it indicates that the search for Z-phase signal is toward the negative direction. For example, if D1312 is k-2, it means that DOG stops immediately after DOG leaves DOG switch and searches in the negative direction for second Z-phase signal (the fixed right-edge trigger) with JOG frequency. Please refer to state 5 for the timing diagram of this function. (The function supports ES2/EX2 series of V1.20 or above, and SS2/SX2 series of V1.20 or above.)
10. When D1312 is not set to be 0 and M1308=On, the function of outputting the designated number of pulses is started. When Dd1312 is a positive value (the maximum value is 30000), it indicates

that the pulses are output in the positive direction. When D1312 is a negative value (the minimum value is -30000), it indicates that the pulses are output in the negative direction. For example, if D1312 is k-100, it means that DOG stops immediately after DOG leaves DOG switch and another 100 pulses will be output in the negative direction with JOG frequency. Please refer to state 6 for the timing diagram of this function. (The function supports ES2/EX2 series of V1.40 or above, and SS2/SX2 series of V1.20 or above.)

11. This instruction should not be used in the interrupt programs or subroutine that only be called once or used with auto-reset flags.

Explanations (Applicable to EC5 Series):

1. $\mathbf{S_1}$ (zero return speed): ranging from 6 Hz to 100k Hz; $\mathbf{S_2}$ (JOG speed for DOG) has to be lower than $\mathbf{S_1}$. And $\mathbf{S_2}$ also refers to the start frequency.
2. $\mathbf{S_3}$ and **D** operands have to be used as an input/output set according to the table below, i.e. when $\mathbf{S_3}$ is specified as X4, **D** has to be specified as Y0; also when $\mathbf{S_3}$ is specified as X6, **D** has to be specified as Y2. **W**hen $\mathbf{S_3}$ is specified as X14, **D** has to be specified as Y4. When $\mathbf{S_3}$ is specified as X16, **D** has to be specified as Y6.
3. M1307 enables (ON) / disables (OFF) left limit switch of CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5), and CH3 (Y6, Y7). M1305, M1306, M1532, and M1533 have to be set up before the instruction executes. M1305 and M1306 can reverse the pulse output direction on Y1, Y3, Y5, Y7 and have to be set up before instruction executes. Associated left limit switch for CH0 (Y0, Y1) is X5, CH1 (Y2, Y3) is X7, CH2 (Y4, Y5) is X15, and CH3 (Y6, Y7) is X17. All functions, input points and output points are arranged as follows:

Channel / Input	CH0 (Y0,Y1)	CH1 (Y2,Y3)	CH2 (Y4,Y5)	CH3 (Y6,Y7)
DOG point	X4	X6	X14	X16
Left limit switch (M1307 = ON)	X5	X7	X15	X17
The left limit switch is triggerred by a rising-edge signal or a falling-edge signal. (OFF: Rising-edge signal; ON: Falling-edge signal)	M1584	M1585	M1586	M1587
Reverse pulse output direction	M1305	M1306	M1532	M1533
Zero point selection	M1106	M1107	M1588	M1589
M1346=On Start output clear signals	Y10	Y11	Y12	Y13
D1312 != 0	M1308 = Off (seeking Z-phase signal)			
	X2	X3	NA#	NA#
D1312 != 0	M1308 = On (outputting the designated number of pulses)			

#: Due to the lack of high-speed input for configuring the Z-phase function, the output of axes Y4 and Y6 does not support this function.

4. When **D** is specified as Y0, its direction signal output is Y1; when **D** is specified as Y2, its direction signal output is Y3. When **D** is specified as Y4, its direction signal output is Y5. When **D** is specified as Y6, its direction signal output is Y7.
5. When pulse output reaches zero point, pulse output execution completed flag M1029 (CH0), M1102 (CH1), M1321 (CH2), or M1322 (CH3) is ON and the register indicating current position is reset to 0.
6. When DZRN instruction executes, if using CH0 (Y0, Y1), CH1 (Y2, Y3), the external interrupt I400/I401(X4)) or I600/I601(X6) in program will be disabled until DZRN instruction is completed. If left limit switch (X5 / X7) is enabled during instruction execution, external interrupt I500501(X5) or I700/I701(X7) will be disabled as well.
7. When the DZRN instruction is executed and channels CH2 (Y4, Y5) and CH3 (Y6, Y7) are used, the scanning instruction method is employed to detect the zero position, negative limit position, and to stop the motion. Therefore, the execution process will be affected by the filtering time of the input points and the PLC scan time. It is recommended that the pulse width time of the near point (DOG) signal (which becomes shorter as the speed increases) should be greater than the input filtering time.
8. Zero point position: the default position of zero point is on the left of DOG switch (the input point On→Off) (as mode 1 shows below). If the user needs to change the zero point to the right of DOG switch, set ON M1106 (CH0), M1107(CH1), M1588 (CH2) or M1589 (CH3) before DZRN instruction executes.
9. Start the pulse-clearing function of the output. When DOG leaves DOG switch and is going to stop, it will output another pulse (the width of On is about 20ms). When the pulse is On→Off, there will be a completed flag output. Please refer to state 4 for the timing diagram of this function.
10. When D1312 is not set to be 0, and M1308=Off, the function of seeking Z phase is started. When D1312 is a positive value (the maximum value is 10), it indicates that the search for Z-phase signal is toward the positive direction. When D1312 is a negative value (the minimum value is -10), it indicates that the search for Z-phase signal is toward the negative direction. For example, if D1312 is k-2, it means that DOG stops immediately after DOG leaves DOG switch and searches in the negative direction for second Z-phase signal (the fixed right-edge trigger) with JOG frequency. Please refer to state 5 for the timing diagram of this function.

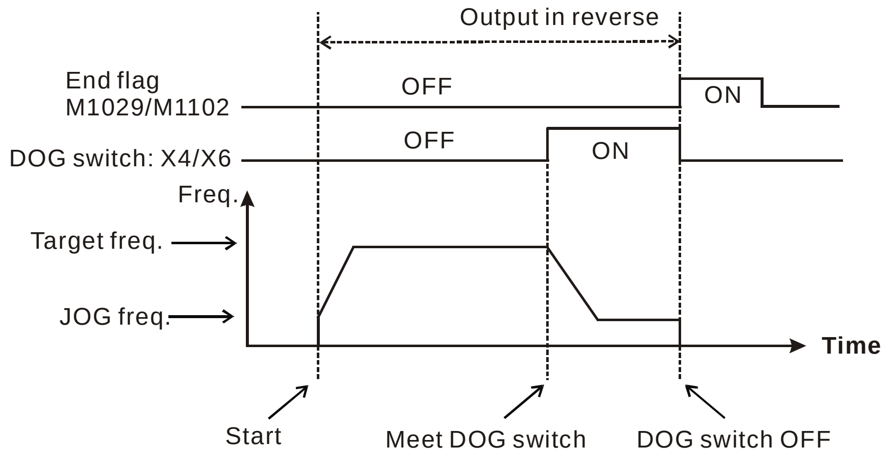
    Note: The output of axes Y4 and Y6 does not support this function.
11. When D1312 is not set to be 0 and M1308=On, the function of outputting the designated number of pulses is started. When Dd1312 is a positive value (the maximum value is 30000), it indicates

that the pulses are output in the positive direction. When D1312 is a negative value (the minimum value is -30000), it indicates that the pulses are output in the negative direction. For example, if D1312 is k-100, it means that DOG stops immediately after DOG leaves DOG switch and another 100 pulses will be output in the negative direction with JOG frequency. Please refer to state 6 for the timing diagram of this function.

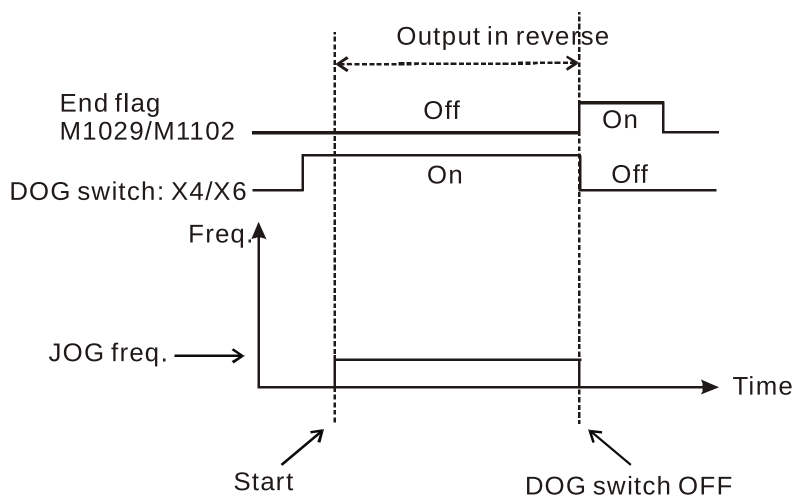
12. This instruction should not be used in the interrupt programs or subroutine that only be called once or used with auto-reset flags.

Timing Diagram:

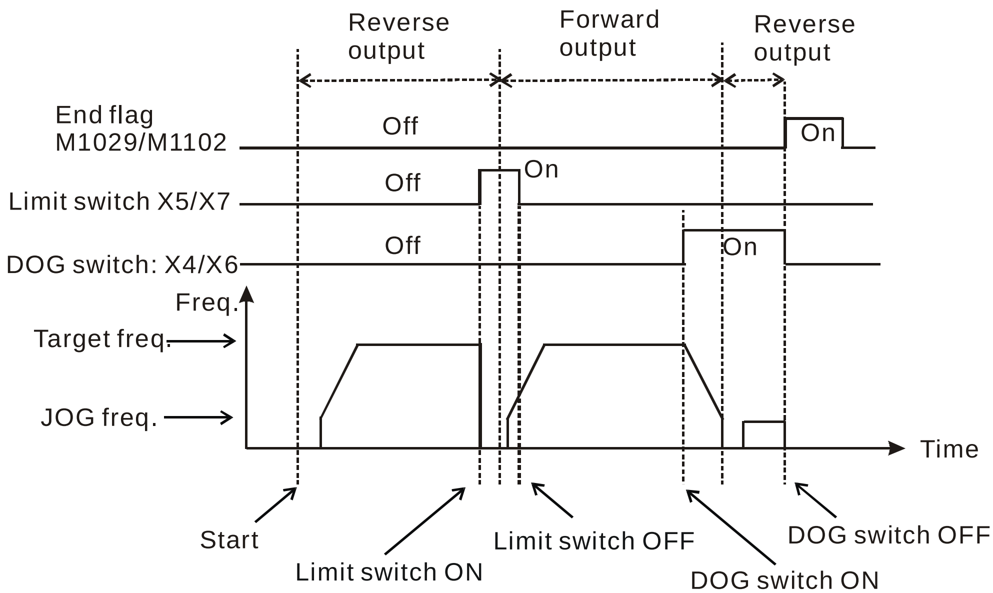
State 1: Current position at right side of DOG switch, pulse output in reverse, limit switch disabled.



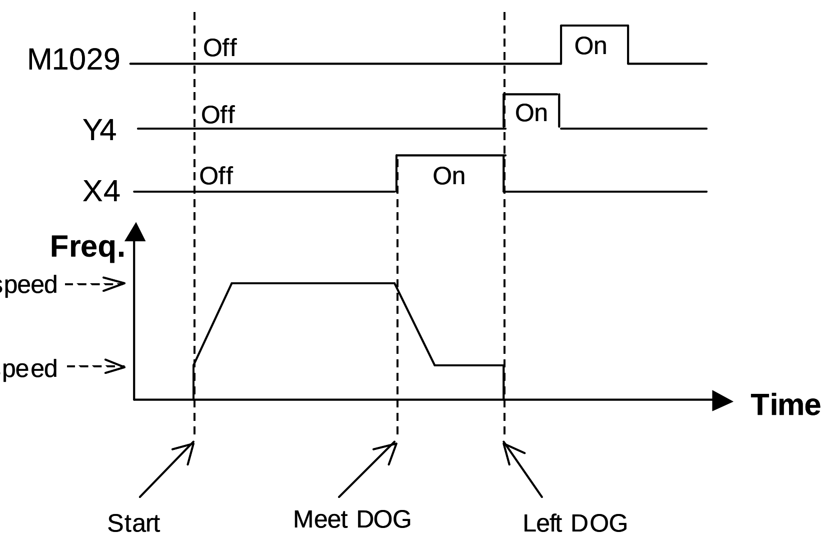
State 2: DOG switch is ON, pulse output in reverse, limit switch disabled.



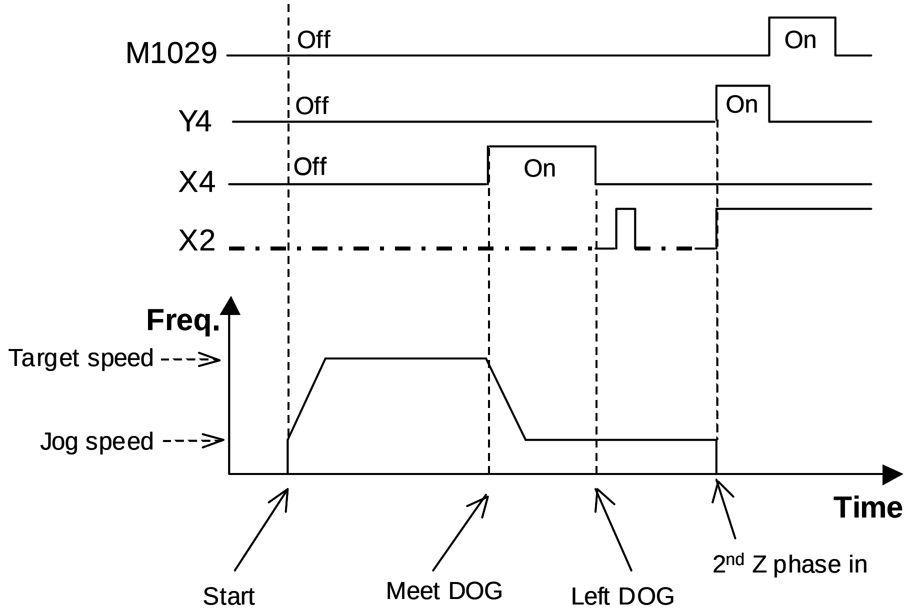
State 3: Current position at left side of zero point, pulse output in reverse, limit switch enabled.



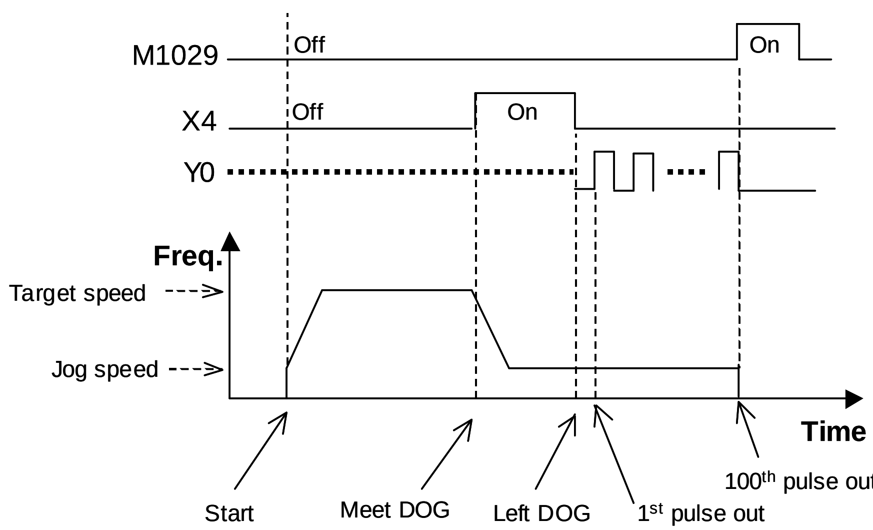
State 4: Current position at right side of zero point, M1346=On.



State 5: Current position at right side of zero point, D1312=-2, M1308=Off, M1346=On.

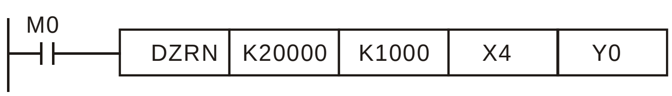


State 6: Current position at right side of zero point, D1312=-100, M1308=On.



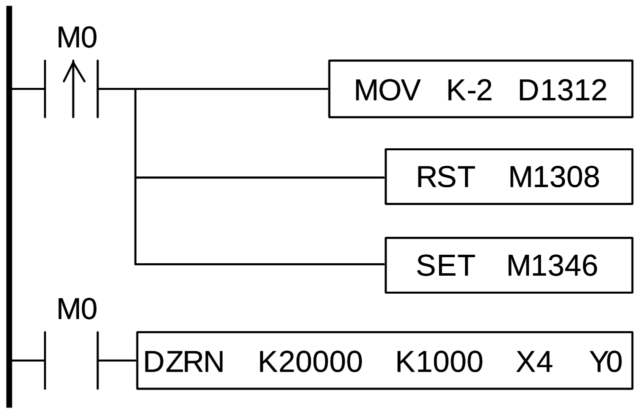
Program Example 1:

When M0 = ON, Y0 pulse output executes zero return with a frequency of 20kHz. When it reaches the DOG switch, X4 = ON and the frequency changes to JOG frequency of 1kHz. Y0 will then stop when X4 = OFF.



Program Example 2:

When M0 = ON, Y0 pulse output executes zero return with a frequency of 20kHz. When it reaches the DOG switch, X4 = ON and the frequency changes to JOG frequency of 1kHz. When X4 = OFF, it seeks the second X2(Z-phase) pulse input (right-edge trigger signal), and Y4 stops after a pulse (the width of On is 20ms) is output from it (M1029=On).



Points to note:

1. Associated Flags:

M1029:	CH0 (Y0, Y1) pulse output execution completed
M1102:	Y2/CH1 (Y2, Y3) pulse output execution completed
M1106:	Zero point selection. M1106=ON, change the zero point to the right of DOG switch for zero return on CH0
M1107:	Zero point selection. M1107=ON, change the zero point to the right of DOG switch for zero return on CH1
M1305:	Reverse Y1 pulse output direction in high speed pulse output instructions
M1306:	Reverse Y3 pulse output direction in high speed pulse output instructions
M1307:	For ZRN instruction, enable left limit switch
M1308:	Output specified pulses (D1312) or seek Z phase signal when zero point is achieved.
M1346:	Output clear signals when ZRN is completed

2. Special D registers:

D1312:	Specify the number of additional pulses for additional pulses output and Z-phase seeking function of ZRN instruction (Has to be used with M1308)

API	Mnemonic		Operands	Function	Controllers			
157	D	PLSV	S D1 D2	Adjustable Speed Pulse Output	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S					*	*	*	*	*	*	*	*	*	*	*	PLSV: 7 steps DPLSV: 13 steps
D_1		*														
D_2		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Pulse output frequency **D_1**: Pulse output device (Y0, Y2, Y4, Y6) **D_2**: Direction signal output

Explanations:

1. The instruction only supports the pulse output type: Pulse + Direction.
2. **S** is the designated pulse output frequency. "+/-" signs indicate forward/reverse output direction. The frequency can be changed during pulse output. However, if the specified output direction is diferent from the current output direction, the instruction will stop for 1 scan cycle then restart with the changed frequency.
3. The operation of **D_2** corresponds to the "+" or "-" of **S**. When **S** is "+", **D_2** will be OFF; when **S** is "-", **D_2** will be ON.
4. M1305 and M1306 can change the output direction of CH0/CH1 set in **D_2**. When **S** is "-", **D_2** will be ON, however, if M1305/M1306 is set ON before instruction executes, **D_2** will be OFF during execution of instruction. For series, other than EC5 Series.

Channel / Input Point	CH0	CH1
Pulse output frequency (S)	-100,000Hz ~ +100,000Hz	
Pulse output device (D_1)	Y0	Y2
Directon signal ouptut (D_2)	Y1	Y3
Reverse pulse output direction	M1305	M1306

For EC5 Series

Channel / Input Point	CH0	CH1	CH2	CH3
Pulse output frequency (S)	-50,000 Hz ~ +50,000 Hz			
Pulse output device (D_1)	Y0	Y2	Y4	Y6
Directon signal ouptut (D_2)	Y1	Y3	Y5	Y7
Reverse pulse output direction	M1305	M1306	M1532	M1533

5. PLSV instruction does not support settings for ramp up or ramp down. If ramp up/down process is required, please use API 67 RAMP instruction.
6. If the drive contact turns off during pulse output process, pulse output will stop immediately.

Program Example:

When M10 = ON, Y0 will output pulses at 20kHz. Y1 = OFF indicates forward direction.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
158	D	DRVI	S1 S2 D1 D2	Relative Position Control				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DDRVI: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D_1		*														
D_2		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Number of pulses (relative positioning) **S_2**: Pulse output frequency **D_1**: Pulse output device **D_2**: Direction signal output

Explanations (Applicable to ES2, EX2, SS2, SA2, SE, SX2 Series):

1. The instruction only supports the pulse output type: Pulse + Direction.
2. **S_1** is the number of pulses (relative positioning). Available range: -2,147,483,648 ~ +2,147,483,647. "+/-" signs indicate forward and reverse direction.
3. **S_2** is the pulse output frequency. Available range: 6 ~ 100,000Hz.
4. **D_1** is the pulse output device. It can designate CH0 (Y0) and CH1 (Y2).
5. **D_2** is the direction signal output device. It can designate CH0 (Y1) and CH1 (Y3).

Pulse output device (**D_1**)	Y0	Y2
Corresponding direction signal output device **(D_2)**	Y1	Y3

6. ES2/EX2: V3.46, ES2-C: V3.48, ES2-E: V1.00, 12SE: V2.02, 26SE: V1.0 and later versions support the settings in **D_1** and **D_2** as shown below.

Pulse output device (**D_1**)	Y0	Y1	Y2	Y3
Corresponding direction signal output device **(D_2)**	Y4	Y5	Y6	Y7

7. The operation of **D_2** corresponds to the "+" or "-" of **S**. When **S** is "+", **D_2** will be OFF; when **S** is "-", **D_2** will be ON. **D_2** will not be OFF immediately after pulse output completion and will be OFF when the drive contact is OFF.
8. The set value in **S_1** is the relative position of
   - current position (32-bit data) of CH0 (Y0, Y1) which is stored in D1031(high), D1030 (low)
   - current position (32-bit data) of CH1 (Y2, Y3) which is stored in D1337(high), D1336 (low). In reverse direction pulse output, value in (D1031, D1330) and (D1336, D1337) decreases.

9. D1343 (D1353) is the ramp up/down time setting of CH0 (CH1). Available range: 20 ~ 32,767ms. Default: 100ms. PLC will take the upper/lower bound value as the set value when specified value exceeds the available range.
10. D1340 (D1352) is start/end frequency setting of CH0 (CH1). Available range: 6 to 100,000Hz. PLC will take the upper/lower bound value as the set value when specified value exceeds the available range.
11. M1305 and M1306 can change the output direction of CH0/CH1 set in $\mathbf{D_2}$. When **S** is "-", $\mathbf{D_2}$ will be ON, however, if M1305/M1306 is set ON before instruction executes, $\mathbf{D_2}$ will be OFF during execution of instruction..
12. Ramp-down time of CH0 and CH1 can be particularly modified by using (M1534, D1348) and (M1535, D1349). When M1534 / M1535 = ON, CH0 / CH1 ramp-down time is specified by D1348 / D1349.
13. If M1078 / M1104 = ON during instruction execution, Y0 / Y2 will pause immediately and M1538 / M1540 = ON indicates the pause status. When M1078 / M1104 = OFF, M1538 / M1540 = OFF, Y0 / Y2 will proceed to finish the remaining pulses.
14. DRVI instruction supports Alignment Mark and Mask function. Please refer to the explanation in API 59 PLSR instruction.
15. When M1334 or M1335 is enabled, execute API158 DDRVI instruction on CH0 (CH1) to ramp-down when the conditional contacts are closed. After the conditional contacts are closed, if you need to use the flags M1334/M1335 to stop the deceleration, you need to enable the flags M1334/M1335 again. This function is available for the followings:

Series	ES2/EX2	ES2-C	ES2-E	EC5	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware version	V3.42	V3.48	V1.00	V1.0	V2.86	V3.28	V2.02	V1.0	V3.0

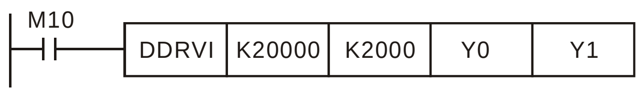
Explanations (Applicable to EC5 Series):

1. The instruction only supports the pulse output type: Pulse + Direction.
2. $\mathbf{S_1}$ is the number of pulses (relative positioning). Available range: -2,147,483,648 ~ +2,147,483,647. “+/-” signs indicate forward and reverse direction.
3. D1340 (D1352, D1379, D1380) is start/end frequency setting of CH0 (CH1, CH2, CH3). Even if the value in $\mathbf{S_2}$ is less than the starting /ending frequency, it still outputs the freuency as the value in $\mathbf{S_2}$ indicated.
4. The special D and special M:

Channel / Input Point	CH0	CH1	CH2	CH3
Pulse output device (D_1)	Y0	Y2	Y4	Y6
Directon signal ouptut (D_2)	Y1	Y3	Y5	Y7
Reverse pulse output direction	M1305	M1306	M1532	M1533
Current position	D1030/ D1031	D1336/ D1337	D1375/ D1376	D1377/ D1378
Ramp up / down time of the 1st group pulse output	D1343	D1353	D1381	D1382
Start / end frequency of the 1st group pulse output	D1340	D1352	D1379	D1380
Enable ramp-down time setting	M1534	M1535	M1536	M1537
Ramp-down time	D1348	D1349	D1350	D1351
Pulse output pause (immediate)	M1078	M1104	M1310	M1311
Pause state	M1538	M1540	M1542	M1543
Ramp-down when the conditional contacts are closed.	M1334	M1335	M1520	M1521

Program Example:

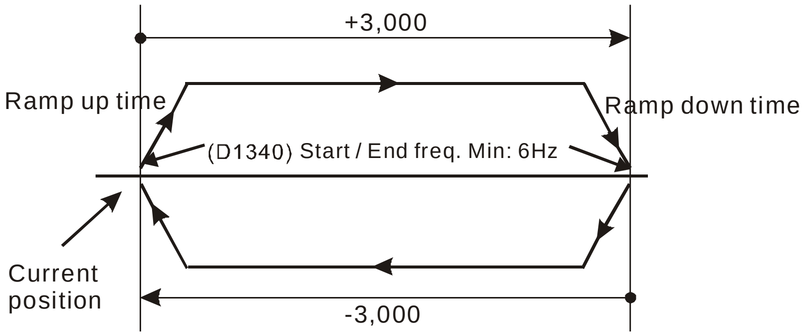
When M10= ON, 20,000 pulses (relative position) at 2kHz frequency will be generated from Y0. Y1= OFF indicates positive direction.



Points to note:

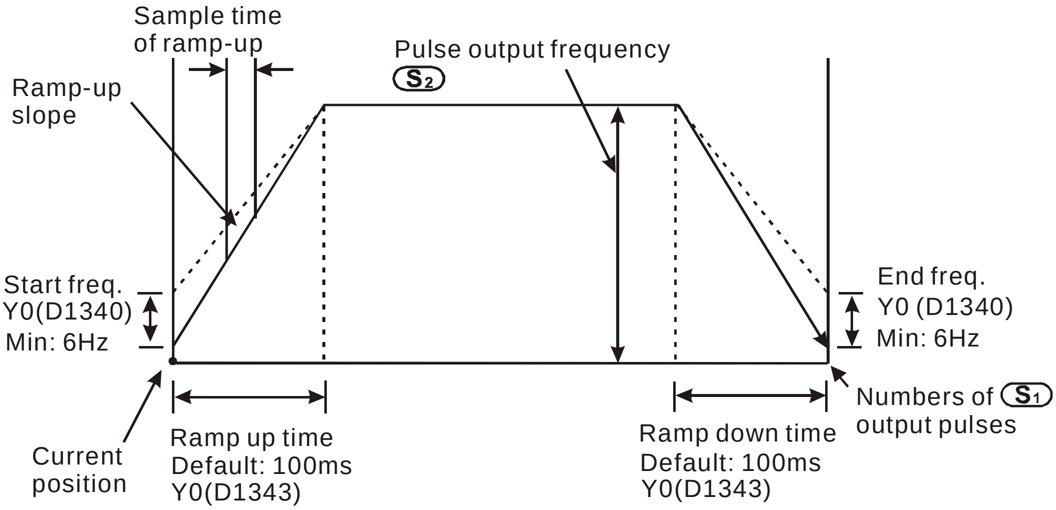
1. Operation of relative positioning:

Pulse output executes according to the relative distance and direction from the current position



2. Registers for setting ramp up/down time and start/end frequency:

- Output Y0:



- This instruction can be used many times in user program, but only one instruction will be activated at a time. For example, if Y0 is currently activated, other instructions use Y0 won't be executed. Therefore, instructions first activated will be first executed.
- After activating the instruction, all parameters cannot be modified unless instruction is OFF.

3. Associated Flags:

M1029: CH0 (Y0, Y1) pulse output execution completed.

M1102: CH1 (Y2, Y3) pulse output execution completed

M1078: CH0 (Y0, Y1) pulse output pause (immediate)

M1104: CH1 (Y2, Y3) pulse output pause (immediate)

M1108: CH0 (Y0, Y1) pulse output pause (ramp down).

M1110: CH1 (Y2, Y3) pulse output pause (ramp down)

M1119: Enabling the DDRVI/DDRVA two speed output function

M1156: Enabling the mask and alignment mark function on I400/I401(X4) corresponding to Y0.

M1158: Enabling the mask and alignment mark function on I600/I601(X6) corresponding to Y2.

M1305: Reverse Y1 pulse output direction in high speed pulse output instructions

M1306: Reverse Y3 pulse output direction in high speed pulse output instructions

M1347: Auto-reset Y0 when high speed pulse output completed

M1524: Auto-reset Y2 when high speed pulse output completed

M1534: Enable ramp-down time setting on Y0. Has to be used with D1348

M1535: Enable ramp-down time setting on Y2. Has to be used with D1349.

M1538: Indicating pause status of CH0 (Y0, Y1)

M1540: Indicating pause status of CH1 (Y2, Y3)

4. Special D registers:

D1030: Low word of the present value of Y0 pulse output

D1031: High word of the present value of Y0 pulse output

D1336: Low word of the present value of Y2 pulse output

D1337: High word of the present value of Y2 pulse output

D1340: Start/end frequency of the 1st group pulse output CH0 (Y0, Y1)

D1352: Start/end frequency of the 2nd group pulse output CH1 (Y2, Y3)

D1343: Ramp up/down time of the 1st group pulse output CH0 (Y0, Y1)

D1353: Ramp up/down time of the 2nd group pulse output CH1 (Y2, Y3)

D1348: CH0(Y0, Y1) pulse output. When M1534 = ON, D1348 stores the ramp-down time

D1349: CH1(Y2, Y3) pulse output. When M1535 = ON, D1349 stores the ramp-down time

D1232: Output pulse number for ramp-down stop when Y0 masking sensor receives signals. (LOW WORD)

D1233: Output pulse number for ramp-down stop when Y0 masking sensor receives signals. (HIGH WORD).

D1234: Output pulse number for ramp-down stop when Y2 masking sensor receives signals (LOW WORD).

D1235: Output pulse number for ramp-down stop when Y2 masking sensor receives signals (HIGH WORD).

D1026: Pulse number for masking Y0 when M1156 = ON (Low word)

D1027: Pulse number for masking Y0 when M1156 = ON (High word)

D1135: Pulse number for masking Y2 when M1158 = ON (Low word)

D1136: Pulse number for masking Y2 when M1158 = ON (High word)

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
159	D	DRVA	S_1 S_2 D_1 D_2	Absolute Position Control				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DRVA: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DDRVA: 17 steps
D_1		*														
D_2		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Numbers of pulses (Absolute positioning) **S_2**: Pulse output frequency **D_1**: Pulse output device **D_2**: Direction signal output

Explanations (Applicable to ES2, EX2, SS2, SA2, SE, SX2 Series):

1. The instruction only supports the pulse output type: Pulse + Direction.
2. **S_1** is the number of pulses (Absolute positioning). Available range: -2,147,483,648 ~ +2,147,483,647. "+/-" signs indicate forward and reverse direction.
5. **S_2** is the pulse output frequency. Available range: 6 ~ 100,000Hz.
6. **D_1** is the pulse output device. It can designate CH0 (Y0) and CH1 (Y2).
7. **D_2** is the direction signal output device. If Y output is designated, only CH0 (Y1) and CH1 (Y3) are available.

Pulse output device (**D_1**)	Y0	Y2
Corresponding direction signal output device **(D_2)**	Y1	Y3

8. ES2/EX2: V3.46, ES2-C: V3.48, ES2-E: V1.00, 12SE: V2.02, 26SE: V1.0 and later versions support the settings in **D_1** and **D_2** as shown below.

Pulse output device (**D_1**)	Y0	Y1	Y2	Y3
Corresponding direction signal output device **(D_2)**	Y4	Y5	Y6	Y7

9. **S_1** is the target position for absolute positioning. The actual number of output pulses (**S_1** – current position) will be calculated by PLC. When the result is positive, pulse output executes forward operation, i.e. **D_2** = OFF; when the results is negative, pulse output executes reverse operation, i.e. **D_2** = ON.
10. The set value in **S_1** is the absolute position from zero point. The calculated actual number of output pulses will be the relative position of
    - current position (32-bit data) of CH0 (Y0, Y1) which is stored in D1031(high), D1030 (low)
    - current position (32-bit data) of CH1 (Y2, Y3) which is stored in D1337(high), D1336 (low).

In reverse direction pulse output, value in (D1031, D1330) and (D1336, D1337) decreases.

11. D1343 (D1353) is the ramp up/down time (between start frequency and pulse output frequency) setting of CH0 (CH1). Available range: 20 ~ 32,767ms. Default: 100ms. PLC will take 20ms as the set value when specified value is below 20ms or above 32,767ms.
12. D1340 (D1352) is start/end frequency setting of CH0 (CH1). Available range: 6 ~ 32,767Hz. PLC will take the start/end frequency as the pulse output frequency when pulse output frequency **S**$_2$ is smaller or equals the start/end frequency.
13. M1305 and M1306 can change the output direction of CH0/CH1 set in **D**$_2$. When **S** is "**-**", **D**$_2$ will be ON, however, if M1305/M1306 is set ON before instruction executes, **D**$_2$ will be OFF during execution of instruction.
14. Ramp-down time of CH0 and CH1 can be particularly modified by using (M1534, D1348) and (M1535, D1349). When M1534 / M1535 = ON, CH0 / CH1 ramp-down time is specified by D1348 / D1349.
15. If M1078 / M1104 = ON during instruction execution, Y0 / Y2 will pause immediately and M1538 / M1540 = ON indicates the pause status. When M1078 / M1104 = OFF, M1538 / M1540 = OFF, Y0 / Y2 will proceed to finish the remaining pulses.
16. DRVA/DDRVA instructions do NOT support Alignment Mark and Mask function.
17. When M1334 or M1335 is enabled, execute API158 DDRVI instruction on CH0 (CH1) to ramp-down when the conditional contacts are closed. After the conditional contacts are closed, if you need to use the flags M1334/M1335 to stop the deceleration, you need to enable the flags M1334/M1335 again. This function is available for the followings:

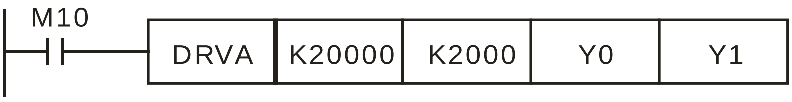
Series	ES2/EX2	ES2-C	ES2-E	12SA2/ SX2	SS2	12SE	26SE	28SA2
Firmware version	V3.42	V3.48	V1.00	V2.86	V3.28	V2.02	V1.0	V3.0

Explanations (Applicable to EC5 Series):

1. The instruction only supports the pulse output type: Pulse + Direction.
2. **S**$_1$ is the number of pulses (Absolute positioning). Available range: -2,147,483,648 ~ +2,147,483,647. "+/-" signs indicate forward and reverse direction.
3. **S**$_2$ is the pulse output frequency. Available range: 6 ~ 50,000Hz.
4. **S**$_1$ is the target position for absolute positioning. The actual number of output pulses (**S**$_1$ – current position) will be calculated by PLC. When the result is positive, pulse output executes forward operation, i.e. **D**$_2$ = OFF; when the results is negative, pulse output executes reverse operation, i.e. **D**$_2$ = ON.
5. The instruction DRVA shares the same table of supported special D and special M, refer to instruction DRVI for more information.

Program Example:

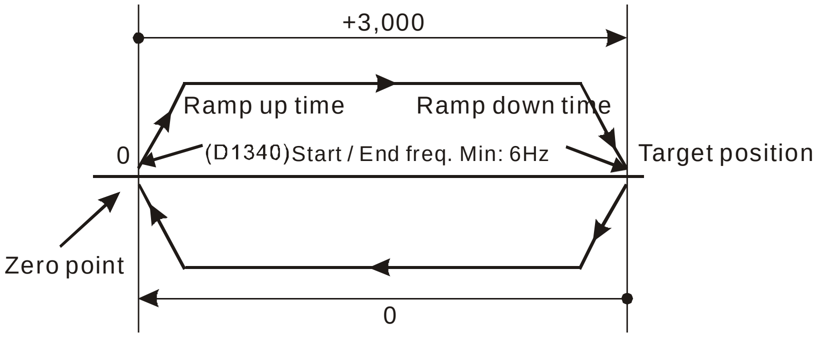
When M10 = ON, DRVA instruction executes absolute positioning on Y0 at target position 20000, target frequency 2kHz. Y1 = OFF indicates positive direction.



Points to note:

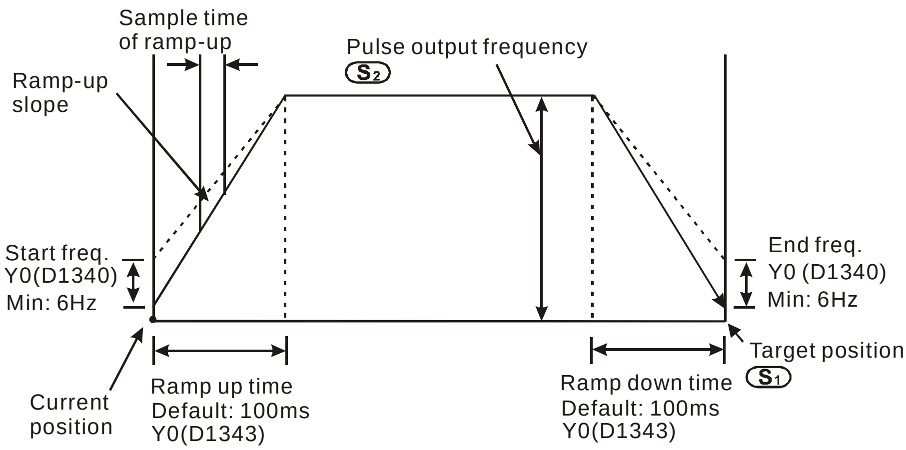
1. Operation of absolute positioning:

   Pulse output executes according to the specified absolute position from zero point



2. Registers for setting ramp up/down time and start/end frequency:

- Output Y0:



- This instruction can be used many times in user program, but only one instruction will be activated at a time. For example, if Y0 is currently activated, other instructions use Y0 won't be executed. Therefore, instructions first activated will be first executed.
- After activating the instruction, all parameters cannot be modified unless instruction is OFF.
- For associated special flags and special registers, please refer to **Points to note** of DDRVI instruction.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
191	D	PPMR	S_1 S_2 S D	2-Axis Relative Point to Point Motion				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DPPMR: 17 steps
S_1					*	*							*			
S_2					*	*							*			
S					*	*							*			
D		*														

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Number of output pulses on X axis **S_2**: Number of output pulses on Y axis **S**: Max. point to point output frequency **D**: Pulse output device

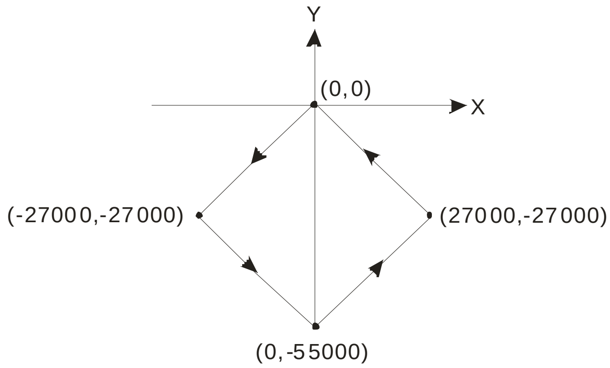
Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function.
2. The instruction only supports the pulse output type: Pulse / Direction.
3. **S_1** and **S_2** specify the number of output pulses (relative positioning) on X axis (Y0) and Y axis (Y2). Range: -2,147,483,648 ~ +2,147,483,647 (The "+/-" sign indicates the forward/backward direction). In forward direction, the present value of pulse output on CH0 (D1031 High, D1030 low), CH1 (D1337 high, D1336 low) increases. In reverse direction pulse output, value in (D1031, D1330) and (D1336, D1337) decreases.
4. **S**: If the max output frequency is smaller than 100Hz, the output will be operated at 100Hz. If the setting is bigger than 100kHz, the output will be operated at 100kHz
5. **D** can designate Y0 only.
   Y0 is the pulse output point of X axis;
   Y1 is the direction signal output of X axis.(OFF: positive; ON: negative)
   Y2 is the pulse output point of Y axis;
   Y3 is the direction signal output of Y axis (OFF: positive; ON: negative)
   When the pulse output is completed, the direction output signal will not be OFF unless the drive contact is OFF.
6. D1340 is start/end frequency setting of X/Y axis. When the set value is smaller than 6Hz, PLC will take 6 Hz as the set value. D1343 is the ramp up/down time setting of X/Y axis. If the ramp up/down time is shorter than 20ms, the frequency will be operated at 20ms. Default: 100ms.
7. When PPMR instruction is enabled, the start frequency and acceleration/deceleration time in Y axis will be the same as the settings in X axis. In addition, setting ramp-down time individually by D1534 is not recommended because it could lead to the inconsistency between X and Y axes. Also, the flags of "pulse output pause (immediate)" are not applicable. To stop the pulse output, simply turn off the drive contact of this instruction.

8. For pulse output with ramp-up/down section, if only 1 axis is specified with pulse output number, i.e. another axis is 0, the pulse output will only be performed on the axis with output pulse number. However, if the output pulse number is less than 20 in any of the 2 axes, the ramp-up/down section will be disabled and pulse output will be executed with the frequency not higher than 3kHz.
9. There is no limitation on the number of times for using the instruction. However, assume CH0 or CH1 pulse output is in use, the X/Y axis synchronized output will not be performed.
10. M1029 will be ON when 2-axis synchronized pulse output is completed.

Program Example:

1. Draw a rhombus as the figure below.

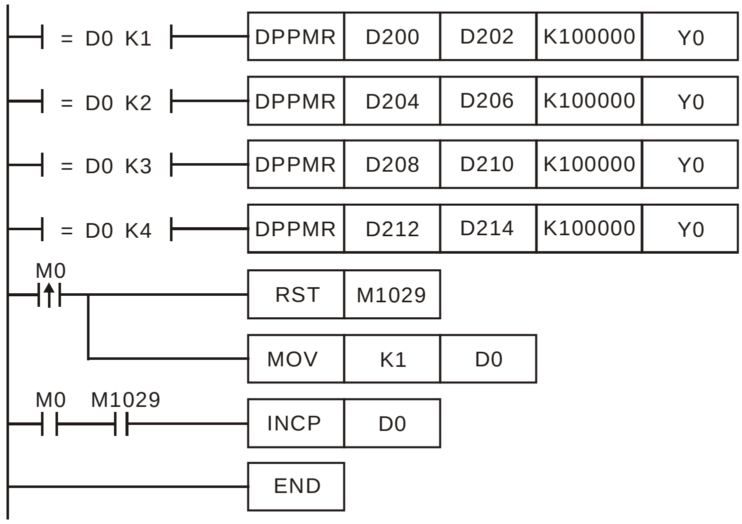


Steps:

a) Set the four coordinates (0,0), (-27000, -27000), (0, -55000), (27000, -27000) (as the figure above). Calculate the relative coordinates of the four points and obtain (-27000, -27000), (27000, -28000), (27000, 27000), and (-27000, 27000). Place them in the 32-bit registers (D200, D202), (D204, D206), (D208, D210), (D212, D214).

b) Design instructions as follows.

c) RUN the PLC. Set ON M0 to start the 2-axis line drawing.



2. Operation:

   When PLC runs and M0 = ON, PLC will start the first point-to-point motion by 100KHz. D0 will plus 1 whenever a point-to-point motion is completed and the second point-to-point motion will start to execute automatically. The operation pattern repeats until the fourth point-to-point motion is completed.

Points to note:

Associated flags and registers:

M1029:	CH0 (Y0, Y1) pulse output execution completed
D1030:	Present number of Y0 output pulses (HIGH WORD).
D1031:	Present number of Y1 output pulses (LOW WORD).
D1336:	Present value of Y2 pulse output. D1336 (High word)
D1337:	Present value of Y2 pulse output. D1337(Low word)
D1340:	Start/end frequency of pulse output CH0 (Y0), CH1(Y2) for DPPMR/DPPMA instruction
D1343:	Ramp up/down time of pulse output CH0 (Y0), CH1(Y2) for DPPMR/DPPMA instruction

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
192	D	PPMA	(S1) (S2) (S) (D)	2-Axis Absolute Point to Point Motion				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																DPPMA: 17 steps
S_1					*	*							*			
S_2					*	*							*			
S					*	*							*			
D		*														

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Number of output pulses on X axis **S_2**: Number of output pulses on Y axis **S**: Max. point to point output frequency **D**: Pulse output device

Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function.
2. The instruction only supports the pulse output type: Pulse / Direction.
3. **S_1** and **S_2** specify the number of output pulses (absolute positioning) on X axis (Y0) and Y axis (Y2). Range: -2,147,483,648 ~ +2,147,483,647 (The “+/-“ sign indicates the forward/backward direction). In forward direction, the present value of pulse output on CH0 (D1031 High, D1030 low), CH1 (D1337 high, D1336 low) increases. In reverse direction pulse output, value in (D1031, D1330) and (D1336, D1337) decreases.
4. **D** can designate Y0 only.

   Y0 is the pulse output point of X axis;

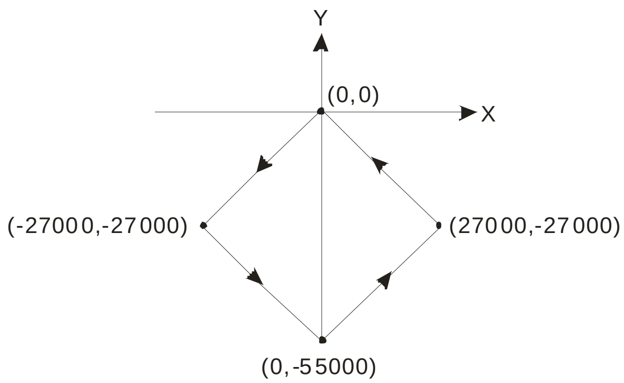
   Y1 is the direction signal output of X axis.(OFF: positive; ON: negative)

   Y2 is the pulse output point of Y axis;

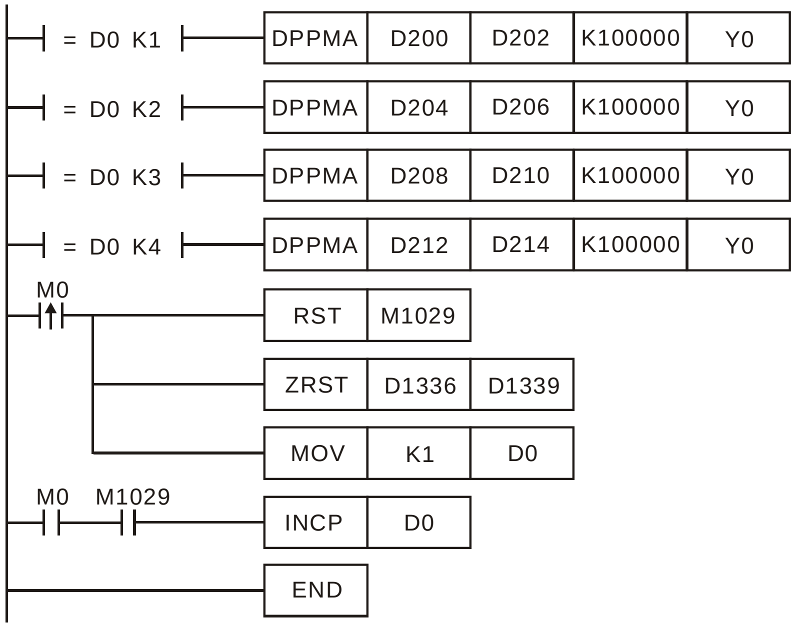
   Y3 is the direction signal output of Y axis (OFF: positive; ON: negative)
5. For the rest of the explanations on the instruction, special D and special M, please refer to API 191 DPPMR instruction.

Program Example:

1. Draw a rhombus as the figure below.



2. Steps:
   a) Set the four coordinates (-27000, -27000), (0, -55000), (27000, -27000) and (0,0) (as the figure above). Place them in the 32-bit registers (D200, D202), (D204, D206), (D208, D210), (D212, D214).
   b) Design instructions as follows.
   c) RUN the PLC. Set ON M0 to start the 2-axis line drawing.



3. Operation:
   When PLC runs and M0 = ON, PLC will start the first point-to-point motion by 100KHz. D0 will plus 1 whenever a point-to-point motion is completed and the second point-to-point motion will start to execute automatically. The operation pattern repeats until the fourth point-to-point motion is completed.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
193	D	CIMR	S1 S2 S D	2-Axis Relative Position Arc Interpolation				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																DCIMR: 17 steps
S_1					*	*							*			
S_2					*	*							*			
S													*			
D		*														

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S1: Number of output pulses of X axis **S2:** Number of output pulses of Y axis **S:** Parameter setting **D:** Pulse output device

Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function.
2. The instruction only supports the pulse output type: Pulse / Direction.
3. **S_1** and **S_2** specify the number of output pulses (relative positioning) on X axis (Y0) and Y axis (Y2). Range: -2,147,483,648 ~ +2,147,483,647 (The "+/-" sign indicates the forward/backward direction). In forward direction, the present value of pulse output on CH0 (D1031 High, D1030 low), CH1 (D1337 high, D1336 low) increases. In reverse direction pulse output, value in (D1031, D1330) and (D1336, D1337) decreases.
4. The low word of **S** (settings of direction and resolution): K0 refers to clockwise 20-segment output; K1 refers to counterclockwise 20-segment output; A 90° arc can be drawn (see figure 1 and 2).
5. The high word of **S** (settings of motion time, unit: 0.1sec): Setting range: K2 ~ K200 (0.2 sec. ~ 20 secs.) This instruction is restricted by the maximum pulse output frequency; therefore when the set time is faster than the actual output time, the set time will be automatically modified.

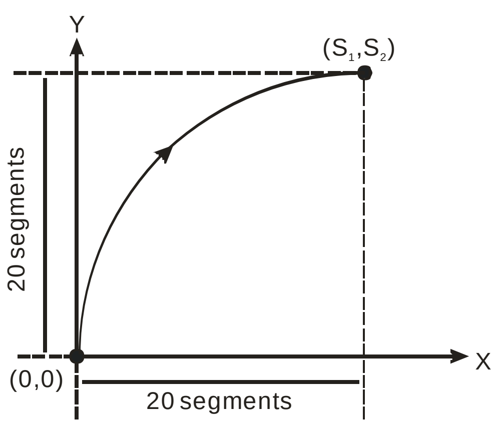


Figure 1

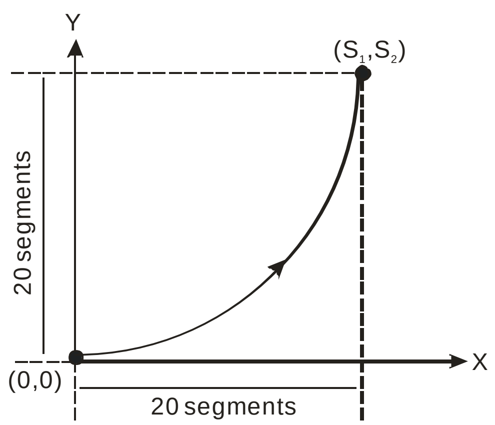


Figure 2

6. Draw four 90° arcs as the figure below.
   When the direction signal is ON, the direction is positive(QI, QIV). When the direction signal is OFF, the direction is negative(QII, QIII). When S is set as K0, the arcs will be clockwise (see figure 3). When S is set as K, the arcs will be counterclockwise (see figure 4).

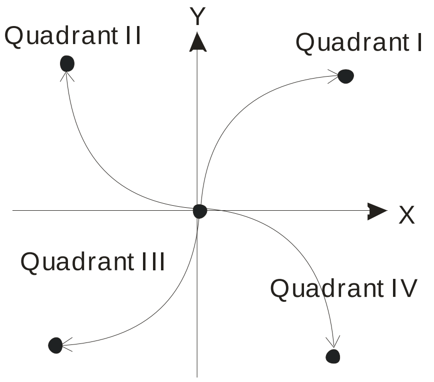


Figure 3

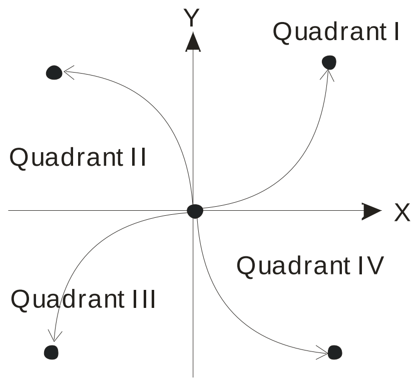


Figure 4

7. The settings of direction and resolution in the lower word of **S** can only be K0 ~ K1
8. The settings of motion time in the high word of **S** shall not be faster than the fastest suggested time. If the motion time is not specified, PLC will use the fastest suggested motion time as the setting. Refer to the table below.

Segments	Max. target position (pulse)	Fastest suggested set time (unit:100ms)
20-segments resolution	500 ~ 20,000	2
	20,000 ~ 29,999	3
	:	:
	Less than 10,000,000	Less than 200

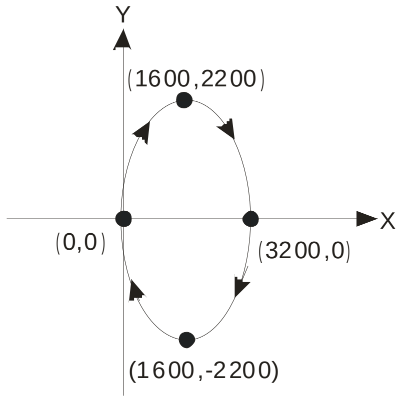
9. **D** can designate Y0 only.
   Y0 is the pulse output point of X axis;
   Y1 is the direction signal output of X axis.(OFF: positive; ON: negative)
   Y2 is the pulse output point of Y axis;
   Y3 is the direction signal output of Y axis (OFF: positive; ON: negative)
   When the pulse output is completed, the direction output signal will not be OFF unless the drive contact is OFF
10. When the 2-axis interpolation is being executed in 20 segments, it takes approximately 2ms for the initialization of this instruction. If only 1 axis is specified with pulse output number (with ramp-up/down section), i.e. another axis is 0, PLC will only execute single-axis positioning according to the specified motion time. If one of the two axes is specified with the pulse number less than 500, PLC will execute 2-axis linear interpolation automatically. However, when either axis is specified for pulse number over 10,000,000, the instruction will not work.
11. If the number of pulses which exceeds the above range is required, the user may adjust the gear ratio of the servo for obtaining the desired results.
12. Every time when the instruction is executed, only one 90° arc can be drawn. It is not necessary

that the arc has to be a 90° arc, i.e. the numbers of output pulses in X and Y axes can be different.

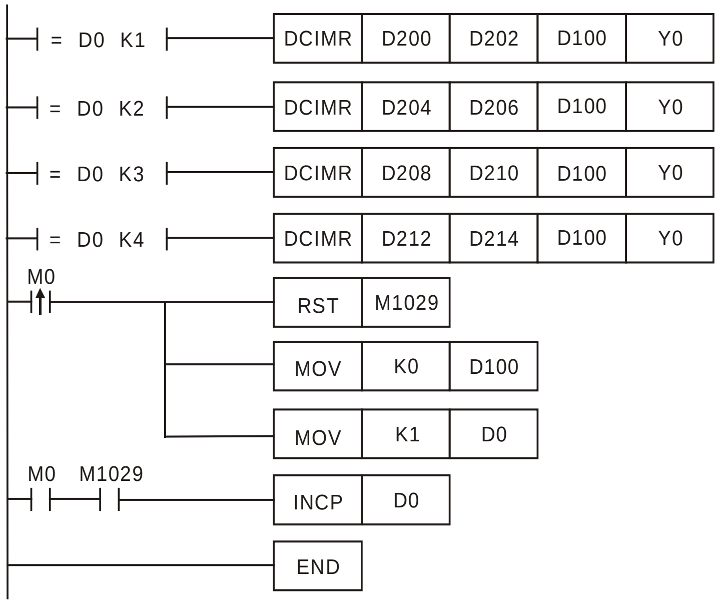
13. There are no settings of start frequency and ramp-up/down time.
14. There is no limitation on the number of times for using the instruction. However, assume CH0 or CH1 output is in use, the X/Y axis synchronized output will not be performed

Program Example 1:

1. Draw an ellipse as the figure below.



2. Steps:
   a) Set the four coordinates (0,0), (1600, 2200), (3200, 0), (1600, -2200) (as the figure above). Calculate the relative coordinates of the four points and obtain (1600, 2200), (1600, -2200), (-1600, -2200), and (-1600, 2200). Place them in the 32-bit registers (D200, D202), (D204, D206), (D208, D210), (D212, D214).
   b) Select "draw clockwise arc" and default "motion time" (**S** = D100 = K0).
   c) RUN the PLC. Set ON M0 to start the drawing of the ellipse.

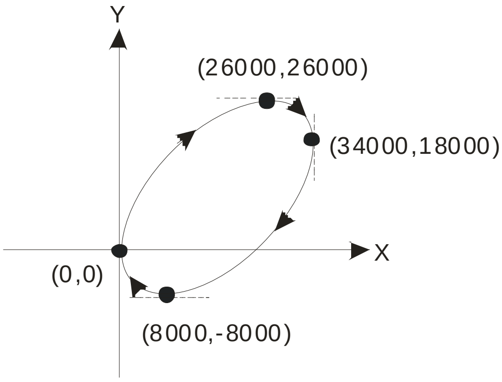


3. Operation:

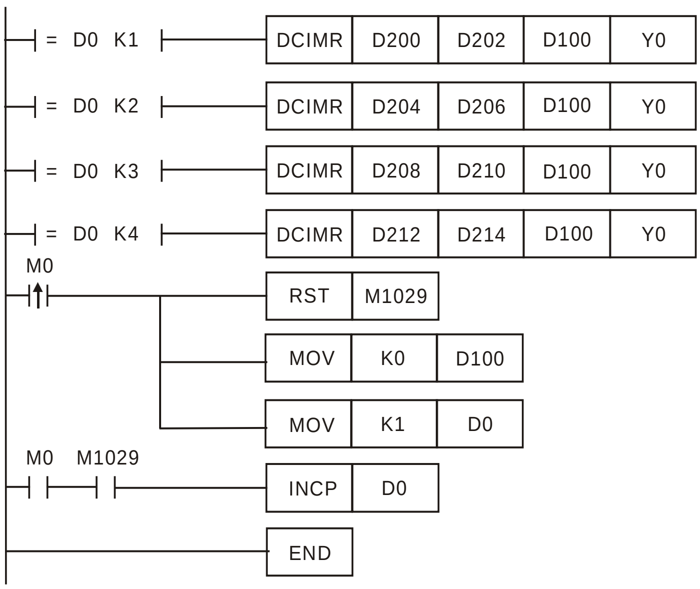
   When PLC runs and M0 = ON, PLC will start the drawing of the first segment of the arc. D0 will plus 1 whenever a segment of arc is completed and the second segment of the arc will start to execute automatically. The operation pattern repeats until the fourth segment of arc is completed.

Program Example 2:

1. Draw a tilted ellipse as the figure below.



2. Steps:
   a) Find the max. and min. coordinates on X and Y axes (0,0), (26000,26000), (34000,18000), (8000,-8000) (as the figure above). Calculate the relative coordinates of the four points and obtain (26000,26000), (8000,-8000), (-26000,-26000), (-8000,8000). Place them respectively in the 32-bit registers (D200,D202), (D204,D206), (D208,D210) and (D212,D214).
   b) Select “draw clockwise arc” and default “motion time” (**S** = D100 = K0).
   c) RUN the PLC. Set ON M0 to start the drawing of a tilted ellipse.



3. Operation:
   When PLC runs and M0 = ON, PLC will start the drawing of the first segment of the arc. D0 will plus 1 whenever a segment of arc is completed and the second segment of the arc will start to execute automatically. The operation pattern repeats until the fourth segment of arc is completed.

Points to note:

Description of associated flags and registers:

M1029: CH0 (Y0, Y1) pulse output execution completed

D1030: Present number of Y0 output pulses (HIGH WORD).

D1031: Present number of Y1 output pulses (LOW WORD).

D1336: Present value of Y2 pulse output. D1336 (High word)

D1337: Present value of Y2 pulse output. D1337(Low word)

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
194	D	CIMA	S1 S2 S D	2-Axis Absolute Position Arc Interpolation				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DCIMA: 17 steps
S_1					*	*							*			
S_2					*	*							*			
S													*			
D		*														

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

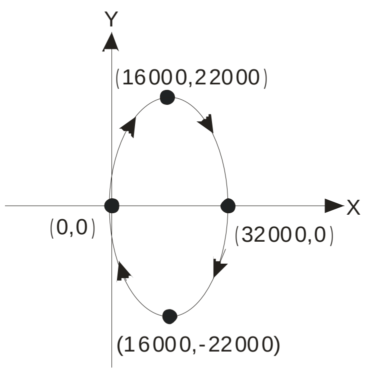
S1: Number of output pulses of X axis **S2:** Number of output pulses of Y axis **S:** Parameter setting **D:** Pulse output device

Explanations:

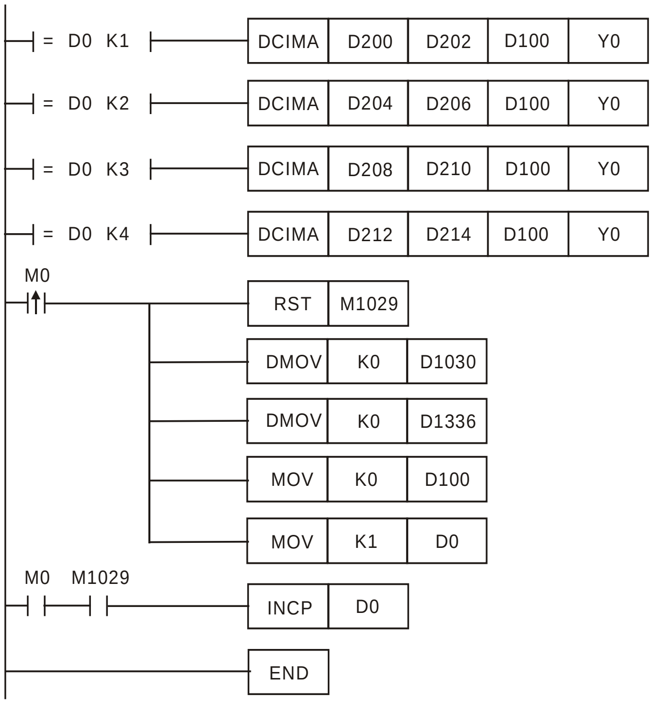
1. For ES2/EX2 models, only V1.20 or above supports the function.
2. The instruction only supports the pulse output type: Pulse / Direction.
3. **S_1** and **S_2** specify the number of output pulses (absolute positioning) on X axis (Y0) and Y axis (Y2). Range: -2,147,483,648 ~ +2,147,483,647. When **S_1** and **S_2** are bigger than PV of pulse output in CH0 (D1031 High, D1030 low) / CH1 (D1337 high, D1336 low), pulse output will operate in positive direction and the direction signal output Y1, Y3 will be OFF. When **S_1** and **S_2** are smaller than PV of pulse output, pulse output will operate in negative direction and the direction signal output Y1, Y3 will be ON.
4. For the rest of the explanations on the instruction, special D and special M, please refer to API 193 DCIMR instruction.

Program Example 1:

1. Draw an ellipse as the figure below.



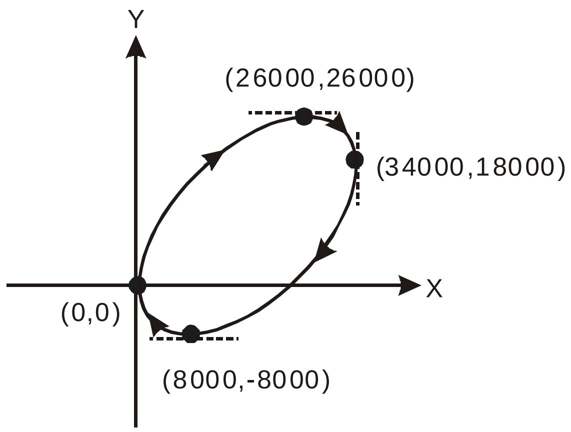
2. Steps:
   a) Set the four coordinates (0,0), (16000, 22000), (32000, 0), (16000, -22000) (as the figure above). Place them in the 32-bit registers (D200, D202), (D204, D206), (D208, D210), (D212, D214).
   b) Select “draw clockwise arc” and default “motion time” (**S** = D100 = K0)
   c) RUN the PLC. Set ON M0 to start the drawing of the ellipse.



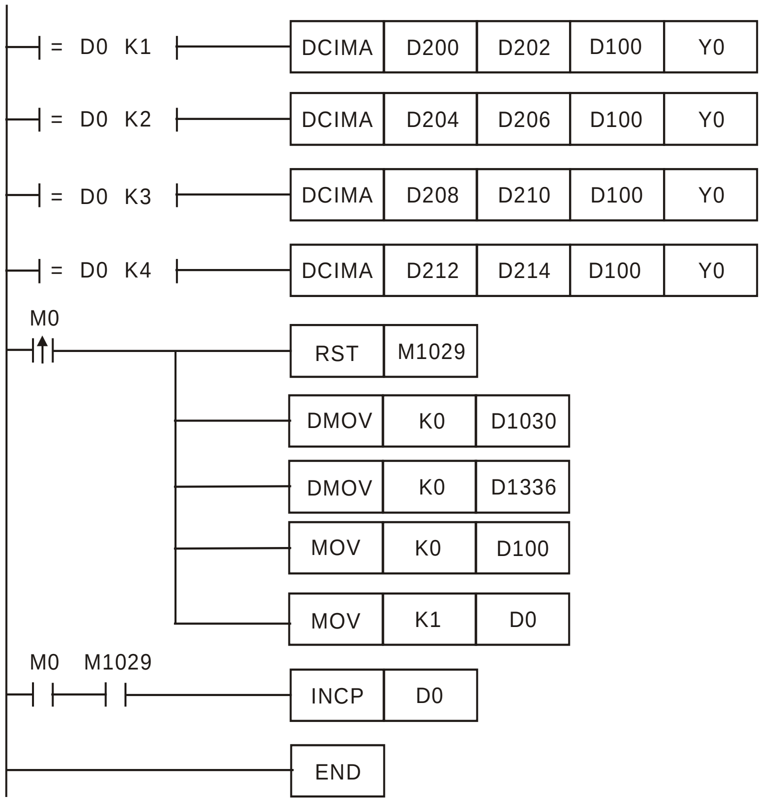
3. Operation:
   When PLC runs and M0 = ON, PLC will start the drawing of the first segment of the arc. D0 will plus 1 whenever a segment of arc is completed and the second segment of the arc will start to execute automatically. The operation pattern repeats until the fourth segment of arc is completed.

Program Example 2:

1. Draw a tilted ellipse as the figure below.



2. Steps:
   a) Find the max. and min. coordinates on X and Y axes (0,0), (26000,26000), (34000,18000), (8000,-8000) (as the figure above). Place them respectively in the 32-bit registers (D200,D202), (D204,D206), (D208,D210) and (D212,D214).
   b) Select “draw clockwise arc” and default “motion time” (**S** = D100 = K0).
   c) RUN the PLC. Set ON M0 to start the drawing of a tilted ellipse.



3. Operation:

   When PLC runs and M0 = ON, PLC will start the drawing of the first segment of the arc. D0 will plus 1 whenever a segment of arc is completed and the second segment of the arc will start to execute automatically. The operation pattern repeats until the fourth segment of arc is completed.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
195	D	PTPO	(S1) (S2) (D)	Single-axis pulse output by table				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
S_1													*			DPTPO: 13 steps
S_2													*			
D		*														

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Source start device **S_2**: Number of segments **D**: Pulse output device

Explanations:

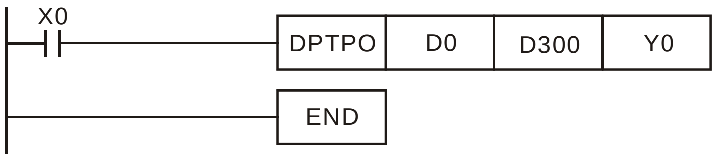
1. **S_1** specifies the output frequency and the number of pulses according to the number of segments set by **S_2**. Each segment occupies consecutive 4 registers, i.e. (**S_1**+0), (**S_1**+1), (**S_1**+2) and (**S_1**+3). (**S_1**+0) and (**S_1**+1) stores the output frequency; (**S_1**+2) and (**S_1**+3) stores the number of output pulses.
2. Available output frequency for **S_1** : 6Hz~100,000Hz.
3. **S_2** + 0: total number of segments (range: 1 ~ 40). **S_2** + 1: The No. of current executing segment. The number in **S_2** + 1 will be updated when the PLC scan reaches this instruction.
4. **D** can only be designated with output devices Y0 and Y2, i.e. only pulse output is supported. Users need to apply other instructions if a control on direction signal output is required.
5. This instruction does not offer ramp up/down function. Therefore, when the instruction is disabled, the output pulses will stop immediately.
6. There is no limitation on the times of using this instruction, however during each scan cycle, Y0 and Y2 can be driven by one instruction at a time.
7. When the instruction is being executed, changes to the instruction parameter will be invalid.
8. Cyclic output can be performed on this instruction by driving ON M1262.

Program Example:

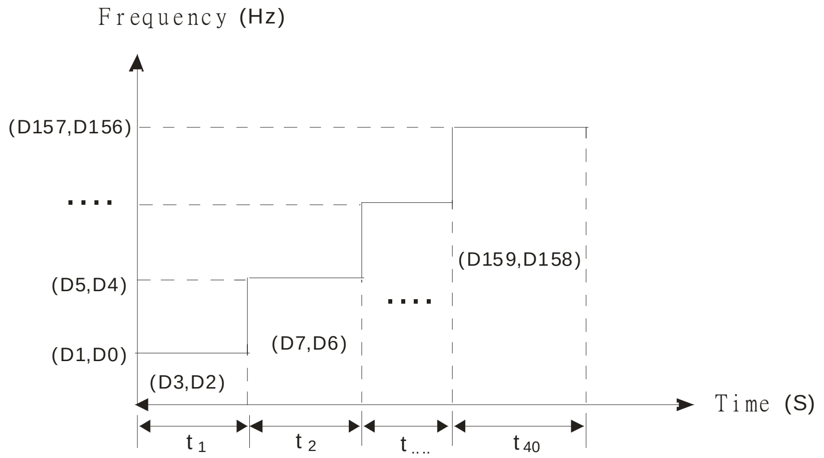
1. When X0 = ON, pulse output will be operated according to the set frequency and number of pulses in every segment.
2. Format of the table:

S_2 = D300, number of segments (D300 = K40)	**S_1** = D0, frequency (**S_1** + 0)	**S_1** = D0, number of output pulses (**S_1** + 2)
K1 (1^{st} segment)	D1, D0	D3, D2
K2 (2^{nd} segment)	D5, D4	D7, D6
:	:	:
K40 (40^{th} segment)	D157, D156	D159, D158

3. Current executing segment can be monitored by D301.



4. Timing diagram:



Points to note:

1. Associated Flags:

   M1029: CH0 (Y0) pulse output execution completed.

   M1102: CH1 (Y2) pulse output execution completed

   M1078: CH0 (Y0) pulse output pause (immediate)

   M1104: CH1 (Y2) pulse output pause (immediate)

   M1262: Enable cyclic output for table output function of DPTPO instruction. ON = enable.

   M1538: Indicating pause status of Y0

   M1540: Indicating pause status of Y2

2. Special registers:

   D1030: Low word of the present value of Y0 pulse output

   D1031: High word of the present value of Y0 pulse output

   D1336: Low word of the present value of Y2 pulse output

   D1337: High word of the present value of Y2 pulse output

API	Mnemonic		Operands	Function	Controllers			
197	D	CLLM	(S1) (S2) (S3) (D)	Close loop position control	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1	*											*				DCLLM: 17 steps
S_2					*	*							*			
S_3					*	*							*			
D		*														

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Feedback source device　　**S_2**: Target number of feedbacks　　**S_3**: Target frequency of output

D: Pulse output device

Explanations:

1. The corresponding interrupt pointers of **S_1**:

Source device	X4	X6	C243 ~ C254	
Associted outout	Y0	Y2	Y0	Y2
No. of Interrupt pointer	I40□	I60□	I010	I050

☐ = 1: rising-edige triggered; ☐ = 0: falling-edge triggered

a) When **S_1** designates input points X and the pulse output reaches the target number of feedbacks in **S_2**, the output will continue to operate by the frequency of the last shift (end frequency) until interrupts occur on input points X.

b) When **S_1** designates high speed counters and the pulse output reaches the target number of feedbacks in **S_2**, the output will continue to operate by the frequency of the last shift (end frequency) until the feedback pulses reaches the target number.

c) **S_1** can be a high speed counter C or an input point X with external interrupt. If **S_1** is C, DCNT instruction should be executed in advance to enable the high-speed counting function, and EI instruction with I0x0 should be enabled for external interrupts. If **S_1** is X, EI instruction with I0x0 should be enabled for external interrupts.

d) If **S_1** is specifed with counters, DHSCS instruction has to be programmed in user program. Please refer to **Program example 2** for details.

2. Range of **S_2**: -2,147,483,648 ~ +2,147,483,647 (+ / - indicates the positive / negative rotation direction). the present value of pulse output in CH0 (Y0, Y1) and CH1 (Y2, Y3) increases in positive direction and decreases in negative direction. Registers storing present value of pulse output: CH0(D1031 High, 1030 Low), CH1(D1337 High, D1336 Low)

3. If **S_3** is lower than 6Hz, the output will operate at 6Hz; if **S_3** is higher than 100kHz, the output will operate at 100kHz.

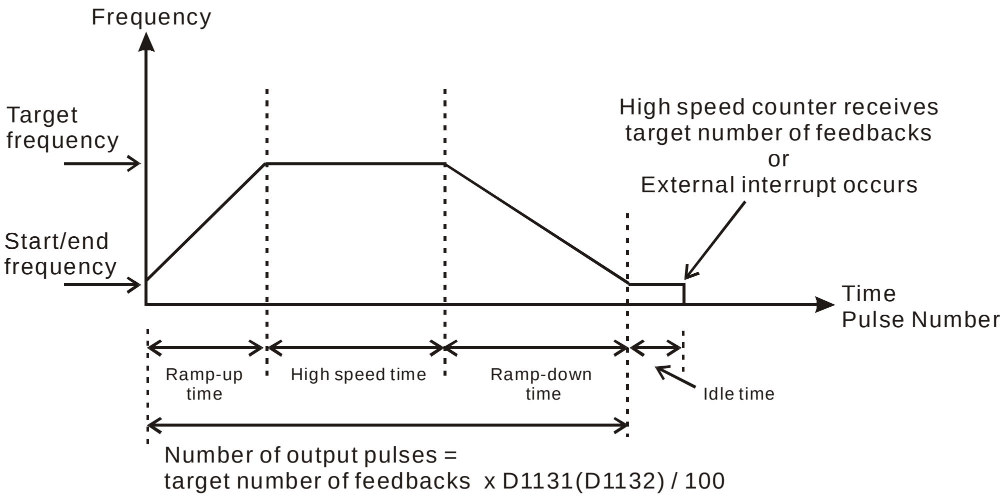
4. **D** can only designate Y0 (Direction signal output: Y1) or Y2 (Direction signal output: Y3). The

direction signal output will be OFF only when the drive contact of the instruction is OFF, i.e. completion of pulse output will not reset Y1 or Y3.

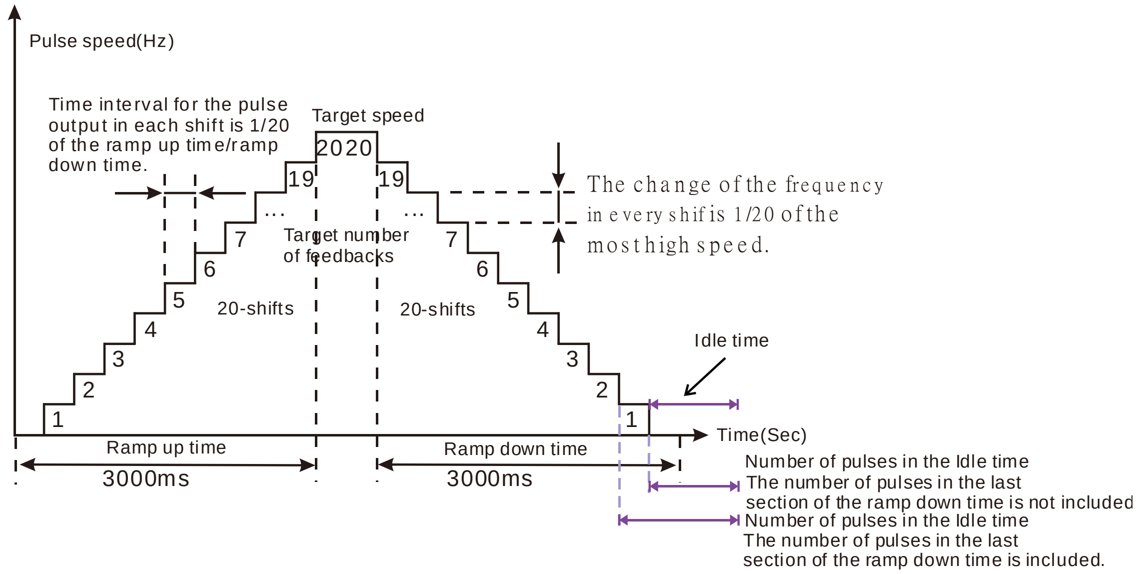
5. D1340 and D1352 stores the start/end frequencies of CH0 and CH1. Min. 6Hz, default: 100Hz.
6. D1343 and D1353 stores the ramp up/down time of CH0 and CH1. If the ramp up/down time is shorter than 20ms, PLC will operate in 20ms. Dafault: 100ms.
7. Ramp-down time of CH0 and CH1 can be particularlily specified by the setting of (M1534, D1348) and (M1535, D1349). When M1534 / M1535 is ON, ramp-down time of CH0 and CH1 is set by D1348 and D1349.
8. D1131 and D1132 are the output/input ratio(%) of the close loop control in CH0 and CH1. K1 refers to 1 output pulse out of 100 feedback pulses; K200 refers to 200 output pulses out of the 100 feedback pulses. In general percentage equation, the value set in D1131 and D1132 represents numerators (output pulses, available range: K1 ~ K10,000) and the denominator (the input feedbacks) is fixed as K100 (System defined).
9. M1305 and M1306 can reverse the direction of CH0, CH1 pulse output. For example, when direction signal output (Y1/Y3) is OFF, pulse output will operate in positive direction. If M1305/M1306 is set ON before the execution of this instruction, the pulse output will be reversed as negative output direction.
10. When $\mathbf{S_1}$ designates input points X with interrupt pointers, D1244 / D1255 can be applied for setting the idle time as limited pulse number, in case the interrupt is not properly triggered.
11. DCLLM instruction supports Alignment Mark and Mask function. Please refer to **PLSR** instruction for details.

Close Loop Explanations:

1. Function: Immediately stop the high-speed pulse output according to the number of feedback pulses or external interruption signals.
2. Timing diagram:



3. Description of the number of output pulses in the idle time:



ES2/EX2 V3.28 (and below), SA2/SX2 V2.82 (and below), and SS2/SE:

The nunmber of ouput pulses in the idle time in D1244/D4245 includes the numbers of pulses in the last section of the ramp down time. If the target number of feedbacks is 50000, the number of output pulses in the idle time is 1000, the number of pulses in the laste section of the ramp down time is 50, and no external interrupt occurs, the total number of pulses will be 50665 (50000+100-50).

ES2/EX2 V3.40 (and above), and SA2/SX2 V2.84 (and above):

The nunmber of ouput pulses in the idle time in D1244/D4245 does not include the numbers of pulses in the last section of the ramp down time. If the target number of feedbacks is 50000, the number of output pulses in the idle time is 1000, the number of pulses in the laste section of the ramp down time is 50, and no external interrupt occurs, the total number of pulses will be 51000 (50000+100).

4. Principles for adjusting the completion time of positioning:

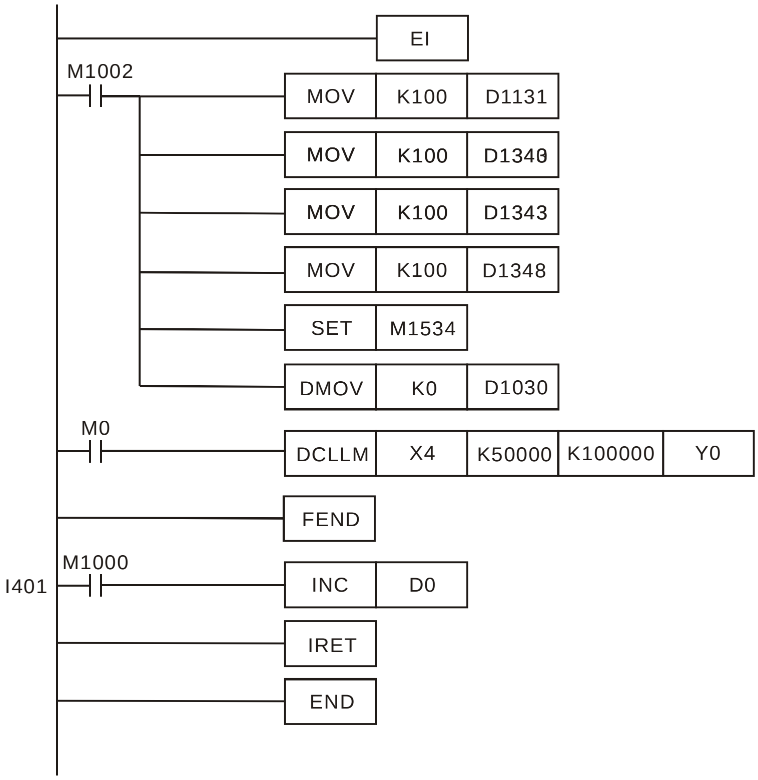
   a) The completion time of positioning refers to the total time of “ramp up + high speed + ramp down + idle” (see the figure above). When percentage value (D1131/D1132) is modified, the total of output pulses will be increased or decreased as well as the completion time.

   b) When **S₁** designates input points X with interrupt pointers, D1244 / D1255 can be applied for setting the idle time as limited pulse number, in case the interrupt is not properly triggered.Users can determine if the execution result is good or bad by the length of the idling time. In theory, a bit of idling left is the best result for a positioning.

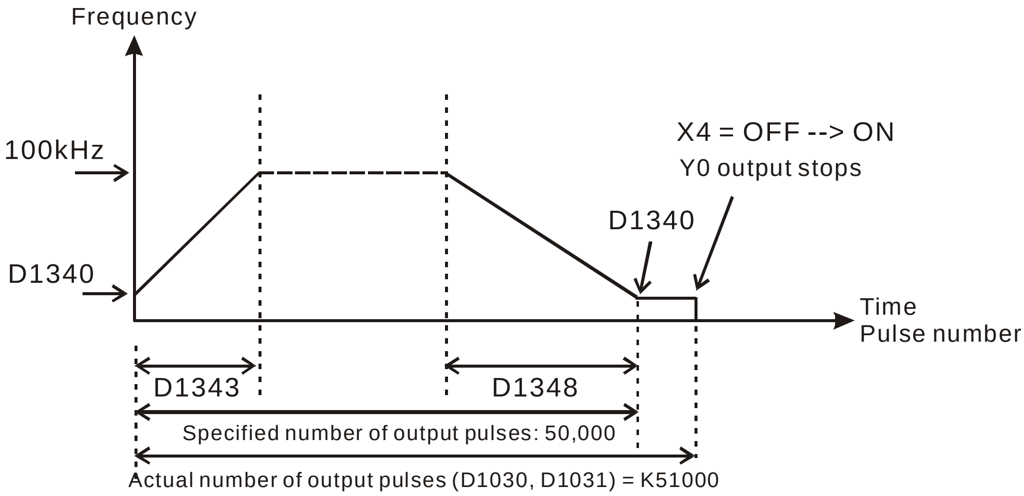
   c) Owing to the close loop operation, the length of idle time will not be the same in every execution. Therefore, when the content in the special D for displaying the actial number of output pulses is smaller or larger than the calculated number of output pulses (target number of feedbacks x percentage value / 100), users can improve the situation by adjusting the percentage value, ramp-up/ramp-down time or target frequency.

Program Example1: Immediate stop high-speed pulse output by external interrupt

1. Adopt X4 as the input for external interrupt and I401 (rising-edge trigger) as the interrupt pointer. Set target number of feedbacks = 50,000; target frequency = 100kHz; pulse output device: Y0, Y1 (CH0); start/end frequency (D1340) = 100Hz; ramp-up time (D1343) = 100ms; ramp-down time (D1348) = 100ms; percentage value (D1131) = 100; present value of output pulses (D1030, D1031) = 0.

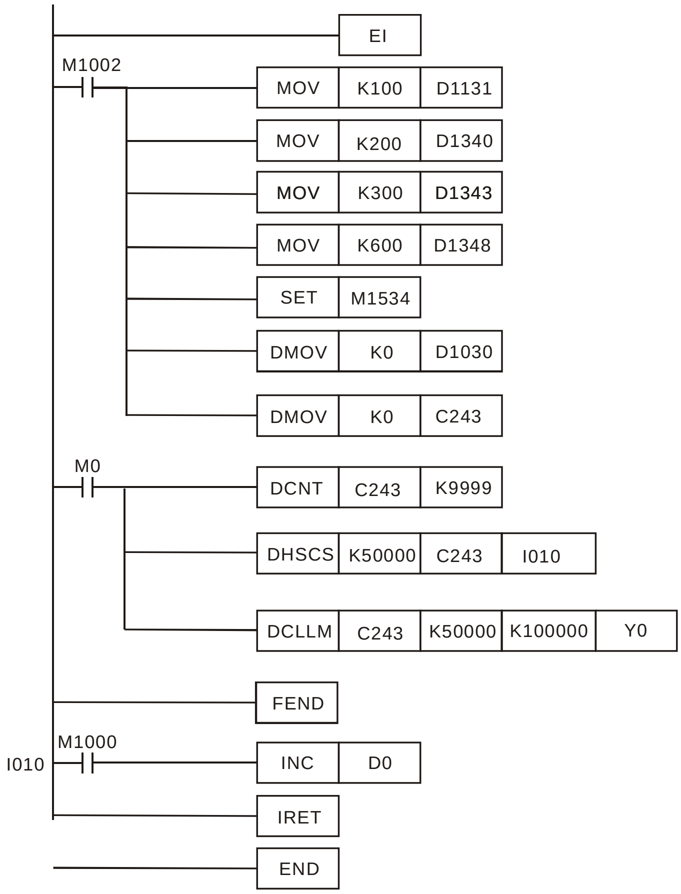


Execution results:

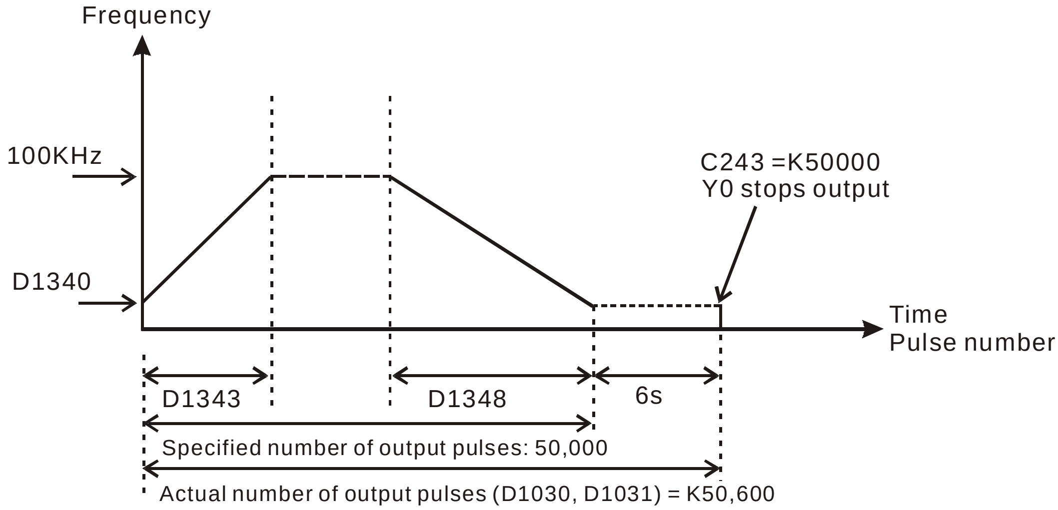


Program Example 2: Immediate stop high-speed pulse output by high speed counter

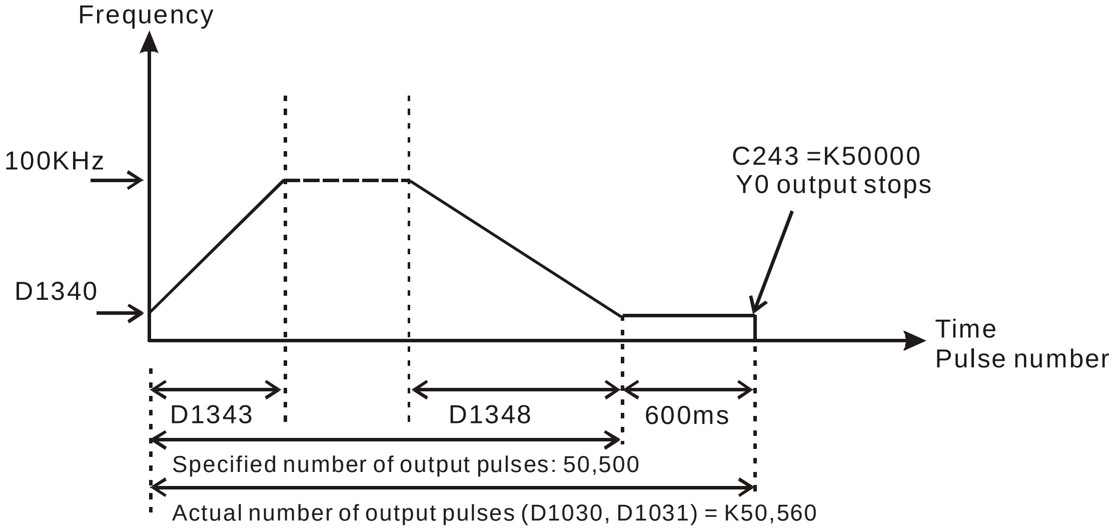
1. Adopt counter C243 (better to be reset before execution) with AB-phase input from the encoder. Set target number of feedbacks = 50,000; target frequency = 100kHz; pulse output device: Y0, Y1 (CH0); start/end frequency (D1340) = 200Hz; ramp-up time (D1343) = 300ms; ramp-down time (D1348) = 600ms; percentage value (D1131) = 100; present value of output pulses (D1030, D1031) = 0.



2. Assume the first execution results are as below:



3. Observe the results of the first execution:
   a) The actual output number 50,600 – specified output number 50,000 = 600
   b) 600 x (1/100Hz) = 6s (idle time)
   c) 3 seconds are too long. Therefore, increase the percentage value (D1131) to K101.
4. Obatin the results of the second execution:



5. Observe the results of the second execution:
   a) The actual output number 50,560 – specified output number 50,500 = 60
   b) 60 x (1/100Hz) = 600ms (idle time)
   c) 600ms is an appropriate value. Therefore, set the percentage value (D1131) as K101 to complete the design.

Points to note:

1. Associated flags:

M1029: CH0 (Y0, Y1) pulse output execution completed.

M1102: CH1 (Y2, Y3) pulse output execution completed.

M1078: M1078 = ON, CH0 (Y0, Y1) pulse output pause (immediate)

M1104: M1104 = ON CH1 (Y2, Y3) pulse output pause (immediate)

M1108: CH0 (Y0, Y1) pulse output pause (ramp down). M1108 = ON during ramp down.

M1110: CH1 (Y2, Y3) pulse output pause (ramp down). M1110 = ON during ramp down.

M1156: Enabling the mask and alignment mark function on I400/I401(X4) corresponding to Y0.

M1158: Enabling the mask and alignment mark function on I600/I601(X6) corresponding to Y2.

M1538: Indicating pause status of CH0 (Y0, Y1).M1538 = ON when output paused.

M1540: Indicating pause status of CH1 (Y2, Y3). M1540 = ON when output paused

M1305: Reverse CH0 (Y0, Y1) pulse output direction. M1305 = ON, pulse output direction is reversed.

M1306: Reverse CH1 (Y2, Y3) pulse output direction. M1306 = ON, pulse output direction is reversed

M1347: Auto-reset CH0 (Y0, Y1) when high speed pulse output completed. M1347 will be reset after CH0 (Y0, Y1) pulse output is completed.

M1524: Auto-reset CH1 (Y2, Y3) when high speed pulse output completed. M524 will be reset after CH1 (Y2, Y3) pulse output is completed.

M1534: Enable ramp-down time setting on Y0. Has to be used with D1348

M1535: Enable ramp-down time setting on Y2. Has to be used with D1349

2. Special registers:

D1026: Pulse number for masking Y0 when M1156 = ON (Low word). The function is disabled when set value≦0. (Default = 0)

D1027: Pulse number for masking Y0 when M1156 = ON (High word). The function is disabled when set value≦0. (Default = 0)

D1135: Pulse number for masking Y2 when M1156 = ON (Low word). The function is disabled when set value≦0. (Default = 0)

D1136: Pulse number for masking Y2 when M1156 = ON (High word). The function is disabled when set value≦0. (Default = 0)

D1030: Low word of the present value of CH0 (Y0, Y1) pulse output

D1031: High word of the present value of CH0 (Y0, Y1) pulse output

D1131: Input/output percentage value of CH0 (Y0, Y1) close loop control. Default: K100

D1132: Input/output percentage value of CH1 (Y2, Y3) close loop control. Default: K100

D1244: Idle time (pulse number) setting of CH0 (Y0, Y1) The function is disabled if set value≦0.

D1245: Idle time (pulse number) setting of CH2 (Y2, Y3) The function is disabled if set value≦0.

D1336: Low word of the present value of CH1 (Y2, Y3) pulse output

D1337: High word of the present value of CH1 (Y2, Y3) pulse output

D1340: Start/end frequency of the 1st group pulse output CH0 (Y0, Y1). Default: K100

D1352: Start/end frequency of the 2st group pulse output CH1 (Y2, Y3). Default: K100

D1343: Ramp up/down time of the 1st group pulse output CH0 (Y0, Y1). Default: K100

D1353: Ramp up/down time of the 2nd group pulse output CH1 (Y2, Y3). Default: K100

D1348: CH0(Y0, Y1) pulse output. When M1534 = ON, D1348 stores the ramp-down time. Default: K100

D1349: (a) CH1(Y2, Y3) pulse output. When M1535 = ON, D1349 stores the ramp-down time. Default: K100

API	Mnemonic			Operands	Function	Controllers
198	D	VSPO		(S1) (S2) (S3) (D)	Variable speed pulse output	ES2/EX2, SS2, SA2 SE, SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																DVSPO: 17 steps
S_1													*			
S_2					*	*							*			
S_3					*	*							*			
D		*														

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

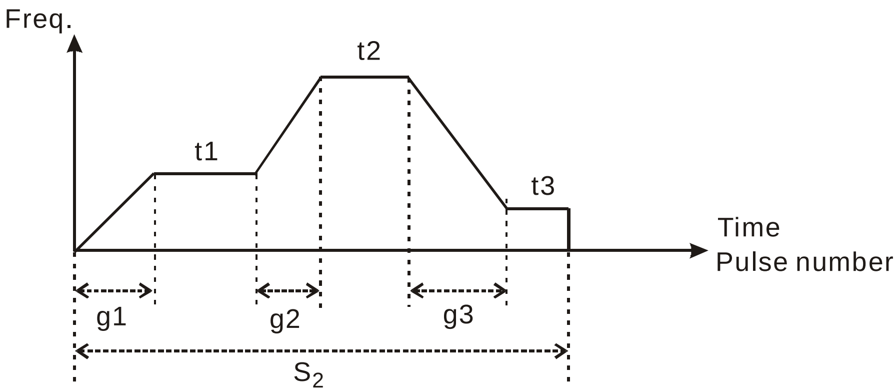
S_1: Target frequency of output **S_2**: Target number of pulses **S_3**: Gap time and gap frequency **D**: Pulse output device (Y0, Y2)

Explanations:

1. Max frequency for **S_1**: 100kHz. Target frequency can be modified during the execution of instruction. When **S_1** is modified, VSPO will ramp up/down to the target frequency according to the ramp-up gap time and gap frequency set in **S_3**.
2. **S_2** target number of pulses is valid only when the instruction is executed first time. **S_2** can NOT be modified during the execution of instruction. **S_2** can be a negative value, however, if the output direction is not specified in D1220/D1221, PLC will take this value as a positive value. When target number of pulses are specified with 0, PLC will perform continuous output.
3. **S_3** occupies 2 consecutive 16-bit devices. **S_3**+0 stores the gap frequency **S_3**+1 stores the gap time. Parameter setting can be modified during the execution of instruction. Set range for **S_3**+0: 1Hz ~ 32767Hz; set range for **S_3**+0: 1ms ~ 100ms. (for SE series, the set range for **S_3**+0 is 1 ~ 40ms) If set value exceeds the available range, PLC will take the upper or lower bound value.
4. **D** pulse output device supports only Y0 and Y2. If Y1 and Y3 is required for output direction control, D1220 or D1221 has tobe set as K1(Pulse/Dir).
5. Parameters set in **S_3** can only be modified while modifying the value in **S_1**. When target frequency is set as 0, PLC will ramp down to stop according to parameters set in **S_3**. When the output is stopped, PLC will enable the flags indicating pause status (Y0: M1538, Y2: M1540). If target frequency other than 0 is specified again, pulse output will ramp up to target frequency and operates untill target number of pulses are completed.

Function Explanations:

Pulse output diagram:



1. Definitions:

   t1 → target frequency of 1st shift

   t2 → target frequency of 2nd shift

   t3 → target frequency of 3rd shift

   g1 → ramp-up time of 1st shift

   g2 → ramp-up time of 2nd shift

   g3 → ramp-down time of 3rd shift

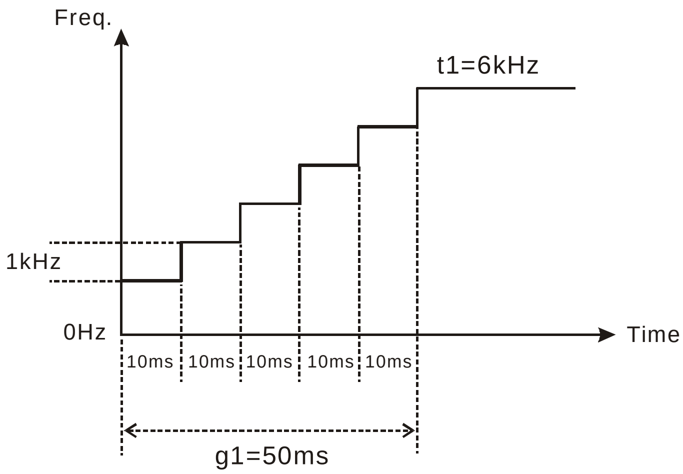
   S_2 → total output pulses

2. Explanations on each shift:

   ◆ 1st shift:

   Assume t1 = 6kHz, gap freqency = 1kHz, gap time = 10ms

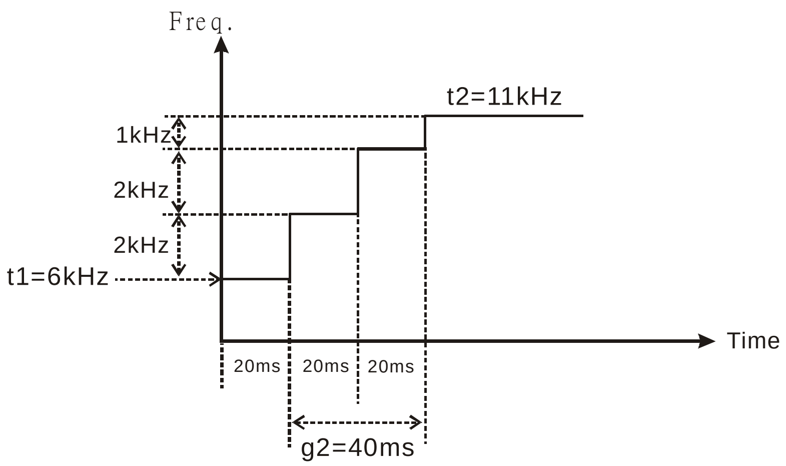
   Ramp-up steps of 1st shift:



◆ 2nd shift:

Assume t2 = 11kHz, internal frequency = 2kHz, gap time = 20ms

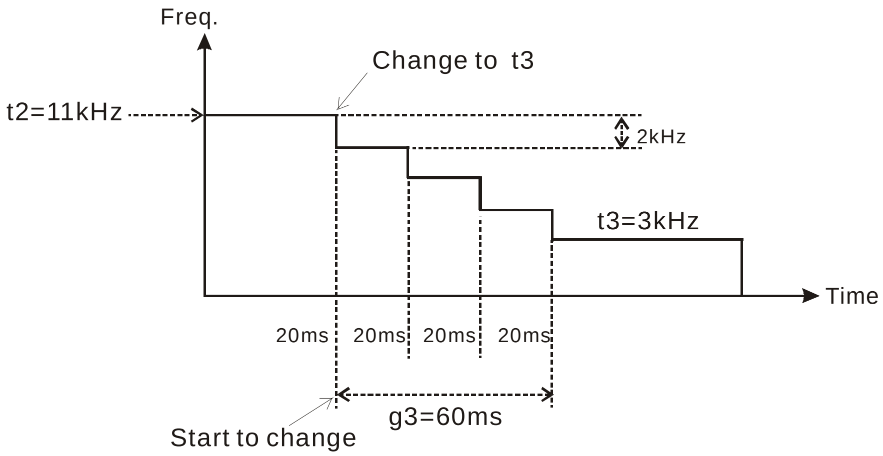
Ramp-up steps of 2nd shift:



◆ 3rd shift:

Assume t3 = 3kHz, gap frequency = 2kHz, gap time = 20ms

Ramp-down steps of 3rd shift:



◆ For program examples please refer to API 199

Points to note:

1. Associated flags:

M1029	CH0 (Y0, Y1) pulse output execution completed
M1102	CH1 (Y2, Y3) pulse output execution completed
M1078	Y0 pulse output pause (immediate)
M1104	Y2 pulse output pause (immediate)
M1305	Reverse Y1 pulse output direction in high speed pulse output instructions

M1306	Reverse Y3 pulse output direction in high speed pulse output instructions
M1538	Indicating pause status of Y0
M1540	Indicating pause status of Y2

2. Special register explanations:

D1030	Low word of the present value of Y0 pulse output
D1031	High word of the present value of Y0 pulse output
D1336	Low word of the present value of Y2 pulse output
D1337	High word of the present value of Y2 pulse output
D1220	Pulse output mode setting of CH0 (Y0, Y1). Please refer to PLSY instruction.
D1221	Pulse output mode setting of CH1 (Y2, Y3). Please refer to PLSY instruction

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
199	D	ICF	(S1) (S2) (D)	Immediately change frequency				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DVSPO: 13 steps
S_1													*			
S_2					*	*							*			
D		*														

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

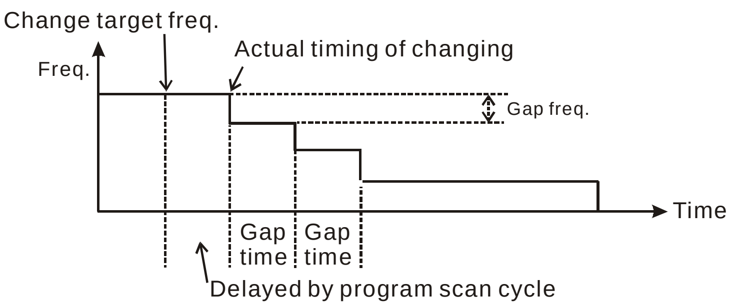
S_1: Target frequency to be changed **S_2**: Gap time and gap frequency **D**: Pulse output device (Y0, Y2)

Explanations:

1. Max frequency for **S_1**: 100kHz. When ICF instruction executes, frequecy changing will start immediately with ramp-up/down process.
2. ICF instruction has to be executed after the execution of DVSPO or DPLSY instructions. When the instruction is used together with DVSPO, operands **S_1**, **S_2**, **D** of DICF has to be assigned the same device with **S_1**, **S_3**, **D** of DVSPO. When the instruction is used with DPLSY, operands **S_1** and **D** has to be assigned the same device with **S_1** and **D** of DPLSY**.**
3. If ICF instruction is used with DPLSY instruction, operand **S_2** is invalid.
4. When ICF instruction is used with DVSPO instruction, parameter setting of **S_2** functions the same as **S_3** in DVSPO instruction, specifying the gap time and gap frequency of ramp-up/down process.
5. **D** pulse output device supports only Y0 and Y2.
6. The instruction is suggested to be applied in interrupt subroutines for obtaining the better response time and eexecution results
7. For associated flags and registers, please refer to **Points to note** of API 198 DVSPO instruction.

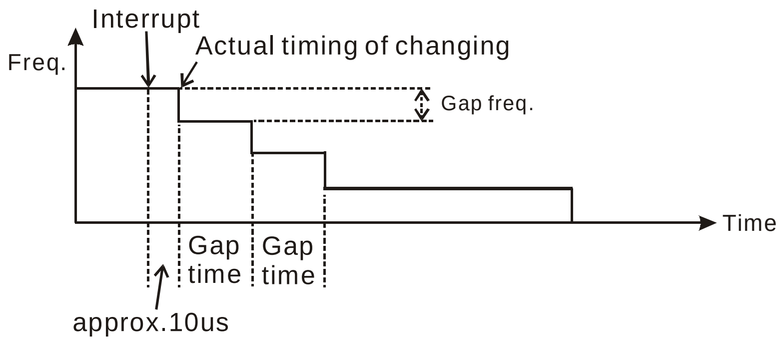
Function Explanations:

1. If users change the target frequency by using DVSPO instruction, the actual changing timing will be delayed due to the program scan time and the gap time as below.



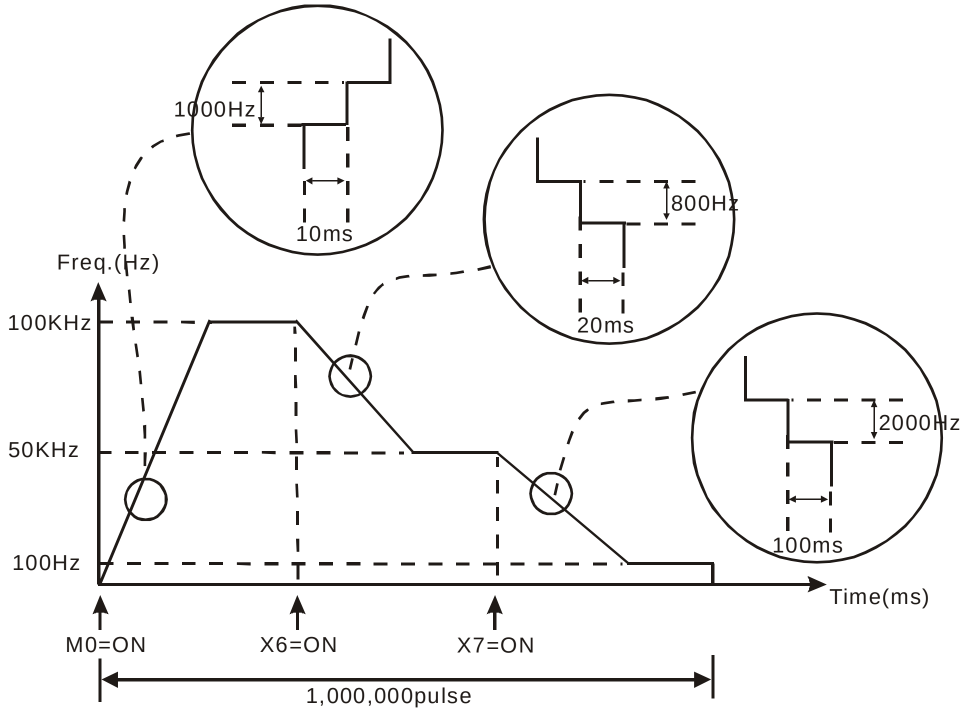
2. If users change the target frequency by applying DICF instruction in insterupt subroutines, the actual changing timing will be executed immediately with only an approx. 10us delay (execution time of DICF instruction).

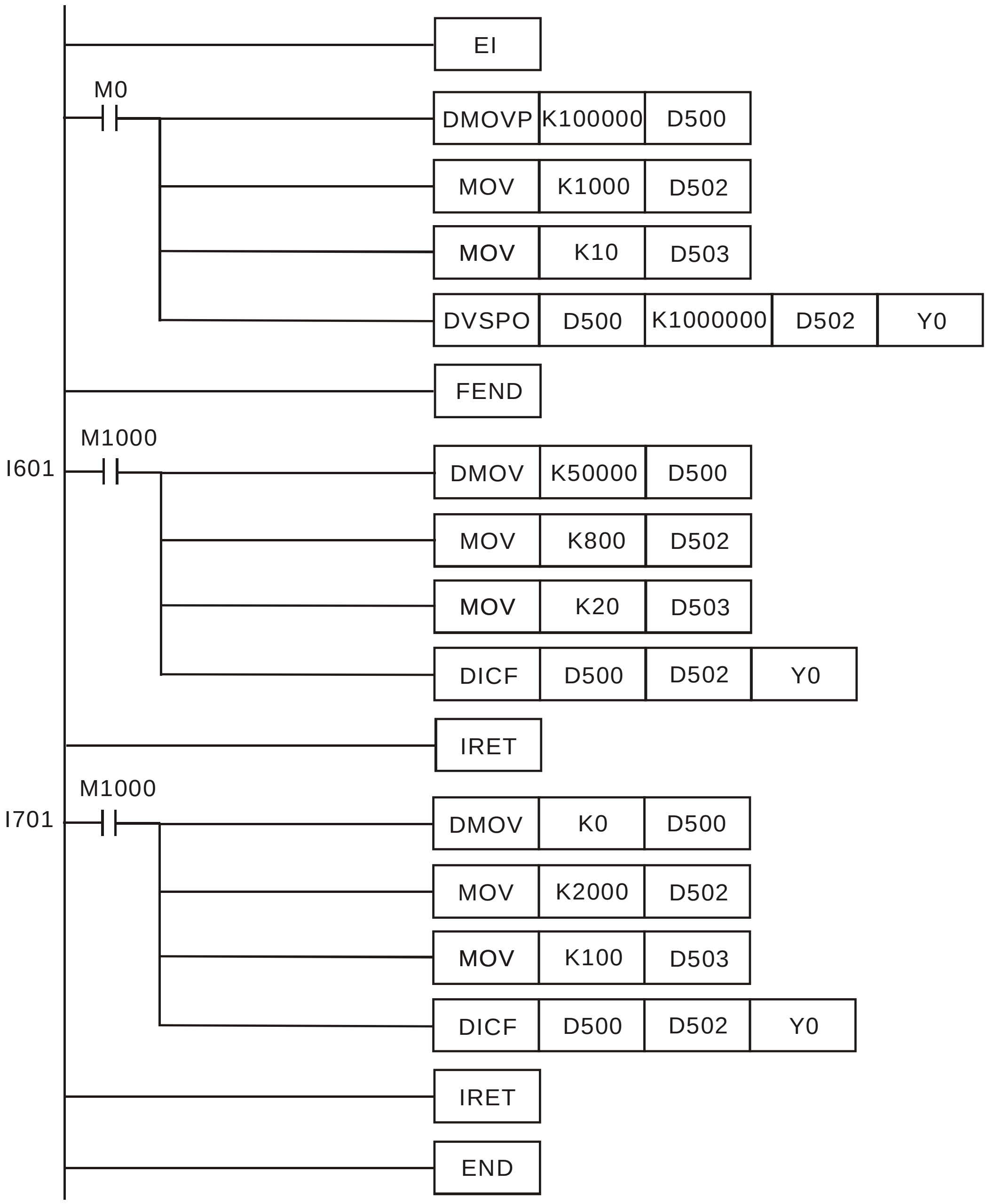
The timing diagram is as below:



Program Example:

1. When M0 = ON, pulse output ramps up to 100kHz. Total shifts: 100, Gap frequency: 1000Hz, Gap time: 10ms. Calculation of total shifts: (100,000 － 0) ÷ 1000 = 100.
2. When X6 external interrupt executes, target frequency is changed and ramp down to 50kHz immediately. Total shifts: 150, Gap frequency: 800Hz, Gap time: 20ms. Calculation of total shifts: (100,000 － 50,000) ÷ 800 = 125
3. When X7 external interrupt executes, target frequency is changed and ramp down to 100Hz immediately. Total shifts: 25, Gap frequency: 2000Hz, Gap time: 100ms. Calculation of total shifts: (50,000 － 100) ÷ 2000 = 25.
4. When pulse output reaches 100Hz, the frequency is kept constant and pulse output stops when 1,000,000 pulses is completed.





EI
M0
DMOVP
K100000
D500
MOV
K1000
D502
MOV
K10
D503
DVSPO
D500
K1000000
D502
Y0
FEND
I601
M1000
DMOV
K50000
D500
MOV
K800
D502
MOV
K20
D503
DICF
D500
D502
Y0
IRET
I701
M1000
DMOV
K0
D500
MOV
K2000
D502
MOV
K100
D503
DICF
D500
D502
Y0
IRET
END

3.6.15 Real Time Calendar

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
160	TCMP	-	✓	Time compare	✓	✓	✓	✓	11	-
161	TZCP	-	✓	Time Zone Compare	✓	✓	✓	✓	9	-
162	TADD	-	✓	Time addition	✓	✓	✓	✓	7	-
163	TSUB	-	✓	Time subtraction	✓	✓	✓	✓	7	-
166	TRD	-	✓	Time read	✓	✓	✓	✓	3	-
167	TWR	-	✓	Time write	✓	✓	✓	✓	3	-
169	HOUR	DHOUR	-	Hour meter	✓	✓	✓	✓	7	13

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
160	TCMP	P	(S1) (S2) (S3) (S) (D)	Time compare				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TCMP, TCMPP: 11 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	
S_2					*	*	*	*	*	*	*	*	*	*	*	
S_3					*	*	*	*	*	*	*	*	*	*	*	
S											*	*	*			
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

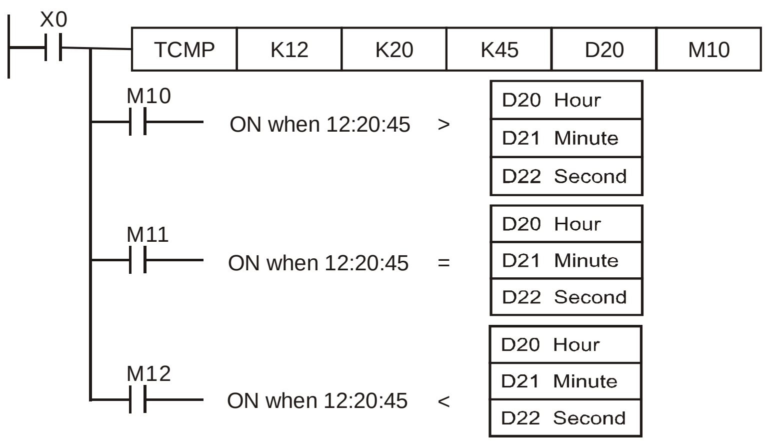
S_1: "Hour" for comparison (K0~K23) **S_2**: "Minute" for comparison (K0~K59) **S_3**: "Second" for comparison (K0~K59) **S**: Current time of RTC (occupies 3 consecutive devices) **D**: Comparison result (occupies 3 consecutive devices)

Explanations:

1. TCMP instruction compares the time set in **S_1**, **S_2**, **S_3** with RTC current value in **S** and stores the comparison result in **D**.
2. **S:** "Hour" of current time of RTC. Content: K0~K23. **S** +1: "Minute" of current time of RTC. Content: K0~K59. **S** +2**:** "Second" of current time of RTC. Content: K0~K59.
3. Usually the time of RTC in **S** is read by TRD instruction first then compared by TCMP instruction. If operand **S** exceeds the available range, operation error occurs and M1067 = ON, M1068 = ON. D1067 stores the error code 0E1A (HEX).

Program Example:

1. When X0 = ON, the instruction executes and the RTC current time in D20~D22 is compared with the set value 12:20:45. Comparison result is indicated by M10~M12. When X0 goes from ON→OFF, the instruction is disabled however the ON/OFF status of M10~M12 remains.
2. Connect M10 ~ M12 in series or in parallel to obtain the results of ≧, ≦, and ≠.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
161	TZCP	P	S1 S2 S D	Time zone compare				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1											*	*	*			TZCP, TZCPP: 9 steps
S_2											*	*	*			
S											*	*	*			
D		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

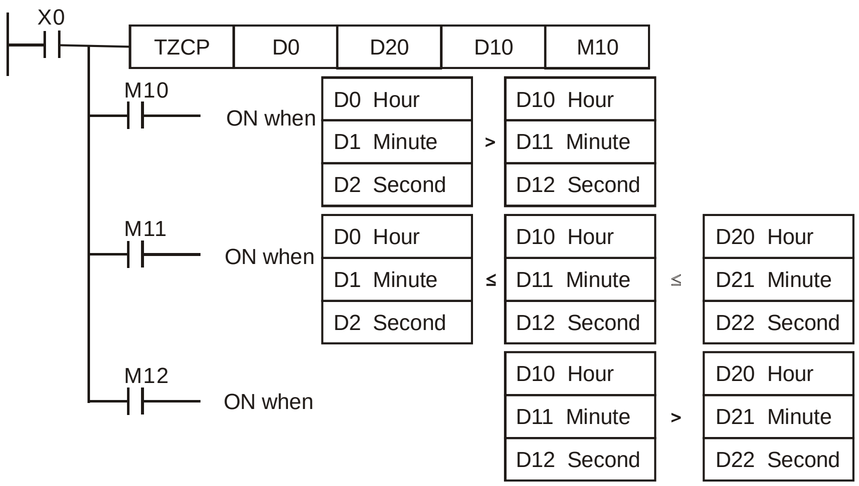
S_1: Lower bound of the time for comparison (occupies 3 consecutive devices) **S_2**: Upper bound of the time for comparison (occupies 3 consecutive devices) **S**: Current time of RTC (occupies 3 consecutive devices) **D**: Comparison result (occupies 3 consecutive devices)

Explanations:

1. TZCP instruction compares current RTC time in **S** with the range set in **S_1**~ **S_2** and the comparison result is stored in **D**.
2. **S_1**, **S_1** +1, **S_1** +2: The "hour", "minute" and "second" of the lower bound value for comparison.
3. **S_2**, **S_2** +1, **S_2** +2: The "hour", "minute" and "second" of the upper bound value for comparison.
4. **S**, **S** +1, **S** +2: The "hour", "minute" and "second" of the current time of RTC.
5. Usually the time of RTC in **S** is read by TRD instruction first then compared by TZMP instruction. If operand **S**, **S_1**, **S_2** exceed the available range, operation error occurs and M1067 = ON, M1068 = ON. D1067 stores the error code 0E1A (HEX).
6. If **S** < **S_1** and **S** < **S_2**, **D** is ON. When **S** > **S_1** and **S** > **S_2**, **D**+2 is ON. For other conditions, **D** + 1 will be ON. (Lower bound **S_1** should be less than upper bound **S_2**.)

Program Example:

When X0 = ON, TZCP instruction executes and M10~M12 will be ON to indicate the comparison results. When X0 = OFF, the instruction is disabled but the ON/OFF status of M10~M12 remains.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
162	TADD	P	(S1) (S2) (D)	Time addition				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1											*	*	*			TADD, TADDP: 7 steps
S_2											*	*	*			
D											*	*	*			

(Bit Devices: X, Y, M, S; Word devices: K, H, KnX, KnY, KnM, KnS, T, C, D, E, F)

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

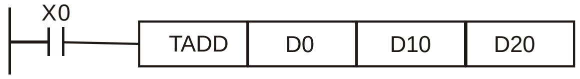
S_1: Time augend (occupies 3 consecutive devices) **S_2**: Time addend (occupies 3 consecutive devices) **D**: Addition result (occupies 3 consecutive devices)

Explanations:

1. TADD instruction adds the time value (Hour, Minute Second) **S_1** with the time value (Hour, Minute Second) **S_2** and stores the result in **D**.
2. If operand **S_1**, **S_2** exceed the available range, operation error occurs and M1067 = ON, M1068 = ON. D1067 stores the error code 0E1A (HEX).
3. If the addition result is larger than 24 hours, the carry flag M1022 will be ON and the value in **D** will be the result of "sum minuses 24 hours".
4. If the sum equals 0 (00:00:00), Zero flag M1020 will be ON.

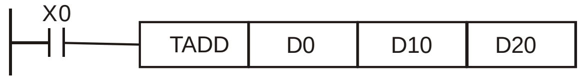
Program Example:

When X0 = ON, TADD instruction executes and the time value in D0~D2 is added with the time value in D10~D12. The addition result is stored in D20~D22.



D0 08(Hour)		D10 06(Hour)		D20 14(Hour)
D1 10(Min)	+	D11 40(Min)	→	D21 50(Min)
D2 20(Sec)		D12 06(Sec)		D22 26(Sec)
08:10:20		06:40:06		14:50:26

If the addition result is greater than 24 hours, the Carry flag M1022 = ON.



D0 18(Hour)		D10 11(Hour)		D20 06(Hour)
D1 40(Min)	+	D11 30(Min)	→	D21 10(Min)
D2 30(Sec)		D12 08(Sec)		D22 38(Sec)
18:40:30		11:30:08		06:10:38

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
163	TSUB	P	S1 S2 D	Time subtraction				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TSUB, TSUBP: 7 steps
S_1											*	*	*			
S_2											*	*	*			
D											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

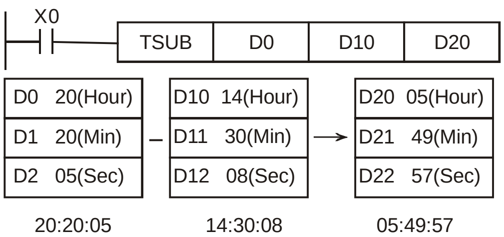
S_1: Time minuend (occupies 3 consecutive devices) **S_2**: Time subtrahend (occupies 3 consecutive devices) **D**: Subtraction result (occupies 3 consecutive devices)

Explanations:

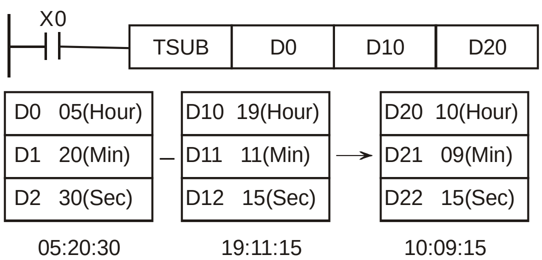
1. TSUB instruction subtracts the time value (Hour, Minute Second) **S_1** with the time value (Hour, Minute Second) **S_2** and stores the result in **D**.
2. If operand **S_1**, **S_2** exceed the available range, operation error occurs and M1067 = ON, M1068 = ON. D1067 stores the error code 0E1A (HEX).
3. If the subtraction result is a negative value (less than 0), Borrow flag M1020 = ON and the value in **D** will be the result of "the negative value pluses 24 hours".
4. If the subtraction result (remainder) equals 0 (00:00:00), Zero flag M1020 will be ON.
5. Besides using TRD instruction, MOV instruction can also be used to move the RTC value to D1315 (Hour), D1314 (Minute), D1313 (Second) for reading the current time of RTC.

Program Example:

When X0 = ON, TSUB instruction executes and the time value in D0~D2 is subtracted by the time value in D10~D12. The subtraction result is stored in D20~D22.



If the subtraction result is a negative value (less than 0), Borrow flag M1021 = ON.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
166	TRD	P	D	Time read				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TRD, TRDP: 3 steps
D											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operand:

D: Current time of RTC (occupies 7 consecutive devices)

Explanations:

1. TRD instruction reads the 7 real-time data of RTC (year (A.D.), day(Mon.Sun.), month, day, hour, minute, second from D1319~D1313 and stores the read data in registers specified by **D.**
2. Only when power is on can RTCs of SS2 series perform the fuction of timing. The RTC data registers D1319~D1313 are latched. When power is resumed, the RTC will resume the stored time value before power down. Therefore, we suggest users modify the RTC value every time when power is ON.
3. RTCs of SA2/SE V1.0 及 ES2/EX2/SX2 V2.0 series can still operate for one or two weeks after the power is off (they vary with the ambient temperature). Therefore, if the machine has not operated since one or two weeks ago, please reset RTC.
4. D1319 only stores the 2-digit year in A.D. If 4-digit year data is required, please refer to **Points to note** below.
5. For relative flags and registers please refer to **Points to note**.

Program Example:

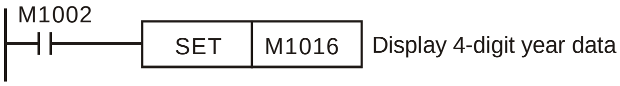
When X0 = ON, TRD instruction reads the current time of RTC to the specified register D0~D6.

The content of D1318: 1 = Monday; 2 = Tuesday … 7 = Sunday.

Special D	Item	Content		Normal D	Item
D1319	Year (A.D.)	00~99	→	D0	Year (A.D.)
D1318	Day (Mon.~Sun.)	1~7	→	D1	Day (Mon.~Sun.)
D1317	Month	1~12	→	D2	Month
D1316	Day	1~31	→	D3	Day
D1315	Hour	0~23	→	D4	Hour
D1314	Minute	0~59	→	D5	Minute
D1313	Second	0~59	→	D6	Second

Points to note:

1. There are two methods to correct built-in RTC:
   - Correcting by API167 TWR instruction
     Please refer to explanation of instruction TWR (API 167)
   - Setting by peripheral device
     Using WPLSoft / ISPSoft (Ladder editor)
2. Display 4-digit year data:
   - D1319 only stores the 2-digit year in A.D. If 4-digit year data is required, please insert the following instruction at the start of program.



   - The original 2-digit year will be switched to a 4-digit year, i.e. the 2-digit year will pluses 2,000. If users need to write in new time in 4-digit year display mode, only a 2-digit year data is applicable (0 ~ 99, indicating year 2000 ~ 2099). For example, 00 = year 2000, 50 = year 2050 and 99 = year 2099. However, 2000 ~ 2099 can be written in ES2/EX2 V3.0, SS2 V3.2, SA2 V2.6, SX2 V2.4, and SE V1.6 (and above).
   - Flags and special registers for RTC

Device	Content	Function
M1016	Year display mode of RTC	OFF: D1319 stores 2-digit year data in A.D. ON: D1319 stores 2-digit year data in A.D + 2000
M1017	±30 seconds correction on RTC	Correction takes place when M1017 goes from OFF to ON (Second data in 0 ~ 29: reset to 0. Second data in 30 ~ 59: minute data pluses 1, second data resets)

Device	Content	Range
D1313	Second	0-59
D1314	Minute	0-59
D1315	Hour	0-23
D1316	Day	1-31
D1317	Month	1-12
D1318	Day (Mon. ~ Sun.)	1-7
D1319	Year	0-99 (two digit year data)

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
167		TWR	P	(S)	Time write				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											TWR, TWRP: 5 steps
S											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operand:

S: Set value for RTC (occupies 7 consecutive devices)

Explanations:

1. TWR instruction updates the RTC with the value set in **S**.
2. If the time data in **S** exceeds the valid calendar range, it will result in an “operation error”. PLC will writes in the smallest valid value automatically, M1067 = ON, M1068 = ON, and error code 0E1A (HEX) is recorded in D1067
3. For explanations of associated flags and the characteristics of RTCS, please refer to **Points to note** of TRD instruction.

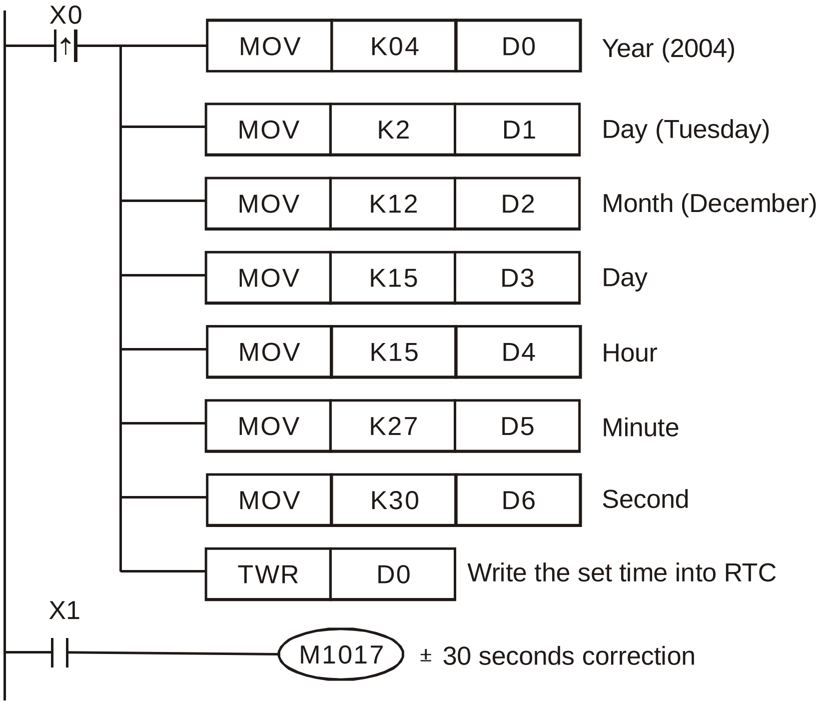
Program Example 1:

When X0 = ON, write the new time into RTC.

	Normal D	Item	Range		Special D	Item	
Set value	D20	Year (A.D.)	00~99	→	D1319	Year (A.D.)	RTC
	D21	Day (Mon.~Sun.)	1~7	→	D1318	Day (Mon.~Sun.)	
	D22	Month	1~12	→	D1317	Month	
	D23	Day	1~31	→	D1316	Day	
	D24	Hour	0~23	→	D1315	Hour	
	D25	Minute	0~59	→	D1314	Minute	
	D26	Second	0~59	→	D1313	Second	

Program Example 2:

1. Set the current time in RTC as 2004/12/15, Tuesday, 15:27:30.
2. The content of D0~D6 is the set value for adjusting RTC.
3. When X0 = ON, update the time of RTC with the set value.
4. When X1 = ON, perform ±30 seconds correction. Correction takes place when M1017 goes from OFF to ON (Second data in 0 ~ 29: reset to 0. Second data in 30 ~ 59: minute data pluses 1, second data resets).


X0
MOV K04 D0
Year (2004)
MOV K2 D1
Day (Tuesday)
MOV K12 D2
Month (December)
MOV K15 D3
Day
MOV K15 D4
Hour
MOV K27 D5
Minute
MOV K30 D6
Second
TWR D0
Write the set time into RTC
X1
M1017
± 30 seconds correction

API	Mnemonic		Operands	Function	Controllers			
169	D	HOUR	(S) (D1) (D2)	Hour meter	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	HOUR: 7 steps
D_1													*			DHOUR: 13 steps
D_2		*	*	*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Set-point value for driving the output device (Unit: hour) **D_1**: Current time being measured

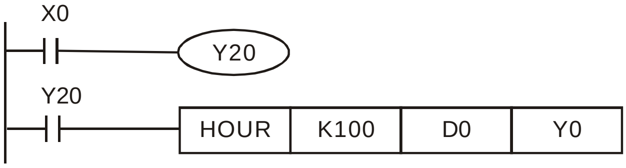
D_2: Output device

Explanations:

1. HOUR instruction drives the output device **D_2** when the measured current time **D_1** reaches the set-point value in **S**.
2. Range of **S**: K1~K32,767; unit: hour. Range of **D_1** in 16-bit instruction: K0~K32,767. Range of **D_1** +1 (current time less than an hour): K0 ~K3,599; unit: second.
3. When the ON-time of the drive contact reaches the set-point value, output device will be ON. The instruction can be applied for controlling the working hours of machine or conducting preventive maintenance.
4. After output device is ON, the current time will still be measured in **D_1**.In 16-bit instruction, when the current time measured reaches the maximum 32,767 hours / 3,599 seconds, the timing will stop. To restart the timing, **D_1** and **D_1** + 1 have to be reset.
5. In 32-bit instruction, when the current time measured reaches the maximum 2,147,483,647 hours / 3,599 seconds, the timing will stop. To restart the timing, **D_1** ~ **D_1** + 2 have to be reset.
6. If operand **S** uses device F, only 16-bit instruction is available. HOUR instruction can be used for four times in the program.

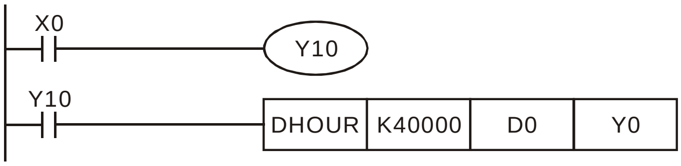
Program Example 1:

In 16-bit instruction, when X0 = ON, Y20 will be ON and the timing will start. When the timing reaches 100 hours, Y0 will be ON and D0 will record the current time measured (in hour). D1 will record the current time less than an hour (0 ~ 3,599; unit: second)..



Program Example 2:

In 32-bit instruction, when X0 = ON, Y10 will be ON and the timing will start. When the timing reaches 40,000 hours, Y0 will be ON. D1 and D0 will record the current time measured (in hour) and D2 will record the current time less than an hour (0 ~ 3,599; unit: second).



3.6.16 Gray Code

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
170	GRY	DGRY	✓	BIN → Gray Code	✓	✓	✓	✓	5	9
171	GBIN	DGBIN	✓	Gray Code → BIN	✓	✓	✓	✓	5	9

API	Mnemonic			Operands	Function	Controllers			
170	D	GRY	P	(S) (D)	BIN → Gray Code	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	GRY, GRYP: 5 steps
S					*	*	*	*	*	*	*	*	*	*	*	DGRY, DGRYP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

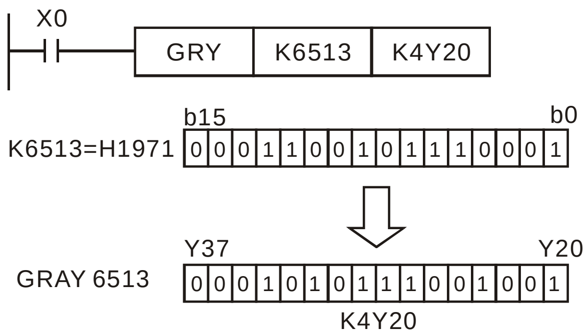
S: Source device **D**: Operation result (Gray code)

Explanations:

1. GRY instruction converts the BIN value in **S** to Gray Code and stores the converted result in specified register **D**.
2. Available range of **S**:
   16-bit instruction: 0~32,767
   32-bit instruction: 0~2,147,483,647
3. If operand **S** exceeds the available range, operation error occurs and M1067 = ON, M1068 = ON. D1067 stores the error code 0E1A (HEX)
4. If operands **S** and **D** use device F, only 16-bit instruction is applicable.

Program Example:

When X0 = ON, GRY instruction executes and converts K6513 to Gray Code. The operation result is stored in K4Y20, i.e. Y20 ~ Y37.



API	Mnemonic			Operands	Function	Controllers			
171	D	GBIN	P	(S) (D)	Gray Code → BIN	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	GBIN, GBINP: 5 steps DGBIN, DGBINP: 9 steps
S					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Source device **D**: Operation result (BIN value)

Explanations:

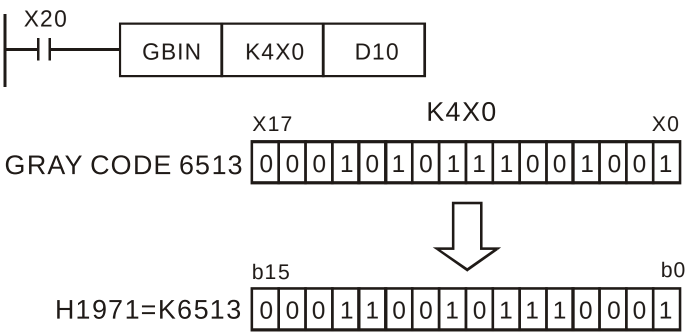
1. GBIN instruction converts the Gray Code in **S** to BIN value and stores the converted result in specified register **D**.
2. This instruction can be used to read the value from an absolute position type encoder (generally a Gray Code encoder) which is connected to the PLC inputs. The Gray code is converted to BIN value and stored in the specified register.
3. Available range of **S**:

   16-bit instruction ：0~32,767

   32-bit instruction ：0~2,147,483,647
4. If operand **S** exceeds the available range, operation error occurs and the instruction is disabled.
5. If operands **S** and **D** use device F, only 16-bit instruction is applicable.

Program Example:

When X20 = ON, the Gray Code value in the absolute position type encoder connected to X0~X17 inputs is converted to BIN value and stored in D10.



3.6.17 Matrix Operation

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
180	MAND	-	✓	Matrix AND	✓	✓	✓	✓	9	-
181	MOR	-	✓	Matrix OR	✓	✓	✓	✓	9	-
182	MXOR	-	✓	Matrix XOR	✓	✓	✓	✓	9	-
183	MXNR	-	✓	Matrix XNR	✓	✓	✓	✓	9	-
184	MINV	-	✓	Matrix inverse	✓	✓	✓	✓	7	-
185	MCMP	-	✓	Matrix compare	✓	✓	✓	✓	9	-
186	MBRD	-	✓	Matrix bit read	✓	✓	✓	✓	7	-
187	MBWR	-	✓	Matrix bit write	✓	✓	✓	✓	7	-
188	MBS	-	✓	Matrix bit shift	✓	✓	✓	✓	7	-
189	MBR	-	✓	Matrix bit rotate	✓	✓	✓	✓	7	-
190	MBC	-	✓	Matrix bit status count	✓	✓	✓	✓	7	-

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
180		MAND	P	S1 S2 D n	Matrix AND				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MAND, MANDP: 9 steps
S_1							*	*	*	*	*	*	*			
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Matrix source device 1 **S_2**: Matrix source device 2 **D**: Operation result

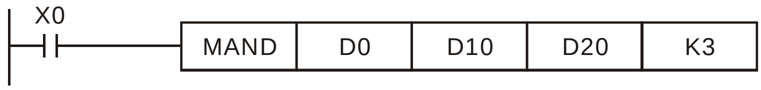
n: Matrix length (**n** = K1~K256)

Explanations:

1. MAND instruction performs matrix AND operation between matrix source device 1 and 2 with matrix length **n** and stores the operation result in **D**.
2. Rule of AND operation: the result is 1 only when both two bits are 1; otherwise the result is 0.
3. If operands **S_1**, **S_2**, **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable.

Program Example:

When X0 = ON, MAND performs matrix AND operation between 16-bit registers D0~D2 and 16-bit registers D10~D12. The operation result is then stored in 16-bit registers D20~D22.



Before Execution

		b15															b0
S1	D0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1
	D1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1
	D2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1

MAND

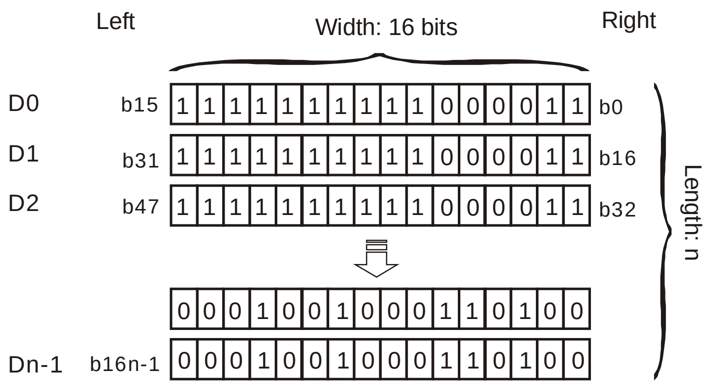
S2	D10	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0
	D11	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0
	D12	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0

After Execution

D	D20	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0
	D21	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0
	D22	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0

Points to note:

1. A matrix consists of more than 1 consecutive 16-bit registers. The number of registers is indicated as the matrix length (n). A matrix contains 16 × n bits (points) and the matrix instructions conduct bit operation, i.e. operation is performed bit by bit.
2. Matrix instructions designate a single bit of the 16 × n bits (b_0 ~ b_{16n-1}) for operation. The bits in matrix are not operated as value operation.
3. The matrix instructions process the moving, copying, comparing and searching of one-to-many or many-to-many matrix operation, which are a very handy and important application instructions.
4. The matrix operation requires a 16-bit register for designating a bit among the 16n bits in the matrix. The register is the Pointer (Pr) of the matrix, designated by the user in the instruction. The valid range of Pr is 0 ~ 16n -1, corresponding to b0 ~ b16n-1 in the matrix.
5. The bit number decreases from left to right (see the figure below). With the bit number, matrix operation such as bit shift left, bit shift right, bit rotation can be performed and identified.



6. The matrix width (C) is fixed as 16 bits.
7. Pr: matrix pointer. E.g. if Pr is 15, the designated bit is b15.
8. Matrix length (R) is n: n = 1 ~ 256.

Example: This matrix is composed of D0, n = 3; D0 = HAAAA, D1 = H5555, D2 = HAAFF

	C_{15}	C_{14}	C_{13}	C_{12}	C_{11}	C_{10}	C_9	C_8	C_7	C_6	C_5	C_4	C_3	C_2	C_1	C_0	
R_0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	D0
R_1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	D1
R_2	1	0	1	0	1	0	1	0	1	1	1	1	1	1	1	1	D2

Example: This matrix is composed of K2X20, n = 3; K2X20 = H37, K2X30 = H68, K2X40 = H45

	C_{15}	C_{14}	C_{13}	C_{12}	C_{11}	C_{10}	C_9	C_8	C_7	C_6	C_5	C_4	C_3	C_2	C_1	C_0	
R_0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	X_{20}~X_{27}
R_1	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	X_{30}~X_{37}
R_2	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	X_{40}~X_{47}

Fill "0" into the blank in R0(C_{15}-C_8), R1(C_{15}-C_8), and R2(C_{15}-C_8).

API	Mnemonic		Operands	Function	Controllers			
181	MOR	P	S1 S2 D n	Matrix OR	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F
S_1							*	*	*	*	*	*	*		
S_2							*	*	*	*	*	*	*		
D								*	*	*	*	*	*		
n					*	*							*		

Program Steps: MOR, MORP: 9 steps

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Matrix source device 1 **S_2**: Matrix source device 2. **D**: Operation result

n: Matrix length (**n** = K1~K256)

Explanations:

1. MOR instruction performs matrix OR operation between matrix source device 1 and 2 with matrix length **n** and stores the operation result in **D**.
2. Rule of matrix OR operation: the result is 1 if either of the two bits is 1. The result is 0 only when both two bits are 0.
3. If operands **S_1**, **S_2**, **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable.

Program Example:

When X0 = ON, MOR performs matrix OR operation between 16-bit registers D0~D2 and 16-bit registers D10~D12. The operation result is then stored in 16-bit registers D20~D22.

X0	MOR	D0	D10	D20	K3

Before Execution

		b15															b0
S1	D0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	D1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	D2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

MOR

S2	D10	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1
	D11	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1
	D12	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1

After Execution

D	D20	0	1	0	1	1	1	1	1	1	1	1	1	0	1	0	1
	D21	0	1	0	1	1	1	1	1	1	1	1	1	0	1	0	1
	D22	0	1	0	1	1	1	1	1	1	1	1	1	0	1	0	1

API	Mnemonic		Operands	Function	Controllers			
182	MXOR	P	S₁ S₂ D n	Matrix XOR	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1							*	*	*	*	*	*	*			MXOR, MXORP: 9 steps
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Matrix source device 1 **S_2**: Matrix source device 2 **D**: Operation result

n: Matrix length (**n** = K1~K256)

Explanations:

1. MXOR instruction performs matrix XOR operation between matrix source device 1 and 2 with matrix length **n** and stores the operation result in **D**
2. Rule of matrix XOR operation: the result is 1 if the two bits are different. The result is 0 if the two bits are the same
3. If operands **S_1**, **S_2**, **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable..

Program Example:

When X0 = ON, MXOR performs matrix XOR operation between 16-bit registers D0~D2 and 16-bit registers D10~D12. The operation result is then stored in 16-bit registers D20~D22

X0

MXOR	D0	D10	D20	K3

Before Execution

		b15															b0
S_1	D0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	D1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	D2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

MXOR

S_2	D10	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1
	D11	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1
	D12	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1

After Execution

D	D20	0	1	0	1	1	0	1	0	1	1	1	0	0	0	0	0
	D21	0	1	0	1	1	0	1	0	1	1	1	0	0	0	0	0
	D22	0	1	0	1	1	0	1	0	1	1	1	0	0	0	0	0

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
183	MXNR	P	S1 S2 D n	Matrix XNR				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1							*	*	*	*	*	*	*			MXNR, MXNRP: 9 steps
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Matrix source device 1 **S_2**: Matrix source device 2 **D**: Operation result

n: Matrix length (K1~K256)

Explanations:

1. MXNR instruction performs matrix XNR operation between matrix source device 1 and 2 with matrix length **n** and stores the operation result in **D**.
2. Rule of matrix XNR operation: The result is 1 if the two bits are the same. The result is 0 if the two bits are different.
3. If operands **S_1**, **S_2**, **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable.

Program Example:

When X0 = ON, MXNR performs matrix XNR operation between 16-bit registers D0~D2 and 16-bit registers D10~D12. The operation result is then stored in 16-bit registers D20~D22.

X0	MXNR	D0	D10	D20	K3

Before Execution

		b15															b0
S1	D0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	D1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	D2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

MXNR

S2	D10	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1
	D11	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1
	D12	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1

After Execution

D	D20	1	0	1	0	0	1	0	1	0	0	0	0	1	1	1	1
	D21	1	0	1	0	0	1	0	1	0	0	0	0	1	1	1	1
	D22	1	0	1	0	0	1	0	1	0	0	0	0	1	1	1	1

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
184	MINV	P	(S) (D) (n)	Matrix inverse				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											MINV, MINVP: 7 steps
S							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Matrix source device **D**: Operation result **n**: Matrix length (K1~K256)

Explanations:

1. MINV instruction performs inverse operation on matrix source device **S** with matrix length **n** and stores the result in **D**.
2. If operands **S**, **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable.

Program Example:

When X0 = ON, MINV performs inverse operation on 16-bit registers D0~D2. The operation result is then stored in 16-bit registers D20~D22

X0	MINV	D0	D20	K3

Before Execution (S)

	b15															b0
D0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
D1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
D2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

MINV

⇩

After Execution (D)

D20	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
D21	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
D22	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
185		MCMP	P	S1 S2 n D	Matrix compare				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
S_1							*	*	*	*	*	*	*			MCMP, MCMPP: 9 steps
S_2							*	*	*	*	*	*	*			
n					*	*							*			
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S_1: Matrix source device 1 **S_2**: Matrix source device 2 **n**: Matrix length (K1~K256)

D: Pointer Pr; comparison result (bit number)

Explanations:

1. MCMP instruction compares each bit between matrix **S_1** and matrix **S_2** and stores the bit number of the comparison result in **D**. The comparison starts from the next bit of the pointer.
2. The matrix comparison flag (M1088) decides to compare between equivalent values (M1088 = ON) or different values (M1088 = OFF). When the comparison is completed, it will stop immediately and M1091= ON to indicate that matched result is found. When the comparison progresses to the last bit, M1089 = ON to indicate that the comparison has come to the end of the matrix and the number of the last bit will be stored in **D**. In next scan cycle, comparison starts again from the first bit (bit 0), at the same time M1090 = ON to indicate the start of the comparison. When **D** (Pr) exceeds the valid range, M1092 = ON to indicate pointer error, and the instruction will be disabled.
3. The matrix operation requires a 16-bit register for designating a bit among the 16n bits in the matrix. The register is the Pointer (Pr) of the matrix, designated by the user in the instruction. The valid range of Pr is 0 ~ 16n -1, corresponding to b0 ~ b16n-1 in the matrix. The value of pointer should not be modified during the execution of matrix instructions so as to prevent execution errors.
4. When M1089 and M1091 take place at the same time, both flags will ON..
5. If operands **S_1**, **S_2**, or **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable.

Program Example:

When X0 goes from OFF to ON with M1090 = OFF (comparison starts from Pr), the search will start from the bit marked with “*” (current Pr value +1) for the bits with different status (M1088 = OFF).

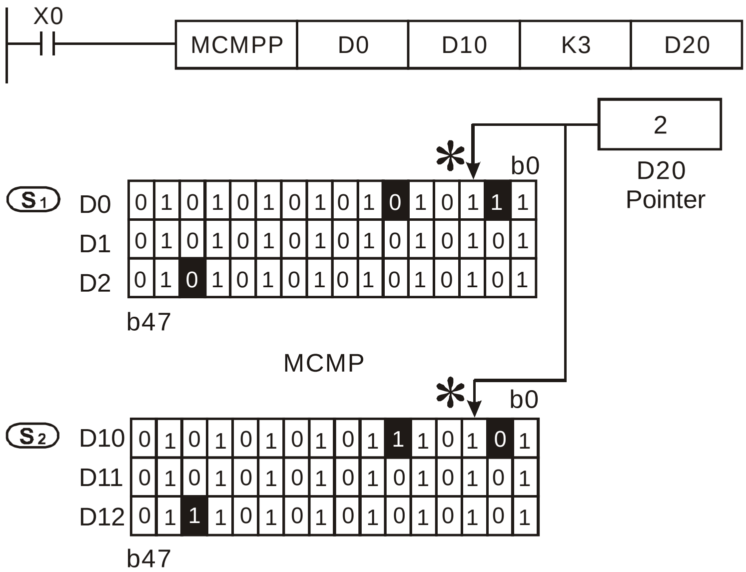
Assume pointer D20 = 2, the following four results (❶, ❷, ❸, ❹) can be obtained when X0 goes from OFF→ON for four times.

❶ D20 = 5, M1091 = ON (matched result found), M1089 = OFF

❷ D20 = 45, M1091 = ON, M1089 = OFF.

❸ D20 = 47, M1091 = OFF, M1089 = ON (comparison proceeds to he last bit)

❹ D20 = 1, M1091 = ON, = OFF.



Points to note:

Associated flags and registers:

M1088:	Matrix comparison. Comparing between equivalent values (M1088 = ON) or different values (M1088 = OFF)
D1089:	Indicating the end of Matrix. When the comparison reaches the last bit, M1089 = ON
D1090:	Indicating start of Matrix comparison. When the comparison starts from the first bit, M1090 = ON
D1091:	Indicating matrix searching results. When the comparison has matched results, comparison will stop immediately and M1091 = ON
D1092:	Indicating pointer error. When the pointer Pr exceeds the comparison range, M1092 = ON.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
186	MBRD	P	S n D	Matrix bit read				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S							*	*	*	*	*	*	*			MBRD, MBRDP: 7 steps
n					*	*							*			
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Matrix source device **n**: Matrix length (K1~K256). **D**: Pointer Pr (bit number)

Explanations:

1. MBRD instruction reads the bit status of the matrix. When MBRD executes, the status of M1094 (Matrix pointer clear flag) will be checked first. If M1094 = ON, Pr value in **D** will be cleared and the instruction reads from the first bit. The bit status is read out and mapped to M1095 (Carry flag for matrix operation). After a bit is read, MBRD checks the status of M1093 (Matrix pointer increasing flag). If M1093 = ON, MBRD instruction will proceed to read the next bit, i.e. Pr value plus 1. When MBRD proceeds to the last bit, M1089 = ON, indicating the end of the Matrix, and **D** records the last bit number. After this, MBRD instruction stops.
2. The Pointer (Pr) of the matrix is designated by the user in the instruction. The valid range of Pr is 0 ~ 16n -1, corresponding to b0 ~ b16n-1 in the matrix. If the Pr value exceeds the valid range, M1092 = ON and the instruction will be disabled.
3. If operands **S** or **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable.

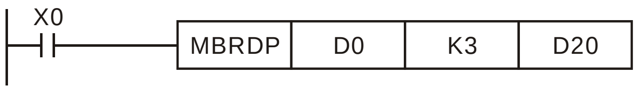
Program Example:

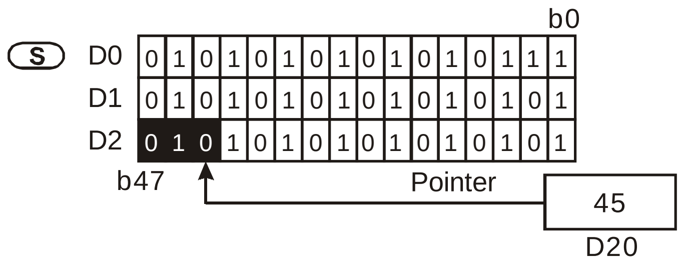
1. When X0 goes from OFF→ON with M1094 = ON (Clear Pr value) and M1093 = ON (Increase Pr value), the reading will start from the first bit and Pr value increases 1 after a bit is read.
2. Assume present value of pointer D20 = 45, the following 3 results (❶, ❷, ❸) can be obtained when X0 is executed from OFF→ON for 3 times.

   ❶ D20 = 45, M1095 = OFF, M1089 = OFF

   ❷ D20 = 46, M1095 = ON (bit status is ON), M1089 = OFF.

   ❸ D20 = 47, M1095 = OFF, M1089 = ON. (reading proceeds to the last bit)





Points to note:

Associated flags and registers:

M1089:	Indicating the end of Matrix. When the comparison reaches the last bit, M1089 = ON
M1092:	Indicating pointer error. When the pointer Pr exceeds the comparison range, M1092 = ON.
M1093:	Matrix pointer increasing flag. Adding 1 to the current value of the Pr
M1094:	Matrix pointer clear flag. Clear the current value of the Pr to 0
M1095:	Carry flag for matrix rotation/shift/output

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
187	MBWR	P	S n D	Matrix bit write				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MBWR, MBWRP: 7 steps
S							*	*	*	*	*	*	*			
n					*	*							*			
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

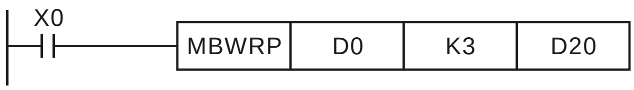
S: Matrix source device **n**: Matrix length (K1~K256) **D**: Pointer Pr (bit number).

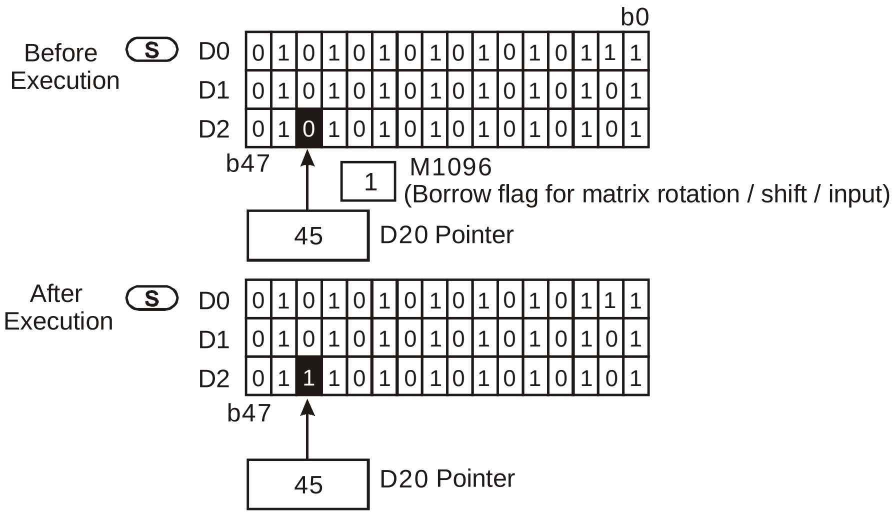
Explanations:

1. MBWR instruction writes the bit status of the matrix. When MBWR executes, the status of M1094 (Matrix pointer clear flag) will be checked first. If M1094 = ON, Pr value in **D** will be cleared and the instruction writes from the first bit. The bit status of M1096 (Borrow flag for matrix operation) is written into the first bit of the matrix. After a bit is written, MBWR checks the status of M1093 (Matrix pointer increasing flag). If M1093 = ON, MBWR instruction will proceed to write the next bit, i.e. Pr value plus 1. When MBWR proceeds to the last bit, M1089 = ON, indicating the end of the Matrix, and **D** records the last bit number. After this, MBWR instruction stops.
2. The Pointer (Pr) of the matrix is designated by the user in the instruction. The valid range of Pr is 0 ~ 16n -1, corresponding to b0 ~ b16n-1 in the matrix. If the Pr value exceeds the valid range, M1092 = ON and the instruction will be disabled.
3. If operands **S** or **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable.

Program Example:

1. When X0 goes from OFF→ON with M1094 = OFF (Starts from Pr value) and M1093 = ON (Increase Pr value), the writing will start from the bit number in Pr and Pr value increases 1 after a bit is written.
2. Assume present value of pointer D20 = 45 and M1096 = ON (1) , the following result can be obtained when X0 is executed once from OFF→ON.





Points to note:

Associated flags and registers:

M1089: Indicating the end of Matrix. When the comparison reaches the last bit, M1089 = ON

M1092: Indicating pointer error. When the pointer Pr exceeds the comparison range, M1092 = ON.

M1093: Matrix pointer increasing flag. Adding 1 to the current value of the Pr

M1094: Matrix pointer clear flag. Clear the current value of the Pr to 0

M1096: Borrow flag for matrix rotation/shift/input

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
188		MBS	P	S D n	Matrix bit shift				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S							*	*	*	*	*	*	*			MBS, MBSP: 7 steps
D								*	*	*	*	*	*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Matrix source device **D**: Operation result **n**: Matrix length (K1~K256)

Explanations:

1. MBS instruction shifts the bits in the matrix to the left or the right. M1097 = OFF, bits shift to the left, M1097 = ON, bits shift to the right. The empty bit (left shift: b0; right shift: b16**n**-1) after every bit is shifted once will be filled with the value of M1096 (Borrow flag for matrix operation). The bit which is shifted out of the matrix (left shift: b16**n**-1; right shift: b0) will be sent to M1095 (Carry flag for matrix operation) and operation result is stored in **D**.
2. The pulse execution instruction (MBSP) is generally adopted.
3. If operands **S** or **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable
4. Associated flags:

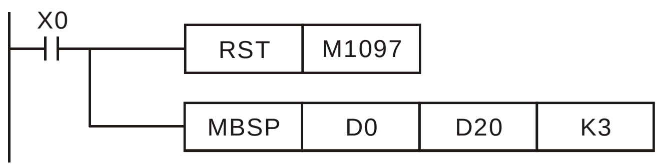
   M1095: Carry flag for matrix rotation/shift/output

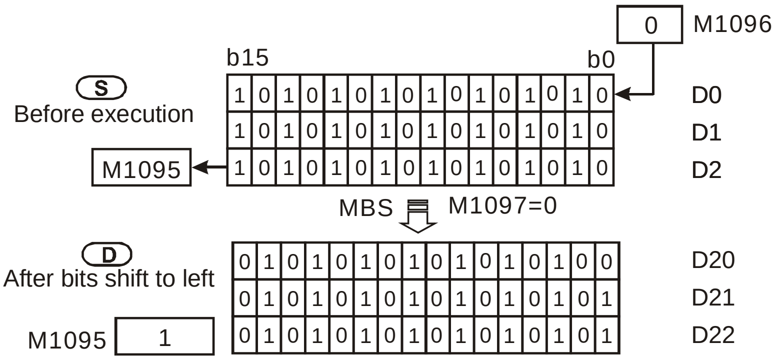
   M1096: Borrow flag for matrix rotation/shift/input

   M1097: Direction flag for matrix rotation/shift

Program Example 1:

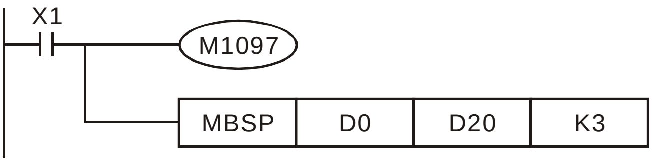
When X0 = ON, M1097 = OFF, indicating a left matrix shift is performed. Assume matrix borrow flag M1096 = OFF (0) and the 16-bit registers D0 ~ D2 will perform a left matrix shift and the result will be stored in the matrix of the 16-bit registers D20 ~ D22, meanwhile the matrix carry flag M1095 will be ON (1). .

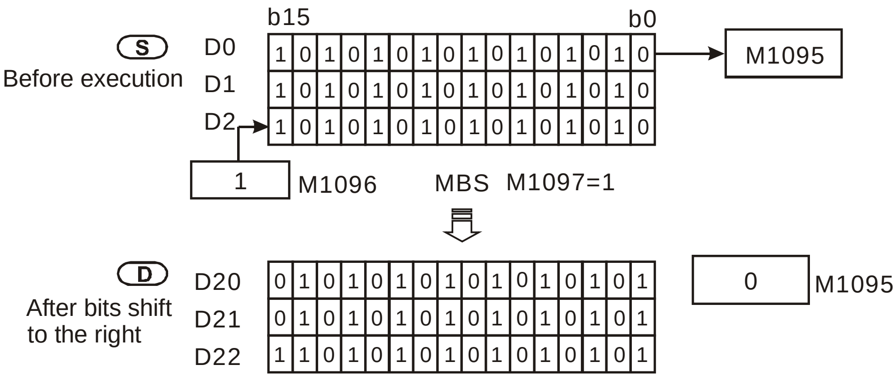




Program Example 2:

When X1 = ON, M1097 = ON, indicating a right matrix shift is performed. Assume matrix borrow flag M1096 = ON (1) and the 16-bit registers D0 ~ D2 will perform a right matrix shift and the result will be stored in the matrix of the 16-bit registers D20 ~ D22, meanwhile the matrix carry flag M1095 will be OFF (0).





API	Mnemonic		Operands	Function	Controllers			
189	MBR	P	(S) (D) (n)	Matrix bit rotate	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
S							*	*	*	*	*	*	*			MBR, MBRP: 7 steps
D								*	*	*	*	*	*			
n					*	*							*			

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Matrix source device **D**: Operation result **n**: Matrix length (K1~K256)

Explanations:

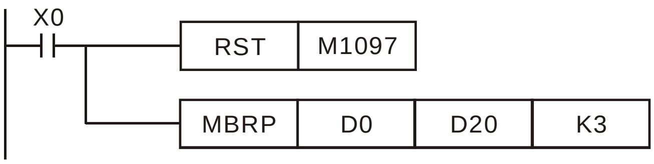
1. MBR instruction rotates the bits in the matrix to the left or the right. M1097 = OFF, bits rotate to the left, M1097 = ON, bits rotate to the right. The empty bit (left rotate: b0; right rotate: b16**n**-1) after rotation performed once will be filled with the bit which is rotated out of the matrix (left rotate: b16**n**-1; right rotate: b0) and the operation result is stored in **D**. In addition, the bit which is rotated out of the matrix will also be moved to M1095 (Carry flag for matrix operation).
2. The pulse execution instruction MBRP is generally adopted.
3. If operands **S** or **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable.
4. Associated flags:

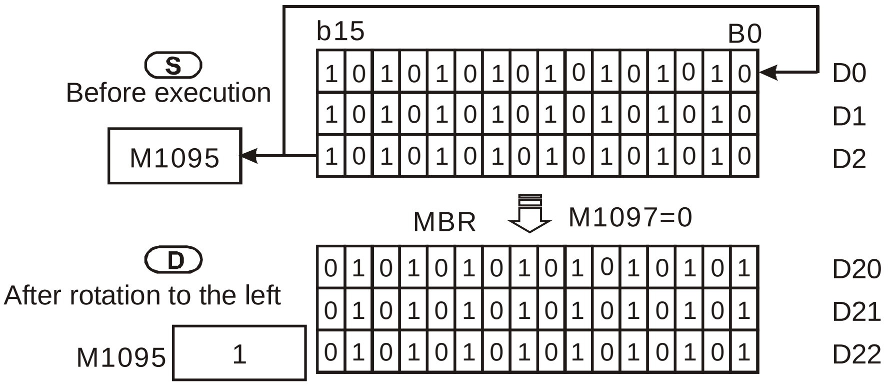
   M1095: Carry flag for matrix rotation/shift/output.

   M1097: Direction flag for matrix rotation/shift

Program Example 1:

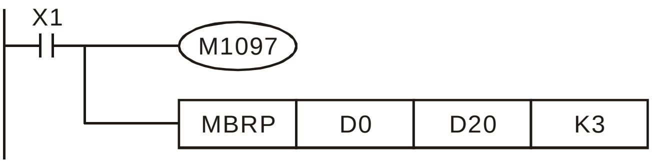
When X0 = ON, M1097 = OFF, indicating a left matrix rotation is performed. The 16-bit registers D0 ~ D2 will perform a left matrix rotation and the result will be stored in the matrix of the 16-bit registers D20 ~ D22. The matrix carry flag M1095 will be ON (1)

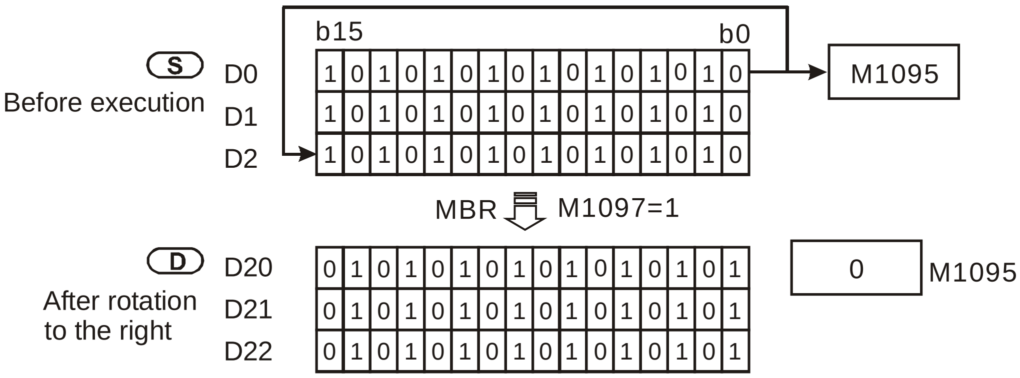




Program Example 2:

When X1 = ON, M1097 = ON, indicating a right matrix rotation is performed. The 16-bit registers D0 ~ D2 will perform a right matrix rotation and the result will be stored in the matrix of the 16-bit registers D20 ~ D22. The matrix carry flag M1095 will be OFF (0).





API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2/ EC5	SS2	SA2 SE	SX2
190		MBC	P	S n D	Matrix bit status count				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											MBC, MBCP: 7 steps
S							*	*	*	*	*	*	*			
n					*	*							*			
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2	ES2/EX2/ EC5	SS2	SA2 SE	SX2

Operands:

S: Matrix source device **n**: Matrix length (K1~K256) **D**: Operation result

Explanations:

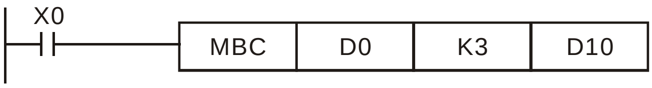
1. MBC instruction counts the number of bit 1 or bit 0 in the matrix with matrix length **n** and stores the counted number in **D**.
2. If operands **S** or **D** use KnX, KnY, KnM, KnS format, only n = 4 is applicable.
3. When M1098 = ON, MBC instruction counts the number of bit 1. M1098 = OFF, MBC counts the number of bit 0. If bits counting result is 0, M1099 = ON
4. Associated flags:

   M1098: Counting the number of bits which are “1” or “0”

   M1099: ON when the bits counting result is “0”.

Program Example:

When X0 = ON with M1098 = ON, MBC instruction counts the number of bit 1 in D0~D2 and store the counted number in D10. When X0 = ON with M1098 = OFF, the instruction counts the number of bit 0 in D0~D2 and store the counted number in D10.



D0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1
D1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1
D2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1

D10	12	M1098=0
D10	36	M1098=1

3.6.18 Contact Type Logic Operation

<table>
<tr><th rowspan="2">API</th><th colspan="2">Mnemonic</th><th rowspan="2">PULSE</th><th rowspan="2">Function</th><th colspan="4">Applicable to</th><th colspan="2">STEPS</th></tr>
<tr><th>16 bits</th><th>32 bits</th><th>ES2
EX2
EC5</th><th>SS2</th><th>SA2
SE</th><th>SX2</th><th>16-bit</th><th>32-bit</th></tr>
<tr><td>215</td><td>LD&</td><td>DLD&</td><td>-</td><td>S_1 & S_2</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>5</td><td>9</td></tr>
<tr><td>216</td><td>LD|</td><td>DLD|</td><td>-</td><td>S_1 | S_2</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>5</td><td>9</td></tr>
<tr><td>217</td><td>LD^</td><td>DLD^</td><td>-</td><td>S_1 ^ S_2</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>5</td><td>9</td></tr>
<tr><td>218</td><td>AND&</td><td>DAND&</td><td>-</td><td>S_1 & S_2</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>5</td><td>9</td></tr>
<tr><td>219</td><td>AND|</td><td>DAND|</td><td>-</td><td>S_1 | S_2</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>5</td><td>9</td></tr>
<tr><td>220</td><td>AND^</td><td>DAND^</td><td>-</td><td>S_1 ^ S_2</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>5</td><td>9</td></tr>
<tr><td>221</td><td>OR&</td><td>DOR&</td><td>-</td><td>S_1 & S_2</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>5</td><td>9</td></tr>
<tr><td>222</td><td>OR|</td><td>DOR|</td><td>-</td><td>S_1 | S_2</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>5</td><td>9</td></tr>
<tr><td>223</td><td>OR^</td><td>DOR^</td><td>-</td><td>S_1 ^ S_2</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>5</td><td>9</td></tr>
</table>

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2 /EC5	SS2	SA2 SE	SX2
215~ 217	D	LD#	S_1 S_2	Contact Type Logic Operation				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*	*	LD#: 5 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DLD#: 9 steps

Bit Devices: X, Y, M, S. Word devices: K, H, KnX, KnY, KnM, KnS, T, C, D, E, F.

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1   **S_2**: Source device 2

Explanations:

1. This instruction conducts logic operation between the content in **S_1** and **S_2**. If the result is not "0", the continuity of the instruction is enabled. If the result is "0", the continuity of the instruction is disabled.

LD# (**#:** &, |, ^) instruction is used for direct connection with Left bus bar.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	Discontinuity condition
215	LD&	**D**LD&	**S_1** & **S_2**≠0	**S_1** & **S_2**＝0
216	LD|	**D**LD|	**S_1** | **S_2**≠0	**S_1** | **S_2**＝0
217	LD^	**D**LD^	**S_1** ^ **S_2**≠0	**S_1** ^ **S_2**＝0

Operation:

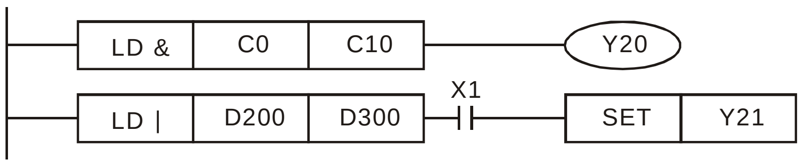
& : Logic "AND" operation, | : Logic "OR" operation, ^ : Logic "XOR" operation

When 32-bit counters (C200 ~ C254) are used in this instruction, make sure to adopt 32-bit instruction (DLD#). If 16-bit instruction (LD#) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When the result of logical AND operation between C0 and C10 ≠ 0, Y20 = ON.

When the result of logical OR operation between D200 and D300 ≠ 0 and X1 = ON, Y21 = ON and latched.



API		Mnemonic		Operands	Function	Controllers			
218~ 220	D	AND#		S1 S2	Serial Type Logic Operation	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*	*	AND#: 5 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DAND#: 9 steps

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1   **S_2**: Source device 2

Explanation:

1. This instruction conducts logic operation between the content in **S_1** and **S_2**. If the result is not "0", the continuity of the instruction is enabled. If the result is "0", the continuity of the instruction is disabled.

AND# (**#:** &, |, ^) instruction is used for serial connection with contacts.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	Discontinuity condition
218	AND&	**D**AND&	**S_1** & **S_2**≠0	**S_1** & **S_2**=0
219	AND|	**D**AND|	**S_1** | **S_2**≠0	**S_1** | **S_2**=0
220	AND^	**D**AND^	**S_1** ^ **S_2**≠0	**S_1** ^ **S_2**=0

Operation:

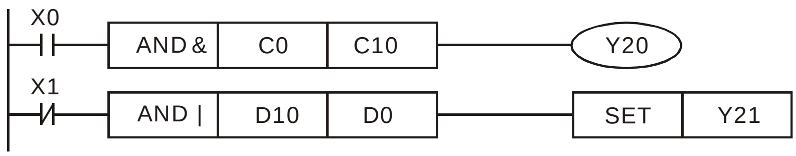
& : Logic "AND" operation, | : Logic "OR" operation, ^ : Logic "XOR" operation

When 32-bit counters (C200 ~ C254) are used in this instruction, make sure to adopt 32-bit instruction (DAND#). If 16-bit instruction (AND#) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash

Program Example:

1. When X0 = ON, and the result of logical AND operation between C0 and C10 ≠ 0, Y20 = ON

When X1 = OFF, and the result of logical OR operation between D10 and D0 ≠ 0, Y21 = ON and latched



API	Mnemonic			Operands	Function	Controllers			
221~223	D	OR#		S1 S2	Parallel Type Logic Operation	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	OR#: 5 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DOR#: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

Explanation:

1. This instruction conducts logic operation between the content in **S_1** and **S_2**. If the result is not "0", the continuity of the instruction is enabled. If the result is "0", the continuity of the instruction is disabled.
2. OR# (**#:** &, |, ^) instruction is used for parallel connection with contacts.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	Discontinuity condition
221	OR&	**D**OR&	**S_1** & **S_2**≠0	**S_1** & **S_2**＝0
222	OR|	**D**OR|	**S_1** | **S_2**≠0	**S_1** | **S_2**＝0
223	OR^	**D**OR^	**S_1** ^ **S_2**≠0	**S_1** ^ **S_2**＝0

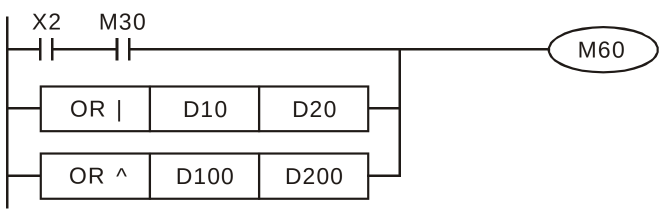
Operation:

& : Logic "AND" operation, | : Logic "OR" operation, ^ : Logic "XOR" operation

1. When 32-bit counters (C200 ~ C254) are used in this instruction, make sure to adopt 32-bit instruction (DOR#). If 16-bit instruction (OR#) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash

Program Example:

M60 will be ON either when both X2 and M30 are "ON", or 1: the result of logical OR operation between D10 and D20 ≠ 0, or 2: the result of logical XOR operation between CD100 and D200 ≠ 0.



3.6.19 Contact Type Comparison

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
224	LD=	DLD=	-	$S_1 = S_2$	✓	✓	✓	✓	5	9
225	LD>	DLD>	-	$S_1 > S_2$	✓	✓	✓	✓	5	9
226	LD<	DLD<	-	$S_1 < S_2$	✓	✓	✓	✓	5	9
228	LD<>	DLD<>	-	$S_1 \neq S_2$	✓	✓	✓	✓	5	9
229	LD<=	DLD<=	-	$S_1 \leqq S_2$	✓	✓	✓	✓	5	9
230	LD>=	DLD>=	-	$S_1 \geqq S_2$	✓	✓	✓	✓	5	9
232	AND=	DAND=	-	$S_1 = S_2$	✓	✓	✓	✓	5	9
233	AND>	DAND>	-	$S_1 > S_2$	✓	✓	✓	✓	5	9
234	AND<	DAND<	-	$S_1 < S_2$	✓	✓	✓	✓	5	9
236	AND<>	DAND<>	-	$S_1 \neq S_2$	✓	✓	✓	✓	5	9
237	AND<=	DAND<=	-	$S_1 \leqq S_2$	✓	✓	✓	✓	5	9
238	AND>=	DAND>=	-	$S_1 \geqq S_2$	✓	✓	✓	✓	5	9
240	OR=	DOR=	-	$S_1 = S_2$	✓	✓	✓	✓	5	9
241	OR>	DOR>	-	$S_1 > S_2$	✓	✓	✓	✓	5	9
242	OR<	DOR<	-	$S_1 < S_2$	✓	✓	✓	✓	5	9
244	OR<>	DOR<>	-	$S_1 \neq S_2$	✓	✓	✓	✓	5	9
245	OR<=	DOR<=	-	$S_1 \leqq S_2$	✓	✓	✓	✓	5	9
246	OR>=	DOR>=	-	$S_1 \geqq S_2$	✓	✓	✓	✓	5	9
296	LDZ>	DLDZ>	-	$\|S_1 - S_2\| > \|S_3\|$	✓	✓	✓	✓	7	13
297	LDZ>=	DLDZ>=	-	$\|S_1 - S_2\| \geqq \|S_3\|$	✓	✓	✓	✓	7	13
298	LDZ<	DLDZ<	-	$\|S_1 - S_2\| < \|S_3\|$	✓	✓	✓	✓	7	13
299	LDZ<=	DLDZ<=	-	$\|S_1 - S_2\| \leqq \|S_3\|$	✓	✓	✓	✓	7	13
300	LDZ=	DLDZ=	-	$\|S_1 - S_2\| = \|S_3\|$	✓	✓	✓	✓	7	13
301	LDZ<>	DLDZ<>	-	$\|S_1 - S_2\| \neq \|S_3\|$	✓	✓	✓	✓	7	13
302	ANDZ>	DANDZ>	-	$\|S_1 - S_2\| > \|S_3\|$	✓	✓	✓	✓	7	13
303	ANDZ>=	DANDZ>=	-	$\|S_1 - S_2\| \geqq \|S_3\|$	✓	✓	✓	✓	7	13
304	ANDZ<	DANDZ<	-	$\|S_1 - S_2\| < \|S_3\|$	✓	✓	✓	✓	7	13
305	ANDZ<=	DANDZ<=	-	$\|S_1 - S_2\| \leqq \|S_3\|$	✓	✓	✓	✓	7	13
306	ANDZ=	DANDZ=	-	$\|S_1 - S_2\| = \|S_3\|$	✓	✓	✓	✓	7	13
307	ANDZ<>	DANDZ<>	-	$\|S_1 - S_2\| \neq \|S_3\|$	✓	✓	✓	✓	7	13
308	ORZ>	DORZ>	-	$\|S_1 - S_2\| > \|S_3\|$	✓	✓	✓	✓	7	13
309	ORZ>=	DORZ>=	-	$\|S_1 - S_2\| \geqq \|S_3\|$	✓	✓	✓	✓	7	13
310	ORZ<	DORZ<	-	$\|S_1 - S_2\| < \|S_3\|$	✓	✓	✓	✓	7	13
311	ORZ<=	DORZ<=	-	$\|S_1 - S_2\| \leqq \|S_3\|$	✓	✓	✓	✓	7	13
312	ORZ=	DORZ=	-	$\|S_1 - S_2\| = \|S_3\|$	✓	✓	✓	✓	7	13
313	ORZ<>	DORZ<>	-	$\|S_1 - S_2\| \neq \|S_3\|$	✓	✓	✓	✓	7	13

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2 /EC5	SS2	SA2 SE	SX2
224~ 230	D	LD※	S_1 S_2	Contact Type Comparison				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*	*	*	LD※: 5 steps DLD※: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

Explanations:

1. This instruction compares the content in **S_1** and **S_2**. Take API224 (LD=) for example, if the result is “=”, the continuity of the instruction is enabled. If the result is “≠”, the continuity of the instruction is disabled.

LD※ (※: =, >, <, <>, ≤, ≥) instruction is used for direct connection with left hand bus bar.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	Discontinuity condition
224	LD＝	**DLD**＝	**$S_1 = S_2$**	**$S_1 \neq S_2$**
225	LD＞	**DLD**＞	**$S_1 > S_2$**	**$S_1 \leqq S_2$**
226	LD＜	**DLD**＜	**$S_1 < S_2$**	**$S_1 \geqq S_2$**
228	LD＜＞	**DLD**＜＞	**$S_1 \neq S_2$**	**$S_1 = S_2$**
229	LD＜＝	**DLD**＜＝	**$S_1 \leqq S_2$**	**$S_1 > S_2$**
230	LD＞＝	**DLD**＞＝	**$S_1 \geqq S_2$**	**$S_1 < S_2$**

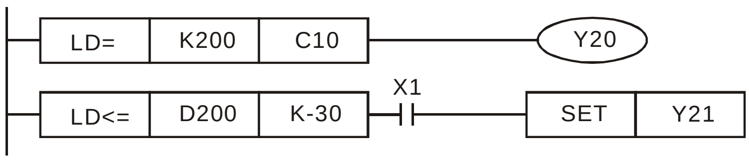
When the MSB (16-bit instruction: b15, 32-bit instruction: b31) of **S_1** and **S_2** is 1, the comparison value will be viewed as a negative value in comparison.

When 32-bit counters (C200 ~ C254) are used in this instruction, make sure to adopt 32-bit instruction (DLD※). If 16-bit instruction (LD※) is adopted, a “program error” will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When the content in C10 = K200, Y20 = ON.

When the content in D200 > K-30 and X1 = ON, Y21 = ON and latched.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2 /EC5	SS2	SA2 SE	SX2
232~ 238	D	AND※	S_1 S_2	Serial Type Comparison				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
S_1					*	*	*	*	*	*	*	*	*	*	*	AND※: 5 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DAND※: 9 steps

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

Explanations:

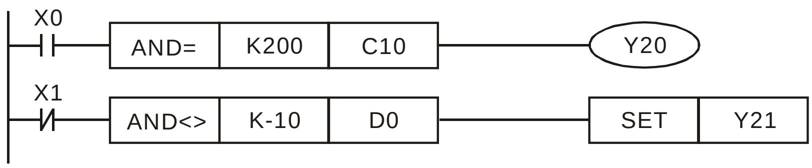
1. This instruction compares the content in **S_1** and **S_2**. Take API232 (AND =) for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled.
2. AND※ (※: =, >, <, <>, ≤, ≥) instruction is used for serial connection with contacts.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	Discontinuity condition
232	AND＝	DAND＝	$S_1 = S_2$	$S_1 \neq S_2$
233	AND＞	DAND＞	$S_1 > S_2$	$S_1 \leq S_2$
234	AND＜	DAND＜	$S_1 < S_2$	$S_1 \geq S_2$
236	AND＜＞	DAND＜＞	$S_1 \neq S_2$	$S_1 = S_2$
237	AND＜＝	DAND＜＝	$S_1 \leq S_2$	$S_1 > S_2$
238	AND＞＝	DAND＞＝	$S_1 \geq S_2$	$S_1 < S_2$

3. When the MSB (16-bit instruction: b15, 32-bit instruction: b31) of **S_1** and **S_2** is 1, the comparison value will be viewed as a negative value in comparison.
4. When 32-bit counters (C200 ~ C254) are used in this instruction, make sure to adopt 32-bit instruction (DAND※). If 16-bit instruction (AND※) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When X0 = ON, and the content in C10 = K200, Y20 = ON
2. When X1 = OFF and the content in D0 ≠ K-10, Y21= ON and latched.



API	Mnemonic		Operands	Function	Controllers			
240~ 246	D	OR※	S_1 S_2	Parallel Type Comparison	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Type OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	OR※: 5 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DOR※: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

Explanations:

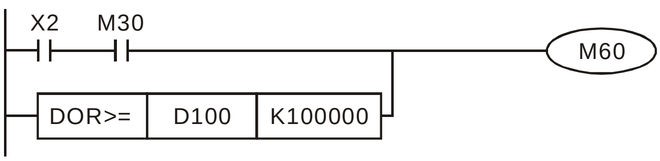
1. This instruction compares the content in **S_1** and **S_2**. Take API240 (OR =) for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled
2. OR※ (※: =, >, <, <>, ≤, ≥) instruction is used for parallel connection with contacts.

API No.	16-bit instruction	32-bit instruction	Continuity condition	Discontinuity condition
240	OR＝	**D**OR＝	$S_1 = S_2$	$S_1 \neq S_2$
241	OR＞	**D**OR＞	$S_1 > S_2$	$S_1 \leqq S_2$
242	OR＜	**D**OR＜	$S_1 < S_2$	$S_1 \geqq S_2$
244	OR＜＞	**D**OR＜＞	$S_1 \neq S_2$	$S_1 = S_2$
245	OR＜＝	**D**OR＜＝	$S_1 \leqq S_2$	$S_1 > S_2$
246	OR＞＝	**D**OR＞＝	$S_1 \geqq S_2$	$S_1 < S_2$

3. When the MSB (16-bit instruction: b15, 32-bit instruction: b31) of **S_1** and **S_2** is 1, the comparison value will be viewed as a negative value in comparison..
4. When 32-bit counters (C200 ~ C254) are used in this instruction, make sure to adopt 32-bit instruction (DOR※). If 16-bit instruction (OR※) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash

Program Example:

M60 will be ON either when both X2 and M30 are "ON", or when the content in 32-bit register D100 (D101) ≥ K100,000.



API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2	SS2	SA2 SE	SX2
258	ATMR	S1 S2	Contact type timer				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																ATMR: 5 steps
S_1											*					
S_2					*	*							*			

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

S_1: Timer number (T0~T255)   **S_2**: Setting value (K0~K32,767, D0~D9,999)。

Explanations:

DVP-ES2/EX2 series PLCs whose version is 3.20/DVP-SS2 series PLCs whose version is 3.00/DVP-SA2 series PLCs whose version is 2.60/DVP-SE series PLCs whose version is 1.20/DVP-SX2 series PLCs whose version is v2.40 (or above) are supported.

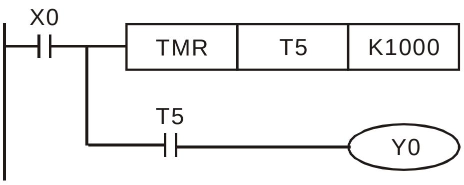
When the instruction ATMR is executed, the coil of the timer specified is driven. When the timer value is equal to the setting value, the state of the normally-open contact is On, and the normally-closed contact is Off.

Normally-open contact	On
Normally-closed contact	Off

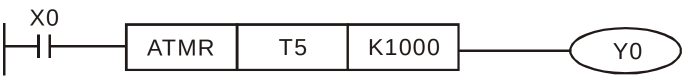
Program Example:

When the normally-open contact X0 is On, the timer T5 begins to measure time intervals. If the timer value is larger than or equal to K1000, the normally-open contact Y0 will be On.

Ladder diagram (The instruction TMR is used.)



Ladder diagram (The instruction ATMR is used.)



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
266	D	BOUT	(D) (n)	Output Specified Bit of a Word				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
D								*	*	*	*	*	*			BOUT: 5 steps
n					*	*	*	*	*	*	*	*	*	*	*	DBOUT: 9 steps

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2/SE	SX2	ES2/EX2	SS2	SA2/SE	SX2	ES2/EX2	SS2	SA2/SE	SX2

Operands:

D: Destination output device **n**: Device specifying the output bit

Explanations:

5. For ES2/EX2 models, only V1.20 or above supports the function.

Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BOUT instruction performs bit output on the output device according to the value specified by operand **n.**

Status of Coils and Associated Contacts:

Evaluation result	BOUT instruction		
	Coil	Associated Contacts	
		NO contact（normally open）	NC contact（normally closed）
FALSE	OFF	Current blocked	Current flows
TRUE	ON	Current flows	Current blocked

Program Example:

Instruction:			Operation:
LDI	X0		Load NC contact X0
AND	X1		Connect NO contact X1 in series.
BOUT	**K4Y0**	**D0**	**When D0 = k1, executes output on Y1 When D0 = k2, executes output on Y2**

API	Mnemonic		Operands	Function	Controllers			
267	D	BSET	(D) (n)	Set ON Specified Bit of a Word	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
D								*	*	*	*	*	*			BSET: 5 steps
n					*	*	*	*	*	*	*	*	*	*	*	DBSET: 9 steps

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

D: Destination device to be Set ON   **n**: Device specifying the bit to be Set ON

Explanations:

6. For ES2/EX2 models, only V1.20 or above supports the function.
7. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

When BSET instruction executes, the output device specified by operand **n** will be ON and latched. To reset the ON state of the device, BRST instruction is required.

Program Example:

Instruction:			Operation:
LDI	X0		Load NC contact X0
AND	X1		Connect NO contact X1 in series.
BSET	**K4Y0**	**D0**	**When D0 = k1, Y1 is ON and latched**
			When D0 = k2, Y2 = ON and latched

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
268	D	BRST	D n	Reset Specified Bit of a Word				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BRST: 5 steps
D								*	*	*	*	*	*			DBRST: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

D: Destination device to be reset **n:** Device specifying the bit to be reset

Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function.
2. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

When BRST instruction executes, the output device specified by operand **n** will be reset (OFF).

Program Example:

Instruction:			Operation:
LD	X0		Load NO contact X0
BRST	**K4Y0**	**D0**	**When D0 = k1,**
			Y1 is OFF
			When D0 = k2,
			Y2 = OFF

API	Mnemonic		Operands	Function	Controllers
269	D	BLD	(S) (n)	Load NO Contact by Specified Bit	ES2/EX2, SS2, SA2 SE, SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
S								*	*	*	*	*	*			BLD: 5 steps
n					*	*	*	*	*	*	*	*	*	*	*	DBLD: 9 steps

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2/SE	SX2	ES2/EX2	SS2	SA2/SE	SX2	ES2/EX2	SS2	SA2/SE	SX2

Operands:

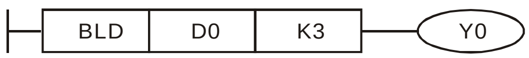
S: Reference source device **n**: Reference bit

Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function.
2. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BLD instruction is used to load NO contact whose contact state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NO contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
BLD	**D0**	**K3**	**Load NO contact with bit status of bit3 in D0**
OUT	Y0		Drive coil Y0

API	Mnemonic		Operands	Function	Controllers			
270	D	BLDI	S n	Load NC Contact by Specified Bit	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S								*	*	*	*	*	*			BLDI: 5 steps DBLDI: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

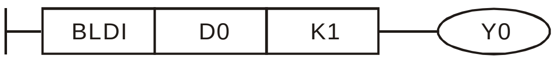
S: Reference source device **n**: Reference bit

Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function.
2. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BLD instruction is used to load NC contact whose contact state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NC contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
BLDI	**D0**	**K1**	**Load NC contact with bit status of bit1 in D0**
OUT	Y0		Drive coil Y0

API	Mnemonic		Operands	Function	Controllers			
271	D	BAND	(S) (n)	Connect NO Contact in Series by Specified Bit	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S								*	*	*	*	*	*			BAND: 5 steps
n					*	*	*	*	*	*	*	*	*	*	*	DBAND: 9 steps

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2/SE	SX2	ES2/EX2	SS2	SA2/SE	SX2	ES2/EX2	SS2	SA2/SE	SX2

Operands:

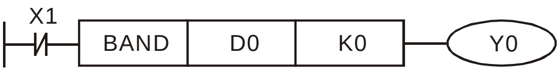
S: Reference source device **n**: Reference bit

Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function.
2. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BAND instruction is used to connect NO contact in series, whose state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NO contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
LDI	X1		Load NC contact X1
BAND	**D0**	**K0**	**Connect NO contact in series , whose state is defined by bit0 of D0**
OUT	Y0		Drive coil Y0

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
272	D	BANI	(S) (n)	Connect NC Contact in Series by Specified Bit				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BANI: 5 steps DBANI: 9 steps
S								*	*	*	*	*	*			
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

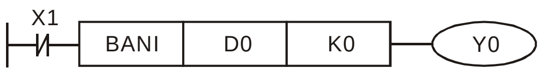
S: Reference source device **n**: Reference bit

Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function
2. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BANI instruction is used to connect NC contact in series, whose state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NC contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
LDI	X1		Load NC contact X1
BANI	**D0**	**K0**	**Connect NC contact in series , whose state is defined by bit0 of D0**
OUT	Y0		Drive coil Y0

API		Mnemonic	Operands	Function	Controllers			
273	D	BOR	(S) (n)	Connect NO Contact in Parallel by Specified Bit	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
S								*	*	*	*	*	*			BOR: 5 steps
n					*	*	*	*	*	*	*	*	*	*	*	DBOR: 9 steps

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2/SE	SX2	ES2/EX2	SS2	SA2/SE	SX2	ES2/EX2	SS2	SA2/SE	SX2

Operands:

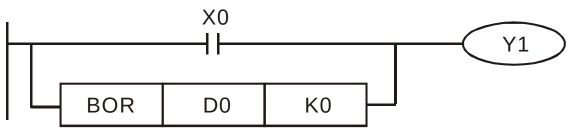
S: Reference source device **n:** Reference bit

Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function.
2. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BOR instruction is used to connect NO contact in parallel, whose state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NO contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
LD	X0		Load NO contact X0
BOR	**D0**	**K0**	**Connect NO contact in parallel, whose state is defined by bit0 of D0**
OUT	Y1		Drive coil Y1

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
274	D	BORI	S n	Connect NC Contact in Parallel by Specified Bit				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BORI: 5 steps
S								*	*	*	*	*	*			DBORI: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

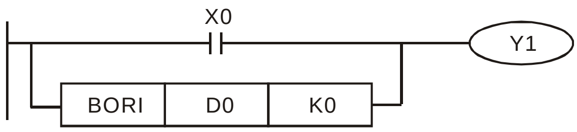
S: Reference source device **n:** Reference bit

Explanations:

1. For ES2/EX2 models, only V1.20 or above supports the function
2. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BORI instruction is used to connect NC contact in parallel, whose state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NC contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
LD	X0		Load NO contact X0
BORI	**D0**	**K0**	**Connect NC contact in parallel, whose state is defined by bit0 of D0**
OUT	Y1		Drive coil Y1

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2	SS2	SA2 SE	SX2
275~ 280	FLD※	S_1 S_2	Floating Point Contact Type Comparison LD※				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1											*	*	*			FLD※: 9 steps
S_2											*	*	*			

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

Explanations:

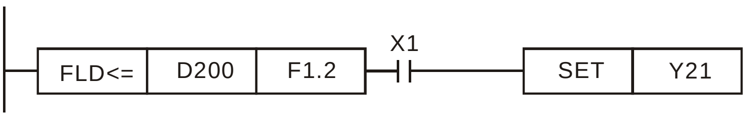
1. This instruction compares the content in **S_1** and **S_2**. Take "FLD=" for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled.
2. The user can specify the floating point value directly into operands **S_1** and **S_2** (e.g. F1.2) or store the floating point value in D registers for further operation.
3. FLD※ instruction is used for direct connection with left hand bus bar.

API No.	32 -bit instruction	Continuity condition	Discontinuity condition
275	FLD＝	$S_1 = S_2$	$S_1 \neq S_2$
276	FLD＞	$S_1 > S_2$	$S_1 \leqq S_2$
277	FLD＜	$S_1 < S_2$	$S_1 \geqq S_2$
278	FLD＜＞	$S_1 \neq S_2$	$S_1 = S_2$
279	FLD＜＝	$S_1 \leqq S_2$	$S_1 > S_2$
280	FLD＞＝	$S_1 \geqq S_2$	$S_1 < S_2$

When either the value in **S_1** or **S_2** comes from external input, or is converted by any editing software, chances are some slight numeral errors may occur and that may lead to not matching "FLD=" or other comparing to be equal instructions. It is suggested to use instructions such as "FLD<" and "FAND>" or "FLD>" and "FAND<" instead.

Program Example:

When the content in D200(D201) ≤ F1.2 and X1 is ON, Y21 = ON and latched.



API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2	SS2	SA2 SE	SX2
281~ 286	FAND※	(S1) (S2)	Floating Point Contact Type Comparison AND※				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
S_1											*	*	*			FAND※: 9 steps
S_2											*	*	*			

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

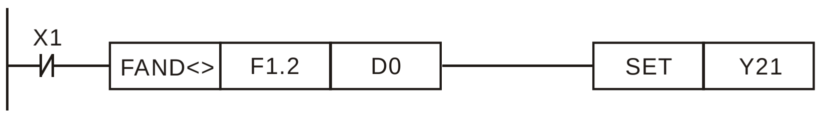
Explanations:

1. This instruction compares the content in **S_1** and **S_2**. Take “FAND =” for example, if the result is “=”, the continuity of the instruction is enabled. If the result is “≠”, the continuity of the instruction is disabled.
2. The user can specify the floating point value directly into operands **S_1** and **S_2** (e.g. F1.2) or store the floating point value in D registers for further operation.
3. FAND※ instruction is used for serial connection with contacts.

API No.	32-bit instruction	Continuity condition	Discontinuity condition
281	FAND＝	$S_1 = S_2$	$S_1 \neq S_2$
282	FAND＞	$S_1 > S_2$	$S_1 \leq S_2$
283	FAND＜	$S_1 < S_2$	$S_1 \geq S_2$
284	FAND＜＞	$S_1 \neq S_2$	$S_1 = S_2$
285	FAND＜＝	$S_1 \leq S_2$	$S_1 > S_2$
286	FAND＞＝	$S_1 \geq S_2$	$S_1 < S_2$

Program Example:

When X1 is OFF and the content in D100(D101) is not equal to F1.2, Y21 = ON and latched.



API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2	SS2	SA2 SE	SX2
287~ 292	FOR※	S1 S2	Floating Point Contact Type Comparison OR※				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1											*	*	*			FOR※: 9 steps
S_2											*	*	*			

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

Explanations:

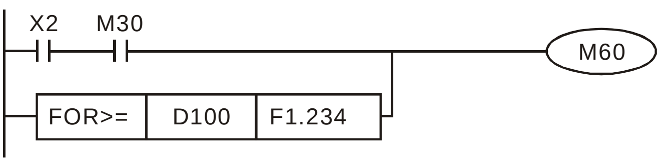
1. This instruction compares the content in **S_1** and **S_2**. Take "FOR =" for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled
2. The user can specify the floating point value directly into operands **S_1** and **S_2** (e.g. F1.2) or store the floating point value in D registers for further operation.

FOR※ instruction is used for parallel connection with contacts.

API No.	32-bit instruction	Continuity condition	Discontinuity condition
287	FOR＝	$S_1 = S_2$	$S_1 \neq S_2$
288	FOR＞	$S_1 > S_2$	$S_1 \leqq S_2$
289	FOR＜	$S_1 < S_2$	$S_1 \geqq S_2$
290	FOR＜＞	$S_1 \neq S_2$	$S_1 = S_2$
291	FOR＜＝	$S_1 \leqq S_2$	$S_1 > S_2$
292	FOR＞＝	$S_1 \geqq S_2$	$S_1 < S_2$

Program Example:

When both X2 and M30 are On and the content in D100(D101) ≥ F1.234, M60 = ON..



API	Mnemonic	Operands	Function	Controllers
295	DMVRW	(S1) (S2) (D1) (D2)	DMV communication command	SS2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DMVRW: 9 steps
S_1													*			
S_2													*			
D_1													*			
D_2		*	*	*												

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

S_1: Communication port on a PLC **S_2**: Function of a DMV **D_1**: Source or destination device

D_2: Communication flag device

Explanations:

1. The models supported are SS2 V3.2 and above.

S_1 specifies a communication port on a PLC for sending/receiving data and station numbers. Only the communication ports on a PLC are supported. Please refer to the description of the PLC used for more information.

S_1+0 ~ **S_1**+3 are described below.

Number	Description	Remark
S_1+0	COM on a PLC	Please refer to the description of a PLC.
S_1+1	Station address of a DMV	Applicable to a serial communication port (RS485/RS232/RS422) K1~K254
S_1+2, **S_1**+3	Reserved	Reserved

Description of **S_1**+0:

Communication port	**S_1**+0	Numbers must be used
COM on a PLC	K1~K5 K1~K5 represent PLC COM1~PLC COM5.	**S_1**+0 ~ **S_1**+1

S_2 is used to set a communication function code. The devices that these operand occupies and the functions of the devices are described below.

Number	Description	Remark
S_2+0	Communication combination function code	Please refer to the description of the function codes below.
S_2+1	Communication address	It is only applicable to K0, and is not applicable to other codes.
S_2+2	Reading/Writing	0: Reading Other values: Writing It is only applicable to K0, and is not applicable to other codes.

Number	Description	Remark
S_2+3	Communication data length	It is used to set the length of the data read/written. A word is a unit of measurement for length. The maximum number of words which can be read/written is 16.

S_2+0: Communication combination function code

Function code	Attribute#1	Function
K0	R or W	There is no communication combination. Users can define a DMV communication command. Please refer to DMO Module Manual for more information about the registers which can be read/written. The data read/written are stored in the devices starting from **D_1**.
K1	W and R	Communication combination commands sent to a DMV#2: 1) DMV trigger 1 is enabled. 2) The value in **S_2**+3 indicates the number of data read from the output data area in a DMV**.** (The maximum number of words which can be read is 16.) The data read is stored in the devices starting from **D_1**.
K2	W	Communication combination commands sent to a DMV: 1) The DMV program number indicated by the value in **D_1** is used. (The value in **D_1** is in the range of 0 to 31.) 2) DMV trigger 1 is enabled.
K3	W and R	Communication combination commands sent to a DMV: 1) The DMV program number indicated by the value in **D_1** is used. (The value in **D_1** is in the range of 0 to 31.) 2) DMV trigger 1 is enabled. 3) The value in **S_2**+3 indicates the number of data read from the output data area in a DMV**.** (The maximum number of words which can be read is 16.) The data read is stored in the devices starting from **D_1**.
K4	W	Communication combination commands sent to a DMV: 1) The values in **D_1**+0 and **D_1**+1 are written into internal memory 1. 2) DMV trigger 1 is enabled.
K5	W and R	Communication combination commands sent to a DMV: 1) The values in **D_1**+0 and **D_1**+1 are written into internal memory 1. 2) DMV trigger 1 is enabled. 3) The value in **S_2**+3 indicates the number of data read from the output data area in a DMV**.** (The maximum number of words which can be read is 16.) The data read is stored in the devices starting from **D_1**.
K6	W	Communication combination commands sent to a DMV: 1) The values in **D_1**+0 and **D_1**+1 are written into internal memory 2. 2) DMV trigger 1 is enabled.
K7	W and R	Communication combination commands sent to a DMV: 1) The values in **D_1**+0 and **D_1**+1 are written into internal memory 2. 2) DMV trigger 1 is enabled. 3) The value in **S_2**+3 indicates the number of data read from the output data area in a DMV**.** (The maximum number of words which can be read is 16.) The data read is stored in the devices starting from **D_1**.

Note#1: W and R mean that a writing communication command is executed first, and then a reading communication command is executed. If the function code used is K3, the **D** operand functions as a source device at first, and then functions as a destination device when a reading command is executed.

Note[#2]: If a communication combination command is used, **S**$_2$+1 and **S**$_2$+2 will be set by the PLC according to the communication combination command.

D$_1$ is a source device or a destination device. Please refer to the description of the function codes above.

D$_2$ is a communication state flag. It occupies three consecutive devices. It is described below.

Number	On	Remark
D$_2$+0	The DMV is busy.	If the DMV is busy, a communication command will be resent automatically until the DMV replies that the communication is complete.
D$_2$+1	The communication with the DMV is complete.	
D$_2$+2	Communication error or timeout	The DMV does not reply after a timeout period.

Whenever the instruction is enabled, the PLC automatically reset **D**$_2$ to Off.

Example 1: Users define a DMV communication command. COM2 on a PLC communicates with a DMV. H0888 is written into the communication address H10D0 in the DMV. The control procedure is described below.

1-1. Write K2 into D0. (COM2 on the PLC is used.) Write K1 into D1. (The station address of the DMV is K1.)

1-2. Write K0 into D4. The users define a DMV communication command by themselves, and write the command message into D5~D7.

Operand	Device	Value	Description
S$_2$+0	D4	K0	Communication combination function code
S$_2$+1	D5	H10D0	Communication address
S$_2$+2	D6	K1	Reading/Writing
S$_2$+3	D7	K1	Communication data length

1-3. When M0 is On, the PLC communicates with the DMV according to the communication data and the communication port set by the users, and H0888 in D8 is written into H10D0 in the DMV.

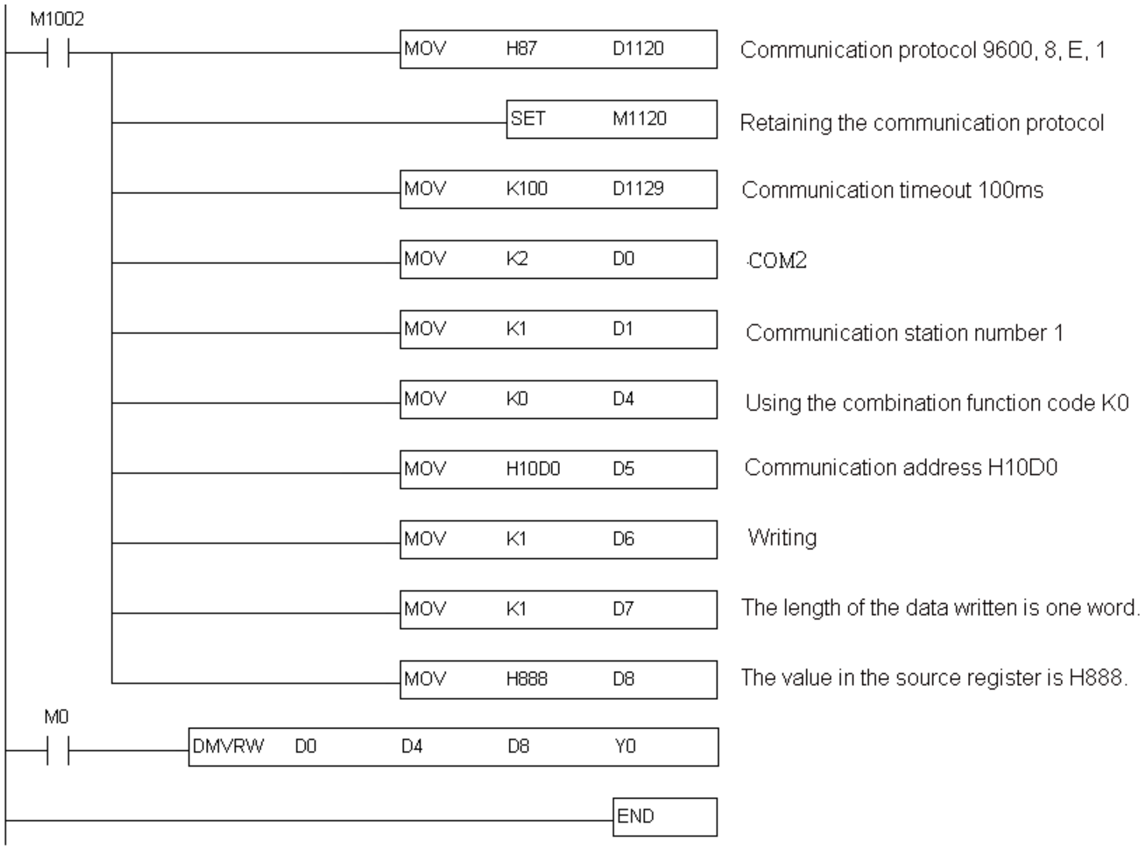
1-4. When the PLC sends the data, the operand **D**$_2$ (Y0) is On (the DMV is busy).

1-5. When the DMV replies successfully, **D**$_2$+1 (Y1) in the PLC is On (the communication with the DMV is complete).

1-6. If the DMV does not reply after the timeout period 100ms, the PLC will set **D**$_2$+2 (Y2) to On (a communication timeout occurs).

1-7. If the DMV replies with an execption code, the PLC will resend the command to the DMV automatically, and go back to step 1-3 ~ step 1-5.

The program in the PLC and the comments are shown below.



Example 2: The combination function code K3 is used. COM2 on a PLC communicates with a DMV. The control procedure is shown below.

2-1. Write K2 into D0. (COM2 on the PLC is used.) Write K1 into D1. (The station address of the DMV is K1.)

2-2. The operand $\mathbf{S_2}$+0 specifies D4. Write K3 into D4. The function code K3 is used (There are three communication commands.)The message required are written into $\mathbf{S_2}$+3 and $\mathbf{D_1}$.

Communication command	Operand	Device	Value	Description
First	$\mathbf{D_1}$	D8	H0014	The DMV program number used is K20.
Second	-	-	-	It does not need to be set. The PLC enables DMV trigger 1 by itself.
Third	$\mathbf{S_2}$+3	D7	K2	The value in $\mathbf{S_2}$+3 indicates the number of data read from the output data area in a DMV.

2-3. When M0 is ON, the PLC sends communication data to the DMV accoding to the communication combination command order specified by the function code K3.

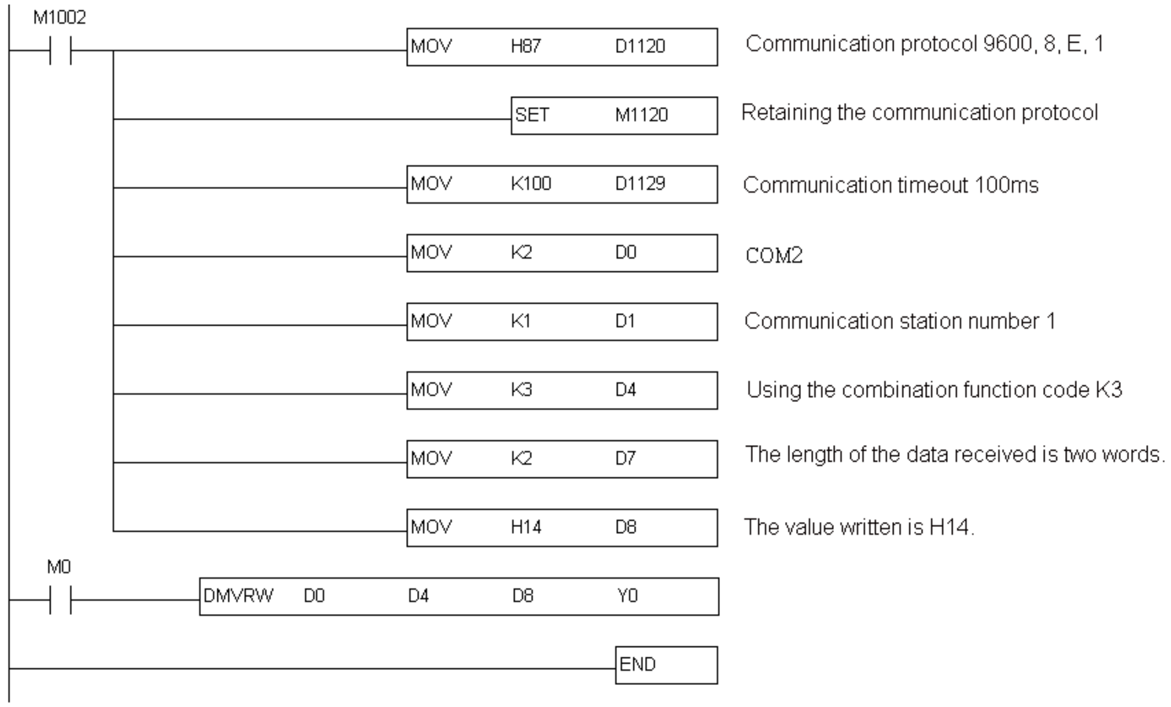
2-4. When the PLC sends the data, the operand $\mathbf{D}_2$ (Y0) is On (the DMV is busy).

2-5. When the DMV replies to the three communication commands successfully, $\mathbf{D}_2$+1 (Y1) in the PLC is On (the communication with the DMV is complete).

2-6. If the DMV does not reply after the timeout period 100ms, the PLC will set **D**$_2$+2 (Y2) to On (a communication timeout occurs).

2-7. If the DMV replies with an execption code, the PLC will resend the command to the DMV automatically, and go back to step 2-3 ~ step 2-5.

The program in the PLC and the comments are shown below.



Remark: D8 in the example is described below.

3-1 When the first command is sent, the value in D8 indicates a program number. In the example, program number 20 is used, and therefore H14 (or K20) is written into D8 in advance.

3-2 The the third command is sent, D8 becomes a start device in which data received from the DMV is stored. In the example, two-word data is read. When the completion flag is ON, the data read is stored in D8 and D9.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/EC5	SS2	SA2 SE	SX2
296~301	D	LDZ※	S_1 S_2 S_3	Comparing contact type absolute values LDZ※				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	LDZ※: 7 steps DLDZ※: 13 steps
S_1					*	*	*	*	*	*	*	*	*			
S_2					*	*	*	*	*	*	*	*	*			
S_3					*	*	*	*	*	*	*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/EC5	SS2	SA2/SE	SX2	ES2/EX2/EC5	SS2	SA2/SE	SX2	ES2/EX2/EC5	SS2	SA2/SE	SX2

Operands:

S_1: Source device 1 S_2: Source device 2 S_3: Source device 3

Explanations:

1. DVP-ES2/EX2 series PLCs whose version is 3.20/DVP-SS2 series PLCs whose version is 3.00/DVP-SA2 series PLCs whose version is 2.60/DVP-SE series PLCs whose version is 1.20/DVP-SX2 series PLCs whose version is 2.40 (or above) are supported.

The absolute value of the difference between S_1 and S_2 is compared with the absolute value of S_3. Take LDZ= for example. If the comparison result is that the absolute value of the difference between S_1 and S_2 is equal to the absolute value of S_3, the condition of the instruction is met. If the comparison result is that the absolute value of the difference between S_1 and S_2 is not equal to the absolute value of S_3, the condition of the instruction is not met.

The instruction can be connected to a busbar.

API No.	16-bit instruction	32-bit instruction	Comparison result	
			On	Off
296	LDZ＞	DLDZ＞	$\|S_1 - S_2\| > \|S_3\|$	$\|S_1 - S_2\| \leqq \|S_3\|$
297	LDZ＞＝	DLDZ＞＝	$\|S_1 - S_2\| \geqq \|S_3\|$	$\|S_1 - S_2\| < \|S_3\|$
298	LDZ＝	DLDZ＜	$\|S_1 - S_2\| < \|S_3\|$	$\|S_1 - S_2\| \geqq \|S_3\|$
299	LDZ＜＝	DLDZ＜＝	$\|S_1 - S_2\| \leqq \|S_3\|$	$\|S_1 - S_2\| > \|S_3\|$
300	LDZ＝	DLDZ＝	$\|S_1 - S_2\| = \|S_3\|$	$\|S_1 - S_2\| \neq \|S_3\|$
301	LDZ＜＞	DLDZ＜＞	$\|S_1 - S_2\| \neq \|S_3\|$	$\|S_1 - S_2\| = \|S_3\|$

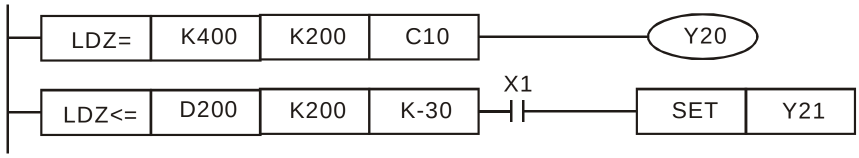
If the values of the most significant bits in S_1, S_2, and S_3 are 1, the values in S_1, S_2, and S_3 are negative values.

A 32-bit counter (C200~) must be used with the 32-bit instruction DLDZ※. If it is used with the 16-bit instruction LDZ※, a program error will occur, and the ERROR LED indicator on the PLC will blink.

Program Example:

1. If the value in C10 is equal to K200 or K-200, Y20 will be On.

If the value in D200 is less than or equal to K230, and is larger than or equal to K170, and X1 is On, Y21 will be On and latched.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC8	SS2	SA2 SE	SX2
302~ 307	D	ANDZ※	S1 S2 S3	Comparing contact type absolute values ANDZ※				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*	*	*	*	*	*	*	*			ANDZ※: 7 steps
S_2					*	*	*	*	*	*	*	*	*			DANDZ※: 13 steps
S_3					*	*	*	*	*	*	*	*	*			

(Bit Devices: X, Y, M, S; Word devices: K, H, KnX, KnY, KnM, KnS, T, C, D, E, F)

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2 **S_3**: Source device 3

Explanations:

1. DVP-ES2/EX2 series PLCs whose version is 3.20/DVP-SS2 series PLCs whose version is 3.00/DVP-SA2 series PLCs whose version is 2.60/DVP-SE series PLCs whose version is 1.20/DVP-SX2 series PLCs whose version is 2.40 (or above) are supported.

The absolute value of the difference between **S_1** and **S_2** is compared with the absolute value of **S_3**. Take AND= for example. If the comparison result is that the absolute value of the difference between **S_1** and **S_2** is equal to the absolute value of **S_3**, the condition of the instruction is met. If the comparison result is that the absolute value of the difference between **S_1** and **S_2** is not equal to the absolute value of **S_3**, the condition of the instruction is not met.

The instruction ANDZ※ is connected to a contact in series.

API No.	16-bit instruction	32-bit instruction	Comparison result	
			On	Off
302	ANDZ＞	DANDZ＞	$\lvert S_1 - S_2 \rvert > \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert \leqq \lvert S_3 \rvert$
303	ANDZ＞＝	DANDZ＞＝	$\lvert S_1 - S_2 \rvert \geqq \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert < \lvert S_3 \rvert$
304	ANDZ＜	DANDZ＜	$\lvert S_1 - S_2 \rvert < \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert \geqq \lvert S_3 \rvert$
305	ANDZ＜＝	DANDZ＜＝	$\lvert S_1 - S_2 \rvert \leqq \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert > \lvert S_3 \rvert$
306	ANDZ＝	DANDZ＝	$\lvert S_1 - S_2 \rvert = \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert \neq \lvert S_3 \rvert$
307	ANDZ＜＞	DANDZ＜＞	$\lvert S_1 - S_2 \rvert \neq \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert = \lvert S_3 \rvert$

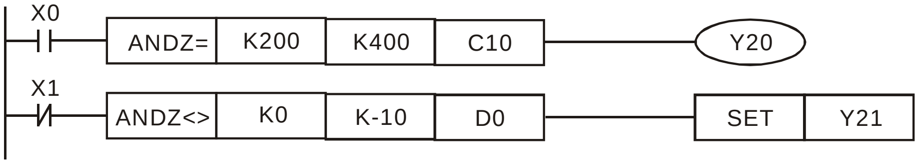
If the values of the most significant bits in **S_1**, **S_2**, and **S_3** are 1, the values in **S_1**, **S_2**, and **S_3** are negative values.

A 32-bit counter (C200~) must be used with the 32-bit instruction DANDZ※. If it is used with the 16-bit instruction ANDZ※, a program error will occur, and the ERROR LED indicator on the PLC will blink.

Program Example:

1. If X0 is On, and the present value in C10 is equal to K200 or K-200, Y20 will be On.

If X1 is Off, and the value in D0 is not equal to K10 or K-10, Y21 will be On and latched.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2/ EC5	SS2	SA2 SE	SX2
308~ 313	D	ORZ※	S_1 S_2 S_3	Comparing contact type absolute values ORZ※				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ORZ※: 7 steps
S_1					*	*	*	*	*	*	*	*	*			DORZ※: 13 steps
S_2					*	*	*	*	*	*	*	*	*			
S_3					*	*	*	*	*	*	*	*	*			

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2 **S_3**: Source device 3

Explanations:

1. DVP-ES2/EX2 series PLCs whose version is 3.20/DVP-SS2 series PLCs whose version is 3.00/DVP-SA2 series PLCs whose version is 2.60/DVP-SE series PLCs whose version is 1.20/DVP-SX2 series PLCs whose version is 2.40 (or above) are supported.

The absolute value of the difference between **S_1** and **S_2** is compared with the absolute value of **S_3**. Take ORZ= for example. If the comparison result is that the absolute value of the difference between **S_1** and **S_2** is equal to the absolute value of **S_3**, the condition of the instruction is met. If the comparison result is that the absolute value of the difference between **S_1** and **S_2** is not equal to the absolute value of **S_3**, the condition of the instruction is not met.

The instruction ANDZ※ is connected to a contact in parallel.

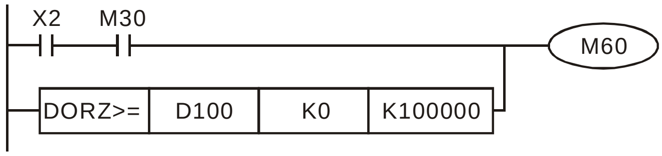
API No.	16-bit instruction	32-bit instruction	Comparison result	
			On	Off
308	ORZ＞	DORZ＞	$\lvert S_1 - S_2 \rvert > \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert \leqq \lvert S_3 \rvert$
309	ORZ＞＝	DORZ＞＝	$\lvert S_1 - S_2 \rvert \geqq \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert < \lvert S_3 \rvert$
310	ORZ＜	DORZ＜	$\lvert S_1 - S_2 \rvert < \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert \geqq \lvert S_3 \rvert$
311	ORZ＜＝	DORZ＜＝	$\lvert S_1 - S_2 \rvert \leqq \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert > \lvert S_3 \rvert$
312	ORZ＝	DORZ＝	$\lvert S_1 - S_2 \rvert = \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert \neq \lvert S_3 \rvert$
313	ORZ＜＞	DORZ＜＞	$\lvert S_1 - S_2 \rvert \neq \lvert S_3 \rvert$	$\lvert S_1 - S_2 \rvert = \lvert S_3 \rvert$

If the values of the most significant bits in **S_1**, **S_2**, and **S_3** are 1, the values in **S_1**, **S_2**, and **S_3** are negative values.

A 32-bit counter (C200~) must be used with the 32-bit instruction DORZ※. If it is used with the 16-bit instruction ORZ※, a program error will occur, and the ERROR LED indicator on the PLC will blink.

Program Example:

If X2 and M30 are On, or the value in the 32-bit register (D101, D100) is larger than or equal to K100000, or is less than or equal to K-100000, M60 will be On.



3.6.20 Specific Bit Control

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
266	BOUT	DBOUT	-	Output specified bit of a word	✓	✓	✓	✓	5	9
267	BSET	DBSET	-	Set ON specified bit of a word	✓	✓	✓	✓	5	9
268	BRST	DBRST	-	Reset specified bit of a word	✓	✓	✓	✓	5	9
269	BLD	DBLD	-	Load NO contact by specified bit	✓	✓	✓	✓	5	9
270	BLDI	DBLDI	-	Load NC contact by specified bit	✓	✓	✓	✓	5	9
271	BAND	DBAND	-	Connect NO contact in series by specified bit	✓	✓	✓	✓	5	9
272	BANI	DBANI	-	Connect NC contact in series by specified bit	✓	✓	✓	✓	5	9
273	BOR	DBOR	-	Connect NO contact in parallel by specified bit	✓	✓	✓	✓	5	9
274	BORI	DBORI	-	Connect NC contact in parallel by specified bit	✓	✓	✓	✓	5	9

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2 /EC5	SS2	SA2 SE	SX2
266	D	BOUT	(D) (n)	Output Specified Bit of a Word				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BOUT: 5 steps
D								*	*	*	*	*	*			DBOUT: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

D: Destination output device **n**: Device specifying the output bit

Explanations:

2. For ES2/EX2 models, only V1.20 or above supports the function.

Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BOUT instruction performs bit output on the output device according to the value specified by operand **n.**

Status of Coils and Associated Contacts:

Evaluation result	BOUT instruction		
	Coil	Associated Contacts	
		NO contact（normally open）	NC contact（normally closed）
FALSE	OFF	Current blocked	Current flows
TRUE	ON	Current flows	Current blocked

Program Example:

Instruction:			Operation:
LDI	X0		Load NC contact X0
AND	X1		Connect NO contact X1 in series.
BOUT	**K4Y0**	**D0**	**When D0 = k1, executes output on Y1** **When D0 = k2, executes output on Y2**

API	Mnemonic		Operands	Function	Controllers			
267	D	BSET	(D) (n)	Set ON Specified Bit of a Word	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BSET: 5 steps
D								*	*	*	*	*	*			DBSET: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

D: Destination device to be Set ON **n**: Device specifying the bit to be Set ON

Explanations:

3. For ES2/EX2 models, only V1.20 or above supports the function.
4. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

When BSET instruction executes, the output device specified by operand **n** will be ON and latched. To reset the ON state of the device, BRST instruction is required.

Program Example:

Instruction:			Operation:
LDI	X0		Load NC contact X0
AND	X1		Connect NO contact X1 in series.
BSET	**K4Y0**	**D0**	**When D0 = k1, Y1 is ON and latched When D0 = k2, Y2 = ON and latched**

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2 /EC5	SS2	SA2 SE	SX2
268	D	**BRST**	(D) (n)	Reset Specified Bit of a Word				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
D								*	*	*	*	*	*			BRST: 5 steps
n					*	*	*	*	*	*	*	*	*	*	*	DBRST: 9 steps

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

D: Destination device to be reset **n:** Device specifying the bit to be reset

Explanations:

5. For ES2/EX2 models, only V1.20 or above supports the function.
6. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

When BRST instruction executes, the output device specified by operand **n** will be reset (OFF).

Program Example:

Instruction:			Operation:
LD	X0		Load NO contact X0
BRST	**K4Y0**	**D0**	**When D0 = k1,**
			Y1 is OFF
			When D0 = k2,
			Y2 = OFF

API	Mnemonic		Operands	Function	Controllers			
269	D	BLD	(S) (n)	Load NO Contact by Specified Bit	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BLD: 5 steps
S								*	*	*	*	*	*			DBLD: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

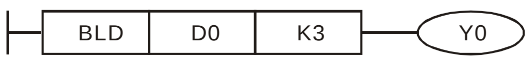
S: Reference source device **n**: Reference bit

Explanations:

7. For ES2/EX2 models, only V1.20 or above supports the function.
8. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BLD instruction is used to load NO contact whose contact state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NO contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
BLD	**D0**	**K3**	**Load NO contact with bit status of bit3 in D0**
OUT	Y0		Drive coil Y0

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2 /EC5	SS2	SA2 SE	SX2
270	D	**BLDI**	S n	Load NC Contact by Specified Bit				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BLDI: 5 steps
S								*	*	*	*	*	*			DBLDI: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

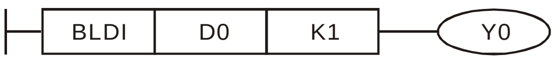
S: Reference source device **n**: Reference bit

Explanations:

9. For ES2/EX2 models, only V1.20 or above supports the function.
10. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BLD instruction is used to load NC contact whose contact state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NC contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
BLDI	**D0**	**K1**	**Load NC contact with bit status of bit1 in D0**
OUT	Y0		Drive coil Y0

API		Mnemonic	Operands	Function	Controllers			
					ES2/EX2 /EC5	SS2	SA2 SE	SX2
271	D	**BAND**	(S) (n)	Connect NO Contact in Series by Specified Bit				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BAND: 5 steps
S								*	*	*	*	*	*			DBAND: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

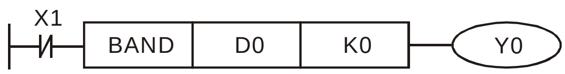
S: Reference source device **n**: Reference bit

Explanations:

11. For ES2/EX2 models, only V1.20 or above supports the function.
12. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BAND instruction is used to connect NO contact in series, whose state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NO contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
LDI	X1		Load NC contact X1
BAND	**D0**	**K0**	**Connect NO contact in series , whose state is defined by bit0 of D0**
OUT	Y0		Drive coil Y0

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2 /EC5	SS2	SA2 SE	SX2
272	D	BANI	(S) (n)	Connect NC Contact in Series by Specified Bit				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BANI: 5 steps
S								*	*	*	*	*	*			DBANI: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

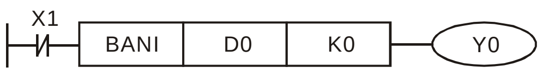
S: Reference source device   **n**: Reference bit

Explanations:

13. For ES2/EX2 models, only V1.20 or above supports the function
14. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BANI instruction is used to connect NC contact in series, whose state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NC contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
LDI	X1		Load NC contact X1
BANI	**D0**	**K0**	**Connect NC contact in series , whose state is defined by bit0 of D0**
OUT	Y0		Drive coil Y0

API	Mnemonic		Operands	Function	Controllers			
273	D	BOR	(S) (n)	Connect NO Contact in Parallel by Specified Bit	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BOR: 5 steps
S								*	*	*	*	*	*			DBOR: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

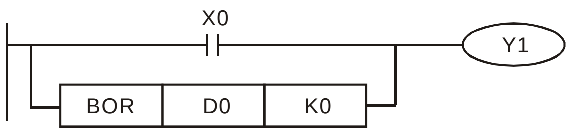
S: Reference source device **n:** Reference bit

Explanations:

15. For ES2/EX2 models, only V1.20 or above supports the function.
16. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BOR instruction is used to connect NO contact in parallel, whose state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NO contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
LD	X0		Load NO contact X0
BOR	**D0**	**K0**	**Connect NO contact in parallel, whose state is defined by bit0 of D0**
OUT	Y1		Drive coil Y1

API	Mnemonic		Operands	Function	Controllers			
274	D	BORI	S n	Connect NC Contact in Parallel by Specified Bit	ES2/EX2 /EC5	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BORI: 5 steps
S								*	*	*	*	*	*			DBORI: 9 steps
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

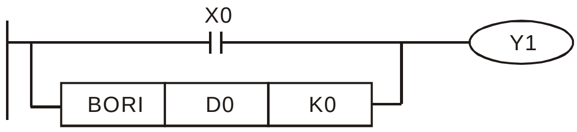
S: Reference source device **n:** Reference bit

Explanations:

17. For ES2/EX2 models, only V1.20 or above supports the function
18. Available range for the value in operand **n**: K0~K15 for 16-bit instruction; K0~K31 for 32-bit instruction.

BORI instruction is used to connect NC contact in parallel, whose state is defined by the reference bit **n** in reference device **D**, i.e. if the bit specified by **n** is ON, the NC contact will be ON, and vice versa.

Program Example:



Instruction:			Operation:
LD	X0		Load NO contact X0
BORI	**D0**	**K0**	**Connect NC contact in parallel, whose state is defined by bit0 of D0**
OUT	Y1		Drive coil Y1

3.6.21 Floating-Point Contact Type Comparison

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2 EC5	SS2	SA2 SE	SX2	16-bit	32-bit
275	-	FLD=	-	$S_1 = S_2$	✓	✓	✓	✓	-	9
276	-	FLD>	-	$S_1 > S_2$	✓	✓	✓	✓	-	9
277	-	FLD<	-	$S_1 < S_2$	✓	✓	✓	✓	-	9
278	-	FLD<>	-	$S_1 \neq S_2$	✓	✓	✓	✓	-	9
279	-	FLD<=	-	$S_1 \leqq S_2$	✓	✓	✓	✓	-	9
280	-	FLD>=	-	$S_1 \geqq S_2$	✓	✓	✓	✓	-	9
281	-	FAND=	-	$S_1 = S_2$	✓	✓	✓	✓	-	9
282	-	FAND>	-	$S_1 > S_2$	✓	✓	✓	✓	-	9
283	-	FAND<	-	$S_1 < S_2$	✓	✓	✓	✓	-	9
284	-	FAND<>	-	$S_1 \neq S_2$	✓	✓	✓	✓	-	9
285	-	FAND<=	-	$S_1 \leqq S_2$	✓	✓	✓	✓	-	9
286	-	FAND>=	-	$S_1 \geqq S_2$	✓	✓	✓	✓	-	9
287	-	FOR=	-	$S_1 = S_2$	✓	✓	✓	✓	-	9
288	-	FOR>	-	$S_1 > S_2$	✓	✓	✓	✓	-	9
289	-	FOR<	-	$S_1 < S_2$	✓	✓	✓	✓	-	9
290	-	FOR<>	-	$S_1 \neq S_2$	✓	✓	✓	✓	-	9
291	-	FOR<=	-	$S_1 \leqq S_2$	✓	✓	✓	✓	-	9
292	-	FOR>=	-	$S_1 \geqq S_2$	✓	✓	✓	✓	-	9

API	Mnemonic	Operands	Function	Controllers			
275~280	FLD※	S_1 S_2	Floating Point Contact Type Comparison LD※	ES2/EX2/EC5	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1											*	*	*			FLD※: 9 steps
S_2											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2/EC5	SS2	SA2/SE	SX2	ES2/EX2/EC5	SS2	SA2/SE	SX2	ES2/EX2/EC5	SS2	SA2/SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

Explanations:

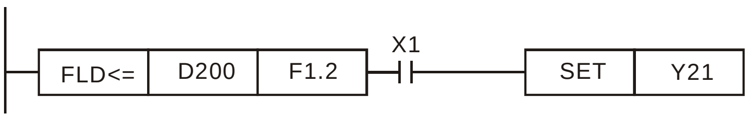
1. This instruction compares the content in **S_1** and **S_2**. Take "FLD=" for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled.
2. The user can specify the floating point value directly into operands **S_1** and **S_2** (e.g. F1.2) or store the floating point value in D registers for further operation.
3. FLD※ instruction is used for direct connection with left hand bus bar.

API No.	32 -bit instruction	Continuity condition	Discontinuity condition
275	FLD＝	**$S_1 = S_2$**	**$S_1 \neq S_2$**
276	FLD＞	**$S_1 > S_2$**	**$S_1 \leqq S_2$**
277	FLD＜	**$S_1 < S_2$**	**$S_1 \geqq S_2$**
278	FLD＜＞	**$S_1 \neq S_2$**	**$S_1 = S_2$**
279	FLD＜＝	**$S_1 \leqq S_2$**	**$S_1 > S_2$**
280	FLD＞＝	**$S_1 \geqq S_2$**	**$S_1 < S_2$**

When either the value in **S_1** or **S_2** comes from external input, or is converted by any editing software, chances are some slight numeral errors may occur and that may lead to not matching "FLD=" or other comparing to be equal instructions. It is suggested to use instructions such as "FLD<" and "FAND>" or "FLD>" and "FAND<" instead.

Program Example:

When the content in D200(D201) ≤ F1.2 and X1 is ON, Y21 = ON and latched.



API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2 /EC5	SS2	SA2 SE	SX2
281~ 286	FAND※	S_1 S_2	Floating Point Contact Type Comparison AND※				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	FAND※: 9 steps
S_1											*	*	*			
S_2											*	*	*			

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

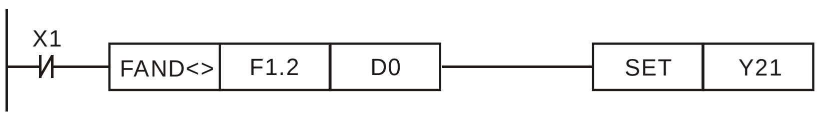
Explanations:

1. This instruction compares the content in **S_1** and **S_2**. Take "FAND =" for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled.
2. The user can specify the floating point value directly into operands **S_1** and **S_2** (e.g. F1.2) or store the floating point value in D registers for further operation.
3. FAND※ instruction is used for serial connection with contacts.

API No.	32-bit instruction	Continuity condition	Discontinuity condition
281	FAND＝	**$S_1 = S_2$**	**$S_1 \neq S_2$**
282	FAND＞	**$S_1 > S_2$**	**$S_1 \leqq S_2$**
283	FAND＜	**$S_1 < S_2$**	**$S_1 \geqq S_2$**
284	FAND＜＞	**$S_1 \neq S_2$**	**$S_1 = S_2$**
285	FAND＜＝	**$S_1 \leqq S_2$**	**$S_1 > S_2$**
286	FAND＞＝	**$S_1 \geqq S_2$**	**$S_1 < S_2$**

Program Example:

When X1 is OFF and the content in D100(D101) is not equal to F1.2, Y21 = ON and latched.



API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2 /EC5	SS2	SA2 SE	SX2
287~ 292	FOR※	S1 S2	Floating Point Contact Type Comparison OR※				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	FOR※: 9 steps
S_1											*	*	*			
S_2											*	*	*			

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

Explanations:

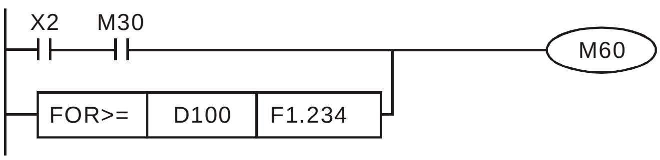
1. This instruction compares the content in **S_1** and **S_2**. Take "FOR =" for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled
2. The user can specify the floating point value directly into operands **S_1** and **S_2** (e.g. F1.2) or store the floating point value in D registers for further operation.

FOR※ instruction is used for parallel connection with contacts.

API No.	32-bit instruction	Continuity condition	Discontinuity condition
287	FOR＝	$S_1 = S_2$	$S_1 \neq S_2$
288	FOR＞	$S_1 > S_2$	$S_1 \leq S_2$
289	FOR＜	$S_1 < S_2$	$S_1 \geq S_2$
290	FOR＜＞	$S_1 \neq S_2$	$S_1 = S_2$
291	FOR＜＝	$S_1 \leq S_2$	$S_1 > S_2$
292	FOR＞＝	$S_1 \geq S_2$	$S_1 < S_2$

Program Example:

When both X2 and M30 are On and the content in D100(D101) ≥ F1.234, M60 = ON..



3.6.22 Delta Special CANopen Communication Instructions

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2-C	SS2	SA2 SE	SX2	16-bit	32-bit
328	INITC	-	-	Initializing the servos for CANopen communication	✓	-	-	-	3	-
329	ASDON	-	-	Servo-ON and servo-OFF	✓	-	-	-	5	-
330	CASD	-	-	Setting the acceleration time and deceleration time for a servo	✓	-	-	-	7	-
331	-	DDRVIC	-	Servo relative position control	✓	-	-	-	-	13
332	-	DDRVAC	-	Servo absolute position control	✓	-	-	-	-	13
333	PLSVC	DPLSVC	-	Servo speed control	✓	-	-	-	5	9
334	ZRNC	DZRNC	-	Homing	✓	-	-	-	7	13
335	COPWL	DCOPWL	-	Writing and reading CANopen communication data	✓	-	-	-	9	17
336	RSTD	-	-	Sending Reset or NMT command	✓	-	-	-	9	-
338	EMER	-	-	Reading Emergency message	✓	-	-	-	11	-
339	ZRNM	-	-	Setting the homing mode for Delta servo drive	✓	-	-	-	9	-
340	CANRS	-	-	User-defined CAN communication sending and receiving	✓	-	-	-	11	-
342	COPRW	-	-	Writing and reading CANopen communication data	✓	-	-	-	13	-

API	Mnemonic	Operands	Function	Controllers
328	INITC	S_1	Initialization for Delta servo drive communication	ES2-C

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*										INITC: 3 steps

PULSE	16-bit	32-bit
--	ES2-C	--

Operands

S_1 Number of station to be initialized

Explanation

1. Before executing the instruction, be sure to set M1614 to ON to enable Delta servo drive function. ASDA-A2 is available for DVP-ES2 with firmware V3.48 or later; for ASDA-A3, it is available for DVP-ES2 with firmware V3.60 or later.

M1614	ON: Delta servo drive function
	OFF: CANopen DS301 mode

After the mode is changed, you need to restart the PLC to activate the setting.

2. It is not available for pulse type instructions. Do not use pulse type contact.

3. The servo range of **S** is K1–K8. When the input value is greater than 8, PLC will automatically process 8 as the value of **S** for the initialization. The station address must start at 1 and the following addresses cannot be skipped or reserved. For example, if S_1 is set to K4, this instruction initializes K1 to K4.

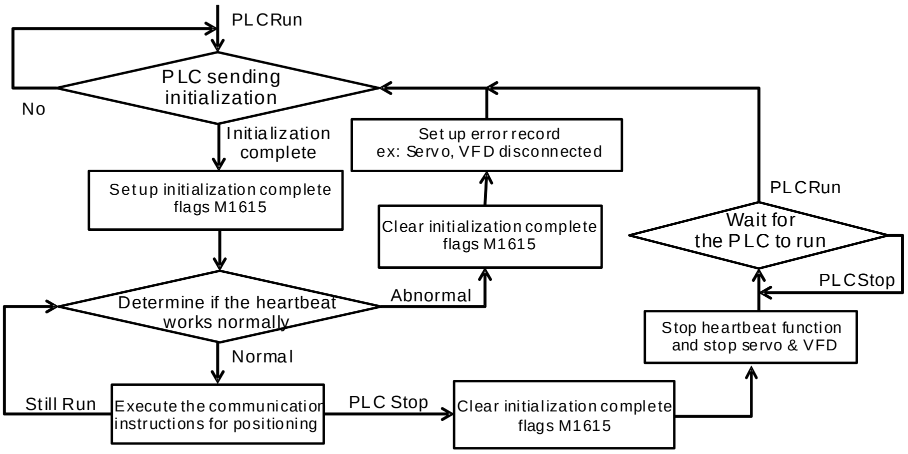
4. Once the instruction is executed, M1615 will be set to OFF. After execution is done, M1615 will be set to ON.

5. Any error occurs during operation, M1616 will be set to ON and the servo drive number that shows error will be stored in D6000, error codes in D6001 and STEP that when error occurs in D6002.

6. Added the heartbeat function for firmware V3.60 or later. If a heartbeat error (M1067 = ON, D1067 = 0x1901~0x1908; the last 2 codes are the slave ID) occurs after the initialization is complete and the M1617 is OFF (default, indicating when one goes down, all the drivers are

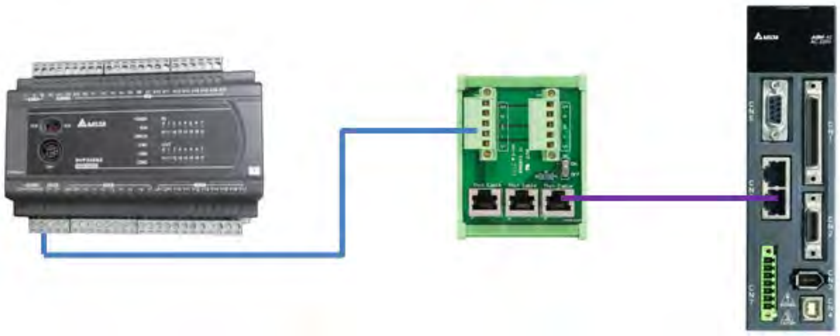
OFF), the initialization complete flag (M1615) will be cleared to OFF and related actions on other slaves will also be paused. After all the troubles are cleared, you need to initialize every slave to restart the operation. If the axes are working independently and the communication is working properly, you can set the M1617 to ON (indicating when one goes down, only the defective driver is OFF) to notify PLC to record the specific error on the error log and other slaves can keep working.

Initialization and operation process chart (Firmware V3.60 or later)



Example of Communication with Delta servo ASDA

1. Connect the ES2-C Series PLC to TAP-CN03 and an ASDA series with a CANopen communication cable as shown in the figure below.



2. Follow the steps below for the basic settings on the panel of the ASDA.

   a. Set the servo parameter P2-08 to 10 to restore the factory settings.

   b. Power the servo off and back on again.

   c. Set P1-01 to 0001 (PR mode).

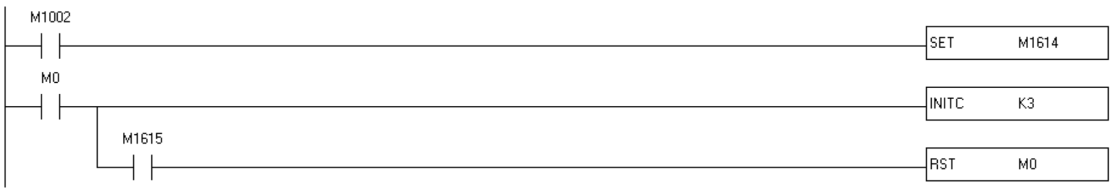
d. Set P3-01 to 0400 and the baud rate of the servo for CANopen communication to 1.0 Mbit/S. The baud rate must be the same as that of the PLC. For firmware V3.48 or earlier versions, the baud rate is fixed to 1.0Mbit/S. For firmware V3.60 or later, you can set the baud rate through CANopen Builder.

e. Set a station address for every servo, based on the number of servos. Set P3-00 of each servo to 1, 2, and 3 in order. You can set a maximum of eight servos.

f. Power the servo off and back on again.

g. Begin operation after the basic setting is complete.

3. Download the sample program and set M1614 to ON to enable Delta servo drive function. The instruction initializes the servos at station addresses 1–3. When M1615 is ON, the initialization is complete. When the servo enters CANopen mode successfully, CO-LD information is displayed.



4. For firmware V3.43-3.47, you can set the DI values in the initialization process, including DI1~DI4 (P2-10=23, P2-11=22, P2-12=21, P2-13=24). For firmware V3.48 or later, this function is cancelled. You can use actual values according to your needs or use default values.

5. The following list shows the settings to initialize a servo drive for firmware V3.48 or later.

a. Set P2-30 (auxiliary function) to 5 to indicate that the servo does not need to store the settings in EEPROM permanently. This can prolong the servo life span. (for firmware V3.60 or later).

b. Reset P6-02 (PATH#1) to 0 and P6-06 (PATH#3) to 0. This indicates that PATH#1 & #3 in PR mode are both cleared.

c. Set P3-06 (SDI source) to 16#0100. This indicates that DI1–DI8 are controlled by the hardware, EDI9 is controlled by the software, and EDI10–EDI14 are controlled by the hardware.

d. Reset P4-07 (SDI status controlled manually) to 0.

e. Set P2-36 (EDI9) to 16#0101. This indicates that the function of EDI9 is set to Servo ON.

f. Set P0-17 (CM1A) to 1. This indicates that the mapping parameter is the pulse command output register CMD_O.

g. Set P0-18 (CM2A) to 64. This indicates that the mapping parameter is the pulse command register CMD_E.

h. Set P5-20–P5-35 (acceleration time) to 1. This indicates that the acceleration time is 1 ms.

i. Set P5-60–P5-75 (target speed) to 1. This indicates that the target speed is 0.1 rpm.

j. Set PDO1 to correspond to P5-07 (PR command), P0-01 (Fault code), P0-46 (state of DO point) and P4-07 (state of DI point)

k. Set PDO2 to correspond to P0-09 (CM1 state: CMD_O) and P0-10 (CM2 state: CMD_E).

l. Set the slave index 0x1017 (Producer Heartbeat Time: 200 ms), the PLC (Consumer Heartbeat Time: 1000 ms) (for firmware V3.48 or earlier versions)

m: Set P3-10 to 16#0010. This indicates that when an error occurs in CAN Bus, the servo drive is OFF. (for firmware V3.60 or later)

n: Set the slave index 0x1017 (Producer Heartbeat Time: 0 ms) (for firmware V3.60 or later).

o: Set the slave index 0x100C (Guard Time: 0 ms) (for firmware V3.60 or later).

p: Set the slave index 0x100D (Life Time Factor: 0) (for firmware V3.60 or later).

q: Set the slave index 0x1016 (Consumer Heartbeat Time: 200 ms), the PLC (Producer Heartbeat Time: 66 ms) (for firmware V3.60 or later).

6. Do not use the COPRW instruction (API342) to modify the servo parameters of the six items a, b, f, g, j, and k above.

7. When you use an absolute-type servo, use the COPRW communication instruction to write 16#0100 to P3-12, which writes the relevant absolute-type servo parameters to EEPROM at the moment the servo powers off.

8. Set the relevant DI signal configuration parameters manually or with the COPRW instruction to modify the hardware DI signal setting of ASDA servo drive. Use COPRW to modify the configuration after execution of the INITC instruction is complete and before the servo is enabled.

9. When the initialization is complete, the servo is in the PR mode. Do not make any communication control on servo P5-18.

10. For more details on the servo parameters, refer to the Delta Servo Operation manual.

API	Mnemonic	Operands	Function	Controllers
329	ASDON	S_1, S_2	Driver ON and OFF	ES2-C

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ASDON: 5 steps
S_1					*	*										
S_2					*	*										

PULSE	16-bit	32-bit
--	ES2-C	--

Operands

S_1 : Station address of driver

S_2 : Driver ON and Driver OFF

Explanation

1. The INITC instruction must be complete before this instruction is executed.
2. It is not available for pulse type instructions. Do not use pulse type contact.
3. The range of **S_1** is K1–K8 (for servo). There will be no execution when the input value is out of the range.
4. **S_2** is a non-zero value, the servo is enabled (Servo-ON). If **S_2** is K0, the servo is disabled (Servo-OFF).
5. Each slave ID has an independent flag to show its state; if the flag is ON, it indicates servo drive is ON; if the flag is OFF, it indicates servo drive is OFF.

Slave	R/W	ID. 1	ID. 2	ID. 3	ID. 4	ID. 5	ID. 6	ID. 7	ID. 8
Flags for servo drives	R	M1640	M1641	M1642	M1643	M1644	M1645	M1646	M1647

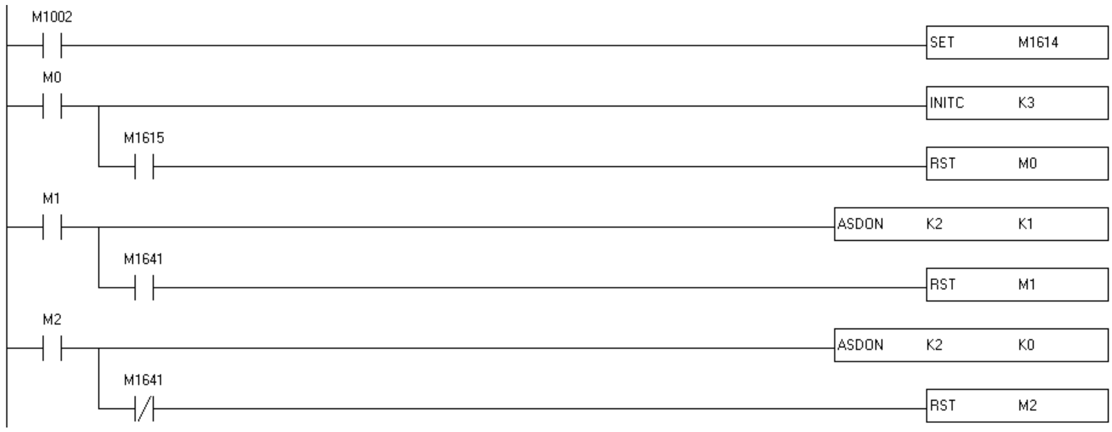
6. Any error occurs during operation, M1616 will be set to ON and the servo drive number that shows error will be stored in D6000, error codes in D6001 and STEP that when error occurs in D6002.

Example 1:

1. When M0 changes from OFF to ON, the INITC instruction starts to initialize the servos at station addresses 1–3 (should be in a consecutive order), until M1615 is ON. (The station address must start at 1 and the following addresses cannot be skipped or reserved.)

2. When M1 changes from OFF to ON, the ASDON instruction starts to enable the servo at station address 2. When SM1641 is ON, it indicates Servo-ON.

3. When M2 changes from OFF to ON, the ASDON instruction starts to disable the servo at station address 2. When SM1641 is OFF, it indicates Servo-OFF.



API	Mnemonic	Operands	Function	Controllers
330	CASD	S_1, S_2, S_3	Set acceleration time and deceleration time for driver	ES2-C

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	CASD: 7 steps
S_1					*	*										
S_2					*	*							*			
S_3					*	*							*			

PULSE	16-bit	32-bit
--	ES2-C	--

Operands

S_1 : Station address of driver

S_2 : Acceleration time (ms)

S_3 : Deceleration time (ms)

Explanation

1. The INITC instruction must be complete before this instruction is executed.
2. It is not available for pulse type instructions. Do not use pulse type contact.
3. The range of **S_1** is K1–K8 (for servo). There will be no execution when the input value is out of the range.
4. Setting range of **S_2** and **S_3** is 1-32767, any value exceeding this range is treated as 1. (unit: ms)
5. **S_2:** Acceleration time is the period of time during which the servo spins up from 0 to 3000.0 rpm. **S_3**: Deceleration time is the period of time during which the servo spins down from 3000.0 rpm to 0.
6. Once the instruction is executed, M1615 will be set to OFF. After execution is done, M1615 will be set to ON.
7. Any error occurs during operation, M1616 will be set to ON and the servo drive number that shows error will be stored in D6000, error codes in D6001 and STEP that when error occurs in D6002.

Example 1

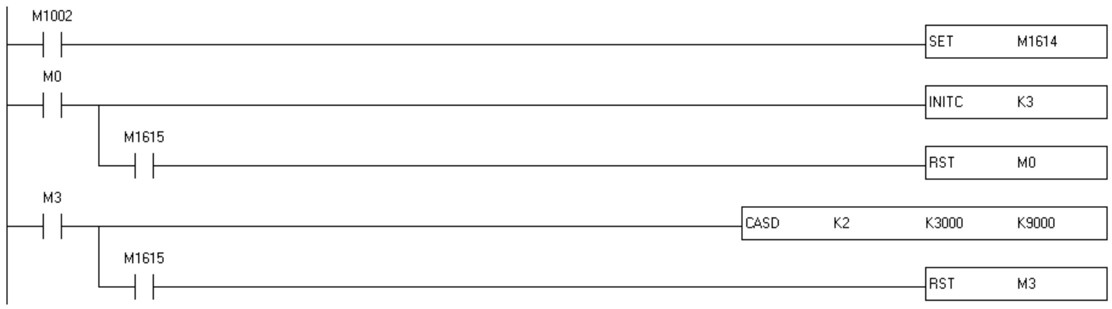
1. When M0 changes from OFF to ON, the INITC instruction starts to initialize the servos at station addresses 1–3 (should be in a consecutive order), until M1615 is ON. (The station address must start at 1 and the following addresses cannot be skipped or reserved.)

2. When M3 changes from OFF to ON and the target speed of the servo at station address 2 is 3000 rpm, the CASD instruction sets the acceleration time of servo 2 to 3000 ms and the deceleration time to 9000 ms.

3. If the target speed of servo 2 is 1000 rpm, the acceleration time and deceleration time are shown below.

   Acceleration time: [3000 ms / 3000 rpm] × 1000 rpm = 1000 ms

   Deceleration time: [9000 ms / 3000 rpm] × 1000 rpm = 3000 ms



API	Mnemonic		Operands	Function	Controllers
331	D	DRVIC	S_1, S_2, S_3	Servo relative position control	ES2-C

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DDRVIC: 13 steps
S_1					*	*										
S_2					*	*							*			
S_3					*	*							*			

PULSE	16-bit	32-bit
		ES2-C

Operands

S_1 : Station address of servo

S_2 : Relative target position

S_3 : Target speed

Explanation

1. The INITC and ASDON (servo ON) instructions must be complete before this instruction is executed.
2. It is not available for pulse type instructions. Do not use pulse type contact.
3. The range of **S_1** is K1–K8 (for servo). There will be no execution when the input value is out of the range.
4. The range of **S_2** is -2147483648 to +2147483647. The +/- sign indicates the forward / reverse direction. The target position is a relative position.
5. For firmware V3.48 or earlier versions, this function is only available for ASDA-A2. The unit of the value of **S_3** is 0.1 rpm. The range is 1–60000, which indicates 0.1–6000.0 rpm.
6. For firmware V3.60 or later, this function is only available for ASDA-A3. When using rotary motor, the unit of the value of **S_3** is 0.1 rpm. The range is 1–60000, which indicates 0.1–6000.0 rpm. When using linear motor, the unit of the value of **S_3** is 10^{-6} m/s. The range is 1-15999999, which indicates 0.000001-15.999999 m/s.
7. You need to use CASD instruction for acceleration and deceleration.
8. Once the target position is reached, the corresponding completion flags of axes M1624-M1631 will be set to ON.

9. Each ID has an independent flag to decelerate to stop (M1632-M1639).

10. Each ID has a corresponding register (D6032-D6047) to store the current position.

11. Refer to the following table for the corresponding SM and SR of the axes.

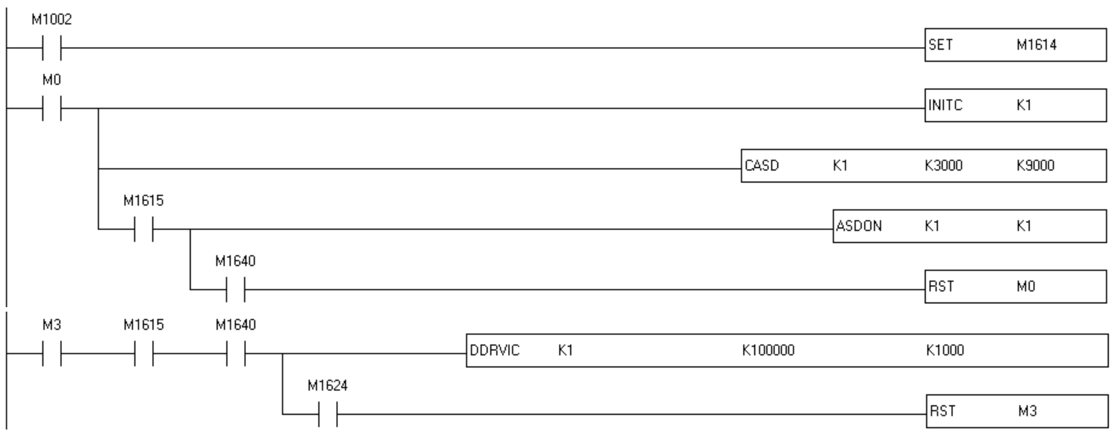
12. Any error occurs during operation, M1616 will be set to ON and the servo drive number that shows error will be stored in D6000, error codes in D6001 and STEP that when error occurs in D6002.

Example 1:

1. When M0 changes from OFF to ON, the INITC instruction starts to initialize the servos at station addresses 1–3 (should be in a consecutive order) The station address must start at 1 and the following addresses cannot be skipped or reserved. Set the acceleration time of servo 1 to 3000 ms and the deceleration time to 9000 ms, until M1615 is ON.

2. When M1615 is ON, the instruction starts enable the servo at station 1 and SM1640 is ON, indicating Servo-ON.

3. When M3 changes from OFF to ON, servo at station 1 moves to the relative position 100000 PUU at 100.0 rpm. The finish flag SM1624 is ON when the target position is reached.



Explanation of special flags (SM) and registers (SR) for ASDA CANopen communication instructions

The following table shows special flags (SM) and registers (SR) related to ASDA CANopen communication.

Flag	ID. 1	ID. 2	ID. 3	ID. 4	ID. 5	ID. 6	ID. 7	ID. 8
Enable specific function	M1614							
Initialization and communication complete flags for INITC and CASD instructions	M1615							
Communication error	M1616							
Pulse output complete	M1624	M1625	M1626	M1627	M1628	M1629	M1630	M1631
Deceleration and then stop	M1632	M1633	M1634	M1635	M1636	M1637	M1638	M1639
Servo-ON	M1640	M1641	M1642	M1643	M1644	M1645	M1646	M1647
Go-back/go-forth enabled Only DDRVAC is supported.	M1648	M1649	M1650	M1651	M1652	M1653	M1654	M1655
Go-back/go-forth direction indicator Only DDRVAC is supported.	M1656	M1657	M1658	M1659	M1660	M1661	M1662	M1663
Heartbeat error code (for firmware V3.60 and later)	M1664	M1665	M1666	M1667	M1668	M1669	M1670	M1671
Heartbeat error handling (for firmware V3.60 and later)	M1617 = OFF (default; when one goes down, all the drivers are OFF.) M1617 = ON (when one goes down, only the defective driver is OFF.)							
Number of the axis with a communication error	D6000							
Communication error code	D6001							
STEP that when error occurs	D6002							

The following table shows how Delta servo parameters of axes correspond to special flags and registers in the CANopen communication.

Servo Parameter Name (Number)	ID. 1	ID. 2	ID. 3	ID. 4	ID. 5	ID. 6	ID. 7	ID. 8
PR command (P5_07)	D6008	D6009	D6010	D6011	D6012	D6013	D6014	D6015
Alarm code (P0_01) (hexadecimal)	D6016	D6017	D6018	D6019	D6020	D6021	D6022	D6023
DO state (P0_46)	D6024	D6025	D6026	D6027	D6028	D6029	D6030	D6031
Servo current position (P0_09)	D6032 D6033	D6034 D6035	D6036 D6037	D6038 D6039	D6040 D6041	D6042 D6043	D6044 D6045	D6046 D6047
Target command position(P0-10)	D6048 D6049	D6050 D6051	D6052 D6053	D6054 D6055	D6056 D6057	D6058 D6059	D6060 D6061	D6062 D6063

The following table shows the CANopen error codes.

Error Code	Cause
0x0002	The slave does not respond to the SDO message.
0x0003	An error occurs in the message received by the slave. This error often occurs when the settings of the COPRW instruction are invalid causing the slave not to receive the complete message.
0x0004	The slave PDO message is not received.
0x0005	An error occurs while using the instruction operand.
0x0006	One of the stations is being used when the INITC instruction is executed.
0x0007	An error occurs in ID assignment
0x0008	RSTD instruction reset error

API	Mnemonic		Operands	Function	Controllers
332	D	DRVAC	**S₁**, **S₂**, **S₃**	Servo absolute position control	ES2-C

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DDRVAC: 13 steps
S_1					*	*										
S_2					*	*							*			
S_3					*	*							*			

PULSE	16-bit	32-bit
		ES2-C

Operands

S_1： Station address of servo

S_2： Absolute target position

S_3： Target speed

Explanation

1. The INITC and ASDON (servo ON) instructions must be complete before this instruction is executed.
2. It is not available for pulse type instructions. Do not use pulse type contact.
3. The range of **S_1** is K1–K8 (for servo). There will be no execution when the input value is out of the range.
4. The range of **S_2** is -2147483648 to +2147483647. The +/- sign indicates the forward / reverse direction. The target position is a relative position.
5. For firmware V3.48 or earlier versions, this function is only available for ASDA-A2. The unit of the value of **S_3** is 0.1 rpm. The range is 1–60000, which indicates 0.1–6000.0 rpm.
6. For firmware V3.60 or later, this function is only available for ASDA-A3. When using rotary motor, the unit of the value of **S_3** is 0.1 rpm. The range is 1–60000, which indicates 0.1–6000.0 rpm. When using linear motor, the unit of the value of **S_3** is 10^{-6} m/s. The range is 1-15999999, which indicates 0.000001-15.999999 m/s.
7. You need to use CASD instruction for acceleration and deceleration.
8. Once the target position is reached, the corresponding completion flags of axes M1624-M1631 will be set to ON.

9. Each ID has an independent flag to decelerate to stop (M1632-M1639).

10. Each ID has a corresponding register (D6032-D6047) to store the current position.

11. Refer to the following table for the corresponding SM and SR of the axes.

12. Any error occurs during operation, M1616 will be set to ON and the servo drive number that shows error will be stored in D6000, error codes in D6001 and STEP that when error occurs in D6002.

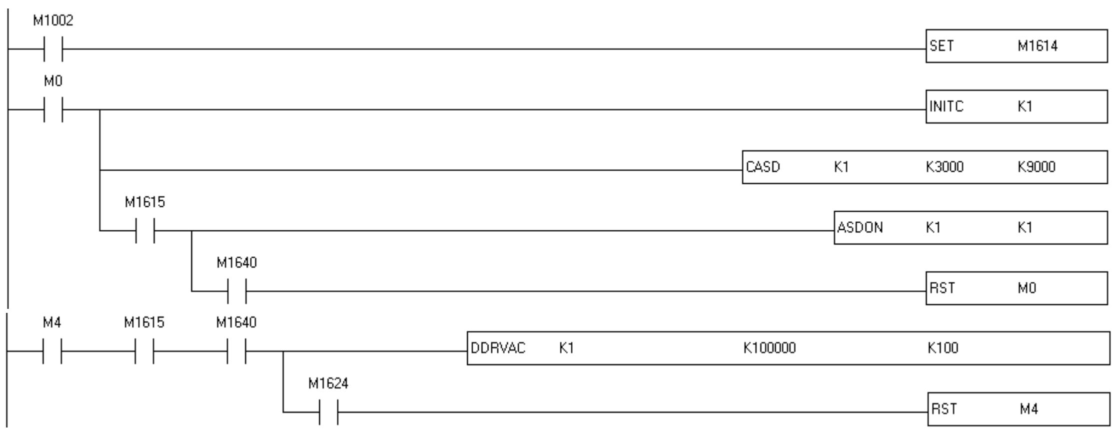
13. Go-back and go-forth functions are included. Each ID has a corresponding flag (M1648-M1655) to be used for you to enable or disable these functions and flags (M1656-M1663) to indicate the direction to go-back or go-forth.

Example 1:

1. When M0 changes from OFF to ON, the INITC instruction starts to initialize the servos at station addresses 1–3 (should be in a consecutive order) The station address must start at 1 and the following addresses cannot be skipped or reserved. Set the acceleration time of servo 1 to 3000 ms and the deceleration time to 9000 ms, until M1615 is ON.

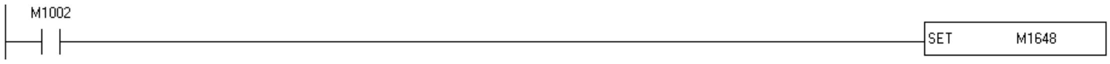
2. When M1615 is ON, the instruction starts enable the servo at station 1 and SM1640 is ON, indicating Servo-ON.

3. When M4 changes from OFF to ON, servo at station 1 moves to the relative position 100000 PUU at 100.0 rpm. The finish flag SM1624 is ON when the target position is reached.



Example 2

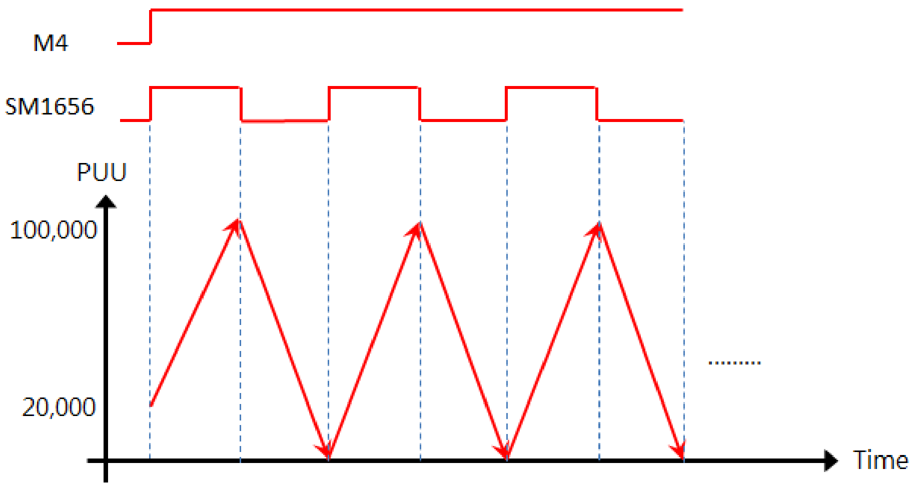
1. Add one line to the program in Example 1. When the PLC runs and SM1648 is ON, the function is enabled for servo 1 to go back and forth.



2. As the figure shows below, the servo moves from its current position (20,000) to the absolute target position (100,000) after M4 is ON. After that, it goes back and forth between the absolute position 100,000 and 0.

   The direction indication flag SM1656 is ON when the servo goes toward the target position for the first time after Servo-ON. After that, the flag repeats the state, changing from ON to OFF.

3. You can modify the target position at any time in the motion, but the new target position is only valid for the next back and forth cycle.



API	Mnemonic		Operands	Function	Controllers
333	D	PLSVC	S_1, S_2	Driver speed control	ES2-C

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	PLSVC: 5 steps DPLSVC: 9 steps
S_1					*	*										
S_2					*	*							*			

PULSE	16-bit	32-bit
	ES2-C	ES2-C

Operands

S_1 : Station address of a driver

S_2 Target speed

Explanation

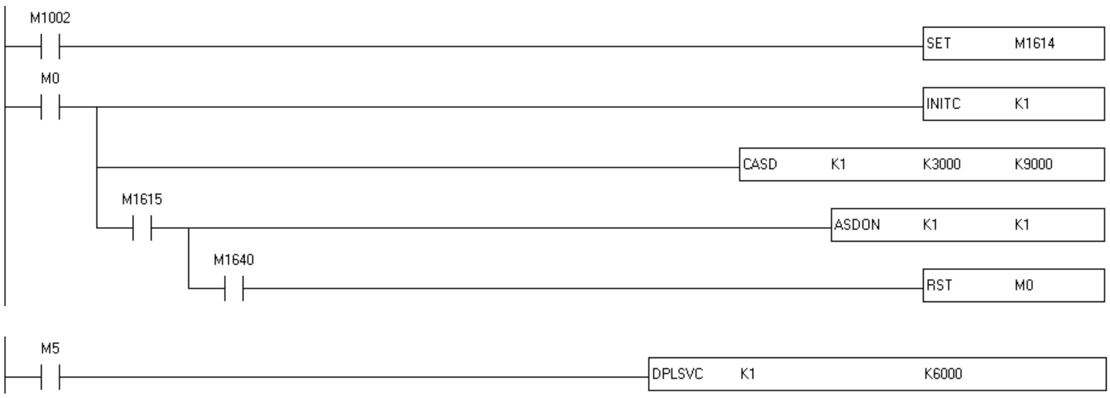
1. The INITC and ASDON (Servo ON) instructions must be complete before this instruction is executed.
2. It is not available for pulse type instructions. Do not use pulse type contact.
3. The range of **S_1** is K1–K8 (for servo). There will be no execution when the input value is out of the range.
4. The range of **S_2** is -60000 to +60000. The +/- sign indicates the forward / reverse direction. The target position is a relative position.
5. You need to use CASD instruction for acceleration and deceleration.
6. Each ID has an independent flag to decelerate to stop. (M1632-M1639).
7. Each ID has a corresponding register (D6032-D6047) to store the current position.
8. Any error occurs during operation, M1616 will be set to ON and the servo drive number that shows error will be stored in D6000, error codes in D6001 and STEP that when error occurs in D6002.
9. For corresponding SM and SR of the axes, refer to the DRVIC instruction (API331).

Example 1

1. When M0 changes from OFF to ON, the INITC instruction starts to initialize the servos at station addresses 1–3 (should be in a consecutive order) The station address must start at 1 and the following addresses cannot be skipped or reserved. Set the acceleration time of servo 1 to 3000 ms and the deceleration time to 9000 ms, until M1615 is ON.

2. When M1615 is ON, the instruction starts enable the servo at station 1 and SM1640 is ON, indicating Servo-ON.

3. When M5 changes from OFF to ON, servo 1 moves at 600.0 rpm until M5 is OFF.



API	Mnemonic			Operands	Function	Controllers
334	D	ZRNC		**S_1, S_2, S_3**	Servo homing	ES2-C

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ZRNC: 7 steps
S_1					*	*										DZRNC: 13 steps
S_2					*	*							*			
S_3					*	*							*			

PULSE	16-bit	32-bit
	ES2-C	ES2-C

Operands

S_1 : Station address of servo

S_2 : The 1st – segment speed

S_3 : The 2nd – segment speed

Explanation

1. DZRNC instruction is supported for firmware V3.60 or later.
2. The INITC and ASDON (servo ON) instructions must be complete before this instruction is executed.
3. It is not available for pulse type instructions. Do not use pulse type contact.
4. The range of **S_1** is K1–K8 (for servo). There will be no execution when the input value is out of the range.
5. For firmware V3.48 or earlier versions, this function is only available for ASDA-A2. The range for **S_2** is 1-20000. The range for **S_3** is 1-5000. The unit of the value of **S_2** and **S_3** is 0.1 rpm.
6. For firmware V3.60 or later, this function is only available for ASDA-A3. When using rotary motor, the unit of the value of **S_2** and **S_3** is 0.1 rpm. The range of **S_2** and **S_3** is 1–60000, which indicates 0.1–6000.0 rpm. When using linear motor, the unit of the value of **S_2** and **S_3** is 10^{-6} m/s. The range of **S_2** and **S_3** is 1-15999999, which indicates 0.000001-15.999999 m/s.
7. You need to use CASD instruction for acceleration and deceleration.
8. Once the target position is reached, the corresponding completion flags of axes M1624-M1631 will be set to ON.
9. Each ID has a corresponding register (D6032-D6047) to store the current position.

10. When M1 changes from OFF to ON, the setting for servo is as below.

Parameters	Description	CANopen address
P5-04 (16bit)	Homing mode	H2504
P6-00 (32bit)	Homing setting	H2600
P6-01 (32bit)	Origin definition	H2601

11. Any error occurs during operation, M1616 will be set to ON and the servo drive number that shows error will be stored in D6000, error codes in D6001 and STEP that when error occurs in D6002.

12. For corresponding SM and SR of the axes, refer to the DRVIC instruction (API331).

Example 1

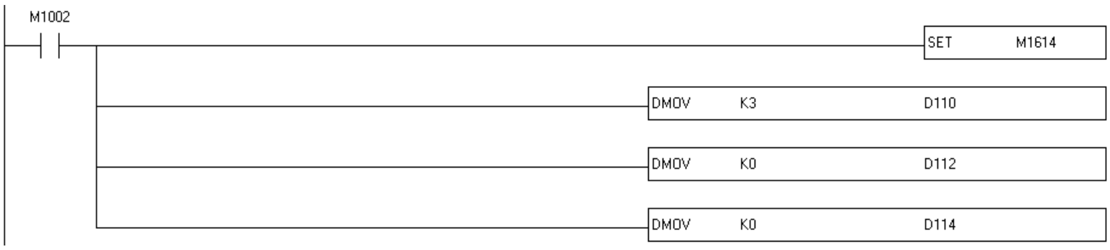
1. When M0 changes from OFF to ON, the INITC instruction starts to initialize the servos at station addresses 1–3 (should be in a consecutive order) The station address must start at 1 and the following addresses cannot be skipped or reserved. Set the acceleration time of servo 1 to 3000 ms and the deceleration time to 9000 ms, until M1615 is ON.

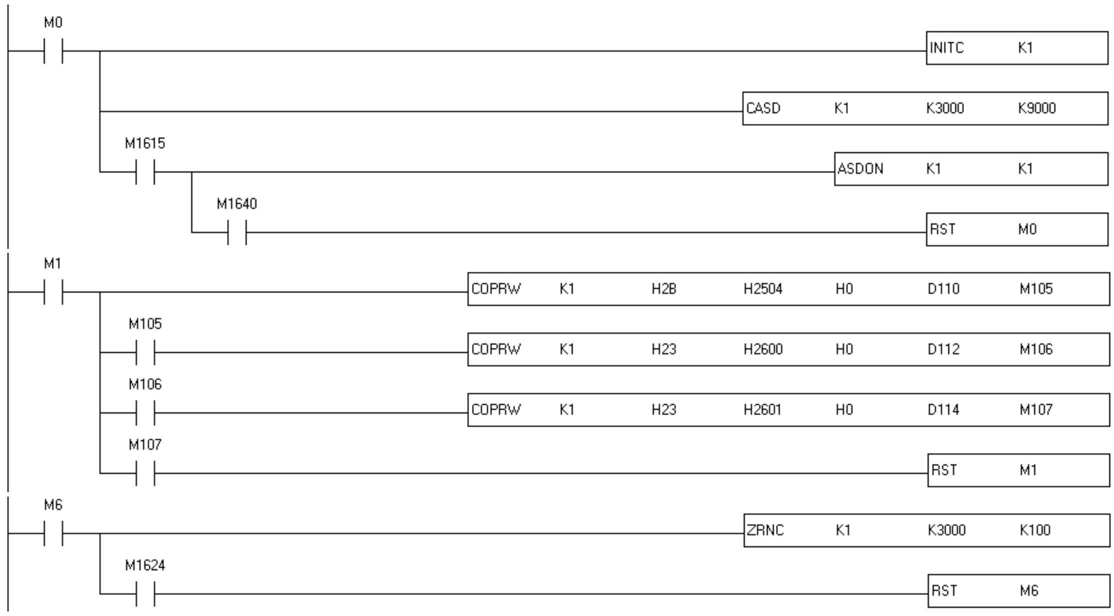
2. When M1615 is ON, the instruction starts enable the servo at station 1 and SM1640 is ON, indicating Servo-ON.

3. When M1 changes from OFF to ON, the setting for servo is as below.

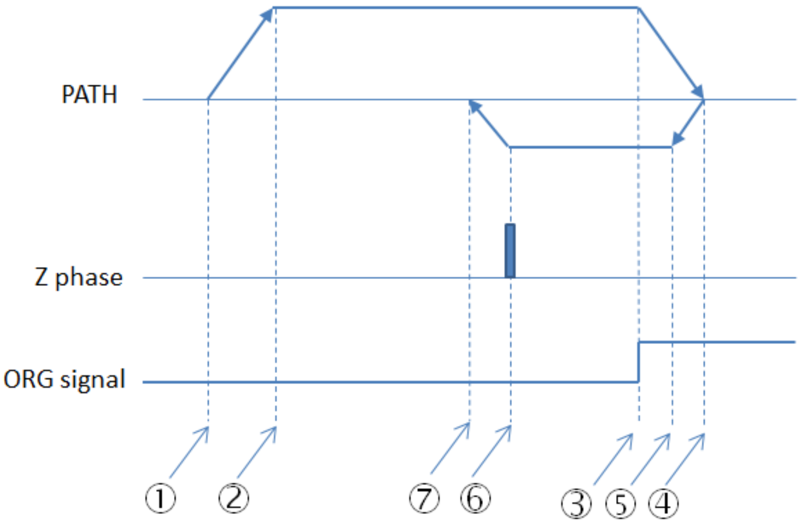
Parameters	Description	CANopen address	Setting value	Completion flag
P5-04 (16bit)	Homing mode	H2504	D110=K3	M105
P6-00 (32bit)	Homing setting	H2600	D112=K0	M106
P6-01 (32bit)	Origin definition	H2601	D114=K0	M107

4. When M6 changes from OFF to ON, the homing function is enabled for servo 1. After homing is complete, M1624 is ON.





For firmware V3.48 or eariler versions: after finding the origin (Sensor or Z), the motor has to decelerate to stop. The stop position will slightly exceed the origin.



M1 = Off→On, starts homing and moves towards the reverse direction

② Reaching the first high speed

③ Finding the origin (Sensor or Z)

④ After decelerating to stop, it moves towards the forward direction

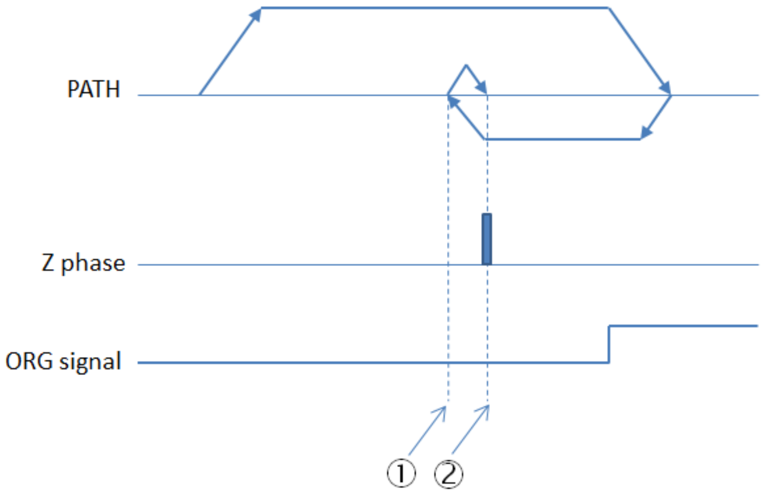
⑤ Reaching the second low speed

⑥ After leaving origin and then meeting the first Z phase, it starts to decelerate

⑦ After decelerating, it stops

For firmware V3.60 or later, you can use ZRNM instruction to set whether executing homing to the exact origin point or not (default is not coming back to the exact origin point).

To the exact origin point:



① After homing, the servo moves according to the established path 1 automatically.

② It stops at the exact origin point.

API		Mnemonic	Operands	Function	Controllers
335	D	COPWL	**S_1, S_2, S_3, D**	Writing multiple CANopen parameter values	ES2-C

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
S_1					*	*										COPWL: 9 steps
S_2													*			DCOPWL: 17 steps
S_3					*	*							*			
D			*													

PULSE	16-bit	32-bit
		ES2-C

Operands

S_1 : Station address(Mac ID)

S_2 : Starting source device where written data are stored

S_3 : Number of messages to consecutively write data

D : Communication completion flag

Explanation

1. COPWL instruction is supported for firmware V3.60 or later. And it can work with CANopen DS301 mode and Delta special mode

2. It is not available for pulse type instructions. Do not use pulse type contact.

3. **S_1** sets the station address within the range of 1~127. If the setting value exceeds the range (< 1 or >127), the instruction will automatically send data at the minimum or maximum value respectively.

4. **S_2** is the starting source device where written data are stored and **S_3** is the number of messages to consecutively write data. E.g., **S_2** specifies D10 as the starting device and the number of messages to consecutively write data is 3. Here is the detailed explanation in the following table.

Instruction name	Message No.	Index address	Subindex address	Written source data
COPWL (Writes 16-bit values)	1	D10	D11	D12
	2	D13	D14	D15
	3	D16	D17	D18
DCOPWL (Writes 32-bit values)	1	D10	D11	D12, D13
	2	D14	D15	D16, D17
	3	D18	D19	D20, D21

The value of **S_3** is in the range of 1~100.

5. For the index addresses and subindex addresses of Delta servo and AC motor drive, refer to the explanation of the COPRW instruction. In principle, the parameter values of Delta servo and AC motor drive are both16-bit or 32-bit values including floating point numbers. If you need write an 8-bit value, use the COPRW instruction.

6. **D** is the communication completion flag. **D** will turn on after the sending of multiple communication messages is complete.

   See the detailed sending process and sequence diagram below.

   ① The COPWL instruction is enabled and starts to send data.

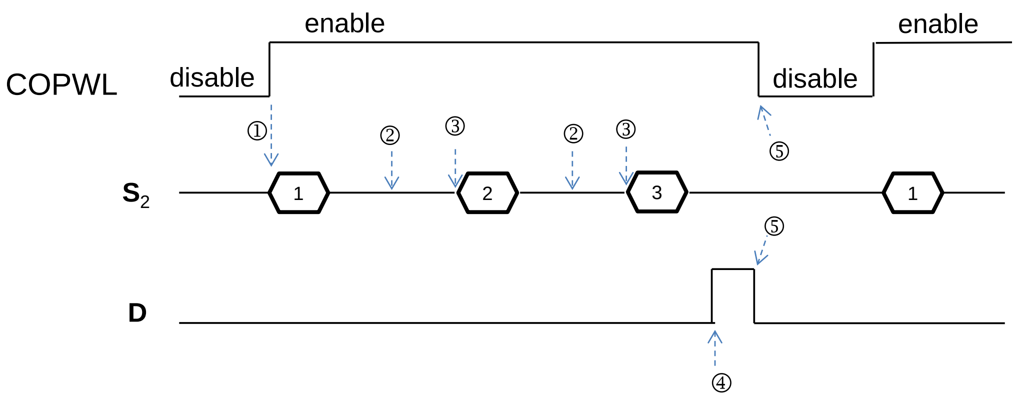
   ② After the COPWL instruction sends one piece of message, the next PLC instruction continues to execute.

   ③ As the COPWL instruction is scanned once again and the prior message has been received by the slave, the COPWL instruction sends the next message.

   ④ When the last written-data sending is done, the instruction will set the completion flag to ON.

   ⑤ When the completion flag turns on, the COPWL instruction need be disabled by manual so that the subsequent COPWL or COPRW instruction can continue to work.

   Note: When you disable the instruction, the completion flag will be automatically cleared accordingly.



   Note: The sequence diagram above shows the sending of 3 pieces of written data.

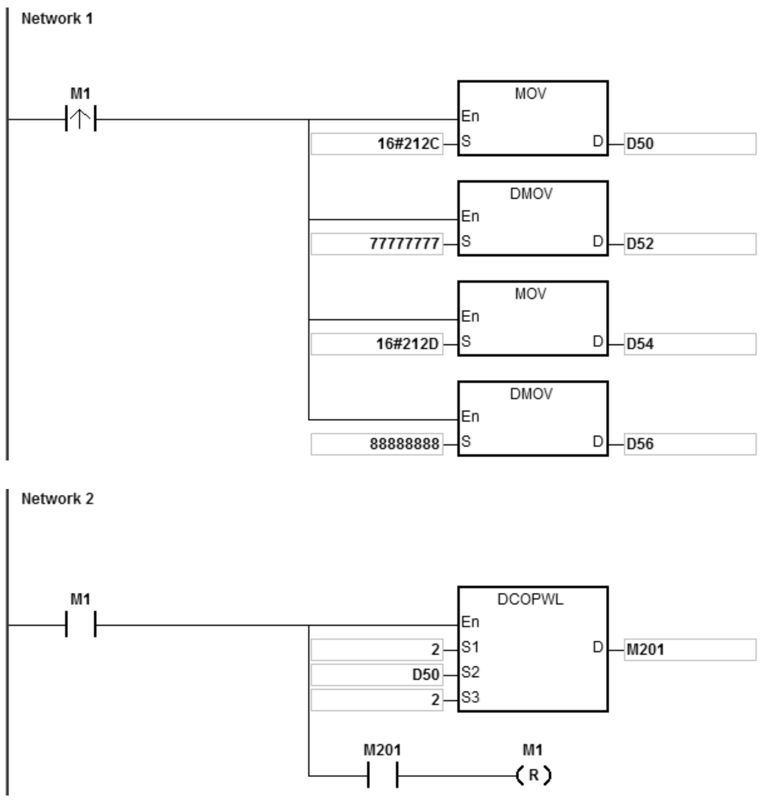
7. After the instruction is enabled, wait until the writing is complete and then disable the instruction. If there is a communication error in the execution, shoot the trouble and then re-enable the instruction to write all data.

Example

1. When M1 = OFF → ON, data are written in D device.

Instruction name	Data No.	Index address	Subindex address	Written data source
DCOPWL (Writes 32-bit values)	1	D50 = 16#212C (E-gear ratio numerator)	D51 = 0	D52, D53 = 77777777
	2	D54 = 16#212D (E-gear ratio denominator)	D55 = 0	D56, D57 = 88888888

2. When M1 = OFF → ON, the instruction writes a 32-bit value for P1-44 of the servo whose station address is 2 and the written value 77777777 is stored in D52. The instruction writes a 32-bit value for P1-45 and the written value 88888888 is stored in D56. As the writing is complete, M201 turns ON.



API	Mnemonic	Operands	Function	Controllers
336	RSTD	Node, Para, Ok, Err	Sending Reset or NMT command	ES2-C

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
Node					*	*							*			RSTD: 9 steps
Para													*			
Ok			*													
Err			*													

PULSE	16-bit	32-bit
	ES2-C	

Operands

Node : Station address which is reset

Para : Setting value of the parameter

Ok : The reset completion flag

Err : The reset error flag

Explanation

1. RSTD instruction is supported for firmware V3.60 or later. And it can work with CANopen DS301 mode and Delta special mode.
2. Before the RSTD instruction is used in Delta special instruction mode, make sure that all Delta drives have been initialized via the INITC instruction and they once worked normally.
3. When used in CANopen DS301 mode, the RSTD instruction works as the NMT communication function and can switch network states via the **Para** parameter.
4. When CAN communication port is specified to work in Delta special driver mode, the value of **Node** can be 0 (for the broadcast function) and 1~8 which are for servo station addresses only. When the station address exceeds the range, the PLC will not perform the reset action and the **Err** flag turns on. (Refer to the explanation of D6001 for error codes)
5. When CAN communication port is specified to work in CANopen DS301 mode, the value of **Node** is in the range of 0~16 and 0 (for the broadcast function). When the value exceeds the range, the PLC will not perform the NMT communication and the **Err** flag turns on. (Refer to the explanation of D6001 for error codes)
6. The setting value of **Para** is only applicable to CANopen DS301 mode. The settings for **Para** (NMT service code) are listed in the following table. If the setting value is not one of the values in the table, the **Err** flag turns on.

NMT service code	16#01	16#02	16#80	16#81	16#82
Function description	Start the slave	Stop the slave	Enter the pre-operational state	Reset the application layer	Reset the communication

7. The RSTD instruction can implement the command action on only one drive or slave every time. If multiple RSTD instructions are enabled simultaneously, the PLC will automatically take priority to perform the instruction which is enabled earlier.
8. The RSTD instruction is executed to send the command when it is enabled. If the instruction is disabled before the **Ok** flag is on, the PLC will not set the **Ok** flag to ON.
9. Apart from notifying the specified drive to clear the error state, the instruction would also re-check if relevant communication parameter values are correct and re-set correct communication parameter values.

   For example, due to the disconnection of the slave of station address 2, the entire system stops running. After the trouble is solved, the slave of station address 2 can return to the state of being controllable by using the RSTD instruction to reset the slave of station address 2 only. So the time of re-initializing all drives are saved.
10. If the slave responds by sending back any communication command fault to the PLC during the communication, the RSTD instruction will turn the **Err** flag on and stop the upcoming actions. (Refer to explanation of D6001 for error codes.)

API	Mnemonic	Operands	Function	Controllers			
				ES2/ EX2	SS2	SE/ ES2-E	SA2/ SX2
337	ETHRS	**S_1, S_2, S_3, S_4, D_1, D_2**	Self-defined Ethernet communication port				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ETHRS: 13 steps
S_1													*			
S_2													*			
S_3													*			
S_4					*	*							*			
D_1													*			
D_2			*													

PULSE				16-bit				32-bit			
ES2/ EX2	SS2	SA2/ SE	SX2	ES2/ EX2	SS2	SE/ ES2-E	SA2/ SX2	ES2/ EX2	SS2	SA2/ SE	SX2

Operands:

S_1: Local communication port, target IP address, communication port and UDP/TCP mode
S_2: Communication mode **S_3**: Data source **S_4**: Data length **D_1**: Receive data address
D_2: Receiving completion flag

Explanations:

1. This instruction is currently available for DVP-SE series PLC with firmware V1.83 or later.
2. **S_1** is for setups of local communication port, target IP address, communication port and UDP/TCP mode. This operand occupies 5 consecutive devices.

   IP address settings: this occupies 2 consecutive devices, **S_1**+1 and **S_1**+2 respectively

   IP definition → IP3.IP2.IP1.IP0 → 192.168.0.2

   If **S_1** is D100, the input value should be:

D100 (**S_1**+0)	D101 (**S_1**+1)		D102 (**S_1**+2)		D103 (**S_1**+3)	D104 (**S_1**+4)
Local port	High (IP1)	Low (IP0)	High (IP3)	Low (IP2)	Target port	UDP/TCP
0~65535	0	2	192	168	0~65535	0, 1
	H'0002		H'C0A8			0=UDP, 1=TCP

3. **S_2** is where you can set up modes. Client mode 0 and 1 are exchangeable and the connections are active. Server mode 2, 3 and 4 are exchangeable and the connections are active. But it is required to disconnect the connection when switching between different modes.

Value in S2	S_2 receiving mode	Description of S2+1	Remark
0	After the sending is complete, no receiving is allowed and a completion flag will be set to ON.	Unused	Client Mode 0 cannot be set in the sending data length **S_4**.
1	Complete the sending first, and then start receiving. After receiving is complete, a completion flag will be set to ON.	Receiving timeout setting; unit: 1ms; setting range: 100~32000	Client Mode A. 0 cannot be set in the sending data length **S_4**. (before firmware V1.90 for DVP12SE) B. 0 can be set in the sending data length **S_4** and that indicates not sending but start to receive data. (available for ES2-E with firmware V1.2, or later, 12SE with firmware V1.92 or later, and 26SE with firmware V1.00 or later)
2	Complete the receiving first, after the receiving is done, send the packets. After the sending is complete, a completion flag will be set to ON.	Receiving timeout setting; unit: 1ms; setting range: 100~32000; when the setting value is 0, it means no timeout.	Server Mode 0 cannot be set in the sending data length **S_4**.
3	When the receiving time is less than setting value in **S_2**+1, after receiving the communication packet, the receiving is complete.	Receiving timeout setting; unit: 1ms; setting range: 100~32000; when the setting value is 0, it means no timeout.	Server Mode **S_4** is invalid in this mode.

4	Sending specific communication data (SE: FW V1.95 or later and ES2E: FW V1.46 or later)	Invalid parameter	Server mode The value in **S_4** cannot be 0 in this mode. The value exceeding the range will be seen as the maximum or the minimum value.

Mode 2: Contents for the packets to be sent should be ready. Once the receiving is done, the sending is executed immediately.

Mode 3+4: Contents for the packets to be sent can be ready before the next scan cycle. The sending is executed in the next scan cycle.

Target port descriptions: S_2 and **S_1**+0, **S_1**+1, **S_1**+2, **S_1**+3

Start Mode	Remote IP	Local communication port	Remote communication port	Description
0, 1	Specific IP address	0	0	Illegal
0, 1	Specific IP address	0	Not equal to 0	Master mode, Specifies the IP address; but not specify the local communication port.
0, 1	Specific IP address	Not equal to 0	0	Illegal
0, 1	Specific IP address	Not equal to 0	Not equal to 0	Master mode, Specifies the IP address, local communication port and remote communication port
0, 1	0.0.0.0	No limit to the value	No limit to the value	Illegal
2, 3, 4	Specific IP address	0	No limit to the value	Illegal
2, 3, 4	Specific IP address	Not equal to 0	0	Slave mode, Not specify the IP address and remote communication port
2, 3, 4	Specific IP address	Not equal to 0	Not equal to 0	Slave mode, Specify the IP address and remote communication port
2, 3, 4	0.0.0.0	0	No limit to the value	Illegal
2, 3, 4	0.0.0.0	Not equal to 0	0	Slave mode, Not specify the IP address and remote communication port
2, 3, 4	0.0.0.0	Not equal to 0	Not equal to 0	Slave mode, Not specify the IP address and remote communication port
2, 3, 4	Specific IP address	0	No limit to the value	Illegal

4. The operand **S**$_3$ and **S**$_4$ specify source data registers and data length. For example : **S**$_3$ specifies D150 and the value in **S**$_4$ is 10. The instruction ETHRS will send 10 bytes of data, starting from the low byte in D150, D151, D152 and so on. Users can use the instruction DTM to transform 16-bit data into 8-bit data when the transformation is required. The setting range for **S**$_4$ is 1~200 words. If the setting values exceed the setting range, the system will use the minimum (1) or the maximum (200) to operate.
5. The operand **D**$_1$ specifies a destination data register. For example, **D** specifies D10 and D10 is the received data length; the unit is byte. The data received will be stored starting from D11, low byte in D11, D12, D13 and so on. The maximum receiving data length is 200 words; data length exceeds this limit will not be stored in **D**. Users can use the instruction DTM to transform 16-bit data into 8-bit data when the transformation is required.
6. **D**$_2$ specifies the reception completion flag and only M device can be used. When the instruction is executed, and the transmission of packets is complete, this flag will be set to ON. Users can learn from this flag to see if the transmission is complete or not. Once it is set to ON, users need to set it to OFF. When there is any error occurred during the instruction execution or any timeout occurred, the flag will not be ON.
7. Once the instruction is executed, the communication begins. There is no need to use any special flag to trigger the sending. When the instruction is executed, there will be a special M shown to indicate the execution.
8. There is no limitation on the times of using this instruction in the program. However, only one instruction can be executed at a time.
9. When the instruction is forcedly stopped, the communication will also be stopped. And the completion flag **D**$_2$ will not be ON.
10. When this instruction is executed, do not use the Online Mode; otherwise errors may occur when receiving and storing data.
11. This instruction is available for the following models and firmware versions.

Series	ES2-E	12SE	26SE
Firmware version	V1.08	V1.88	V1.0

(For mode 4 in **S**$_2$, it is only available for SE PLC CPU with firmware V1.95 or later and ES2-E with firmware V1.6 or later.)

12. Relative special flags and registers for the instruction ETHRS:

Item	Function	Defaults	Stop⇔Run	Attributes
M1196	ON: the connection of the self-defined Ethernet communication port is enabled. When the instruction ETHRS stops, the connection will still be kept. ON=> OFF: the connection will be disabled. Off: use the instruction ETHRS to control the connection, when the instruction is executed, the connection is enabled.	Off	Off	R/W
M1197	ON: the instruction ETHRS is being executed.	Off	Off	R
M1198	ON: when there is a communication error or a communication timeout, the control on the connection of the self-defined Ethernet communication port is through M1196. When the communication timeout occurs, the communication instruction has to be stopped and then start the instruction again to start the communication.	Off	Off	R/W
D1176	Error code	0	0	R
D1227 D1228	During execution of the ETHRS instruction, if it is in the receiving mode, D1227 and D1228 show the sender's IP address. (available for ES2-E V1.2, 12SE V1.92, 26SE V2.00) (available for ES2-E with firmware V1.2, or later, 12SE with firmware V1.92 or later, and 26SE with firmware V1.00 or later)	0	-	R

13. If M1198 is ON, it means communication errors occur and an error code will be stored in D1176. For other error codes, please refer to the following table.

When **S1**+4=0 (UDP mode)

Error code	Description
H2003	The value exceeds the range.
H600C	The local socket has been used.
H600D	Ethernet network is not connected.
H6209	UDP Socket illegal IP address
H620A	UDP Socket illegal communication mode
H620C	UDP Socket illegal address for sending data

Error code	**Description**
H620D	UDP Socket the length of sent data exceeds the range
H620E	UDP Socket the device where data are sent exceeds the range
H620F	UDP Socket illegal address for receiving data
H6210	UDP Socket the length of data actually received exceeds the range.
H6211	UDP Socket the device where data are received exceeds the range.
H6213	UDP Socket the size of data actually received is larger than the set data.
H6215	UDP Socket is not connected
H6217	UDP Socket connection has been triggered

When $\mathbf{S_1}$+4=1 (TCP mode)

Error code	**Description**
H2003	The value exceeds the range.
H600C	The local socket has been used.
H600D	Ethernet network is not connected.
H6200	TCP Socket illegal IP address
H6201	Illegal TCP Socket communication mode setting
H6202	Illegal TCP Socket mode setting
H6203	TCP Socket illegal address for sending data
H6204	TCP Socket the length of sent data exceeds the range
H6205	TCP Socket the device where data are sent exceeds the range
H6206	TCP Socket illegal address for receiving data
H6207	TCP Socket the length of received data exceeds the range
H6208	TCP Socket the device for receiving data exceeds the range
H6212	TCP Socket communication timeout
H6213	TCP Socket the size of data actually received is larger than the set data.
H6214	TCP Socket connection is rejected by the remote equipment
H6215	TCP Socket has not been connected
H6217	TCP Socket connection has been triggered.

14. The already used communication ports are as below.

UDP/TCP	Communication Port	Description
TCP	502	Modbus TCP communication
TCP	44818	EtherNet/IP explicit message
UDP	67	DHCP communication
UDP	68	
UDP	2222	EtherNet/IP implicit message
UDP	44818	EtherNet/IP explicit message
UDP	20006	For internal parameter download
UDP	20008	

15. Descriptions for relevant flags during communication:

TCP MODE:

M1196=ON: Communication port is connected

- Master/Slave mode; communication is working fine.
  - M1197 = ON, this indicates the communication is active. Make sure the TCP connection is ready or is waiting to be connected and checking its relative communication settings are set, the data length is less than 200 characters and if the slave is responding.
  - After data is sent or received, M1197 stays ON and a completion flag will be set to ON. (You can reset this flag to OFF.)
  - If the ETHRS instruction is executed again, the completion flag will be reset to OFF.

- Master/Slave mode; an error occurs during communication.
  - M1197 = ON, this indicates the communication is active. Make sure the TCP connection is ready or is waiting to be connected.
  - When an error occurs, M1198 is ON and the error codes will be shown in D1176.
  - Execute ETHRS instruction again, after the problem is fixed, and M1198 is reset to OFF.
  - If receiving time out is enabled in Master mode, it starts counting after the sending is done.
  - If receiving time out is enabled in Slave mode, it starts counting after the connection is established.

M1196=OFF: Use ETHRS instruction to control the connection; when it is executed, the connection is established.

- Master/Slave mode; communication is working fine.
  - M1197 = ON, this indicates the communication is active. Make sure the TCP connection is ready or is waiting to be connected and checking its relative communication settings are set, the data length is less than 200 characters and if the slave is responding.
  - After data is sent or received, M1197 stays ON and a completion flag will be set to ON. (You can reset this flag to OFF.)
  - When the connection time is exceeding the setting value in Keep Alive Timeout (default: 30ms), the connection will be switched off. M1197 is set to OFF.
  - If the ETHRS instruction is executed again, the completion flag will be reset to OFF.

- Master/Slave mode; an error occurs during communication.
  - M1197 = ON, this indicates the communication is active. Make sure the TCP

connection is ready or is waiting to be connected.

- When an error occurs, M1198 is ON and the error codes will be shown in D1176.
- Execute ETHRS instruction again, after the problem is fixed, and M1198 is reset to OFF.
- If receiving time out is enabled in Master mode, it starts counting after the sending is done.
- If receiving time out is enabled in Slave mode, it starts counting after the connection is established.

UDP MODE:

➢ Master/Slave mode; communication is working fine.
(Note: if M1196 is switched from ON to OFF during communication, the connection will be switched off. M1197 is reset to OFF and the completion flag will be set to ON.
- M1197 = ON, this indicates the communication is active. Make sure the TCP connection is ready or is waiting to be connected and checking its relative communication settings are set, and the data length is less than 200 characters.
- After data is sent or received, M1197 stays ON and a completion flag will be set to ON. (You can reset this flag to OFF.)
- If the ETHRS instruction is executed again, the completion flag will be reset to OFF.

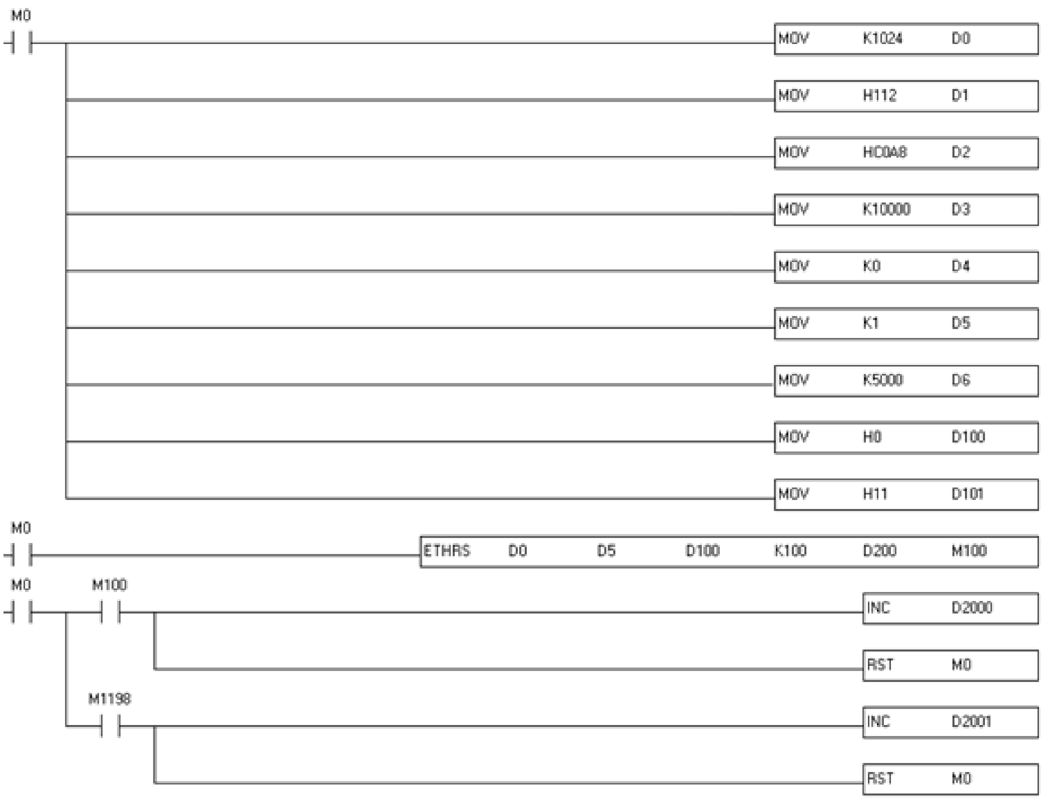
➢ Master/Slave mode; an error occurs during communication.
- M1197 = ON, this indicates the communication is active. Make sure the UDP connection is ready or is waiting to be connected.
- When an error occurs, M1198 is ON and the error codes will be shown in D1176.
- Execute ETHRS instruction again, after the problem is fixed, and M1198 is reset to OFF.
- If receiving time out is enabled in Master mode, it starts counting after the sending is done.
- If receiving time out is enabled in Slave mode, it starts counting after the connection is established.

Program Example: (The command is sent and received through the Ethernet port built in DVP-SE.)

This example uses DVP-SE series as the client, M0 to activate and UDP connection mode to send and receive data. The value in **S**$_2$ is K1. When the data is received, M100 is set to ON. The relative parameters are stated below.

TCP Socket Connection	
Remote IP	192.168.1.18
Remote port	10000
Local port	1024
Send Data Address	D100
Send Data Length	100
Receive Data Address	D200
Communication timeout (ms)	5000

1. When M0 is ON, the transmission starts and M1197 is ON. If M1198 is ON, it means communication errors occur and an error code will be stored in D1176.
2. When the data is received correctly and a response is received from the remote device, M100 will be ON. The data length and the contents will be stored in D200.

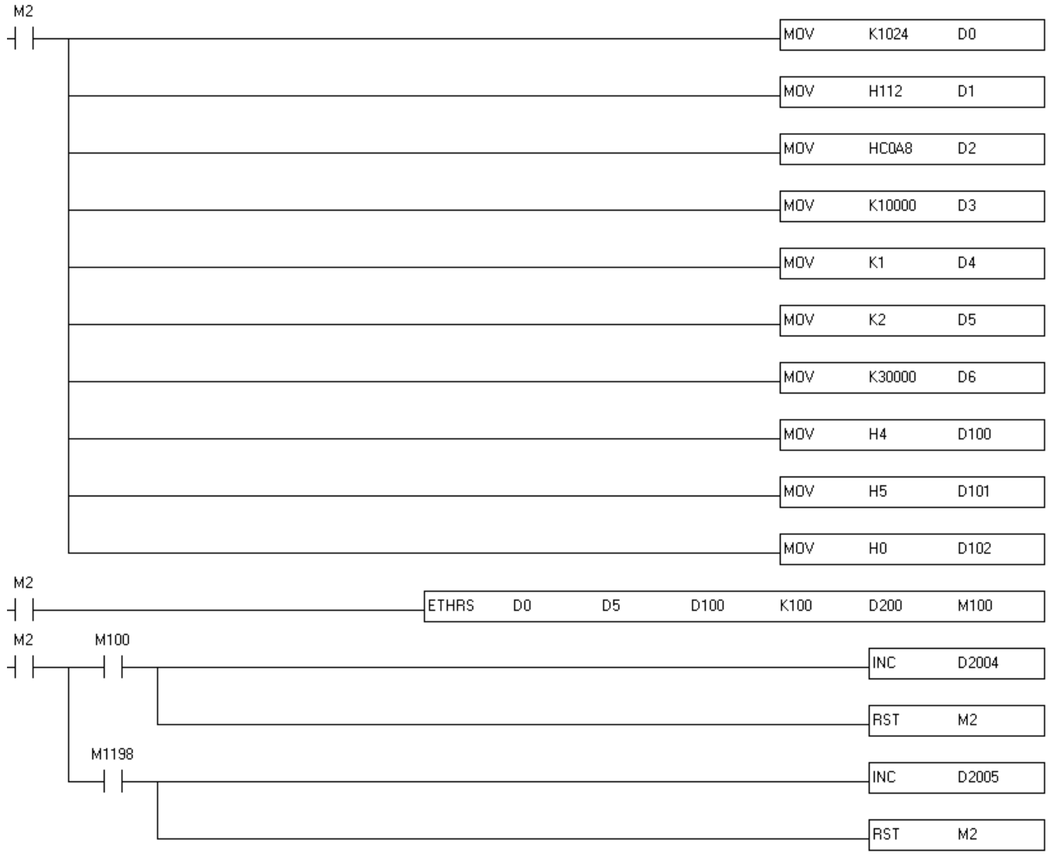


Program Example 2: (The command is sent and received through the Ethernet port built in DVP-SE.)

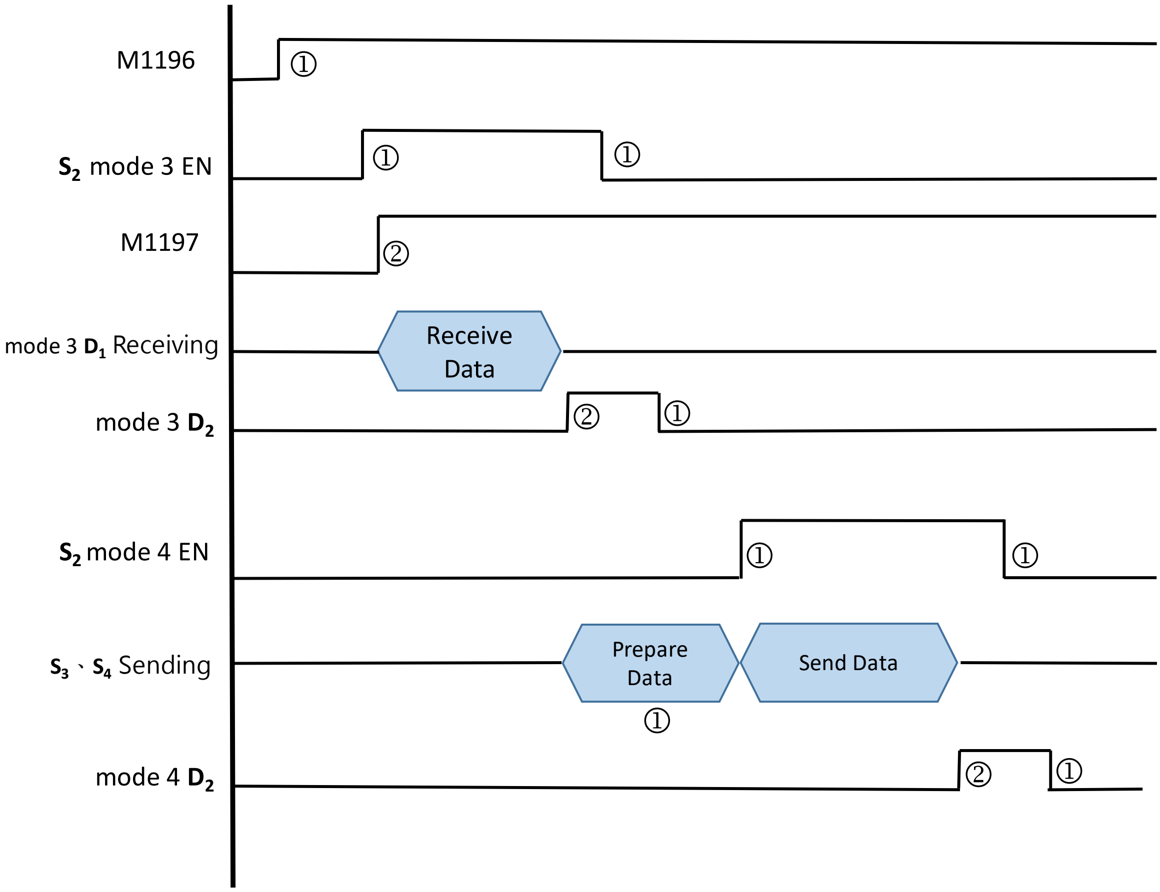
This example uses DVP-SE series as the client, M2 to activate and TCP connection mode to send and receive data. The value in **S_2** is K2. The relative parameters are stated below.

TCP Socket Connection	
Remote IP	192.168.1.31
Remote port	10000
Local port	1024
Send Data Address	D100
Send Data Length	100
Receive Data Address	D200
Communication timeout (ms)	30000

1. Set M1196 to ON. When using the TCP connection mode, it is suggested to set M1196 to ON to avoid disconnecting if a communication timeout occurs.
2. When M2 is ON, DVP-SE is waiting for the TCP connection to be established. When M100 is ON, it means the receiving is complete successfully and the data length and contents are stored in D200 and data in D100 has been sent, the data length is 100 bytes.
3. If M1198 is ON, it means communication errors occur and an error code will be stored in D1176.



Program Example 2: (The command is sent and received through the Ethernet port built in DVP-SE and mode 3 and 4 are used in S2.)



Note: ① users set and clear. ② PLC sets and clears.

Execution steps:

1. Set M1196 to ON. When using the TCP connection mode, it is suggested to set M1196 to ON to avoid disconnecting if a communication timeout occurs.
2. Activate mode 3 to receive data and wait for the completion flag to be ON, indicating the receiving is complete.
3. After receiving is complete, you need to clear the completion flag. Arrange the received data and store the data to be sent in the specified registers.
4. Activate mode 4 to send data and wait for the completion flag to be ON, indicating the sending is complete.

API	Mnemonic	Operands	Function	Controllers
338	EMER	**Node, Dest, Len, Ok, Err**	Reading Emergency message	ES2-C

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				**Word devices**											
Node					*	*							*			EMER: 11 steps
Dest													*			
Len													*			
Ok			*													
Err			*													

PULSE	16-bit	32-bit
	ES2-C	

Operands

Node : Specified node ID

Dest : Target device for storing data

Len : Total number of 4 words of data which have been read

Ok : Completion flag

Err : Error flag

Explanation

1. EMER instruction is supported for firmware V3.60 or later. And it can work with CANopen DS301 mode and Delta special mode.

2. After receiving the Emergency message from the slave **Node**, the PLC will automatically store the data in the device that is specified by **Dest** and set the **Ok** flag to ON.

3. It is recommended that the **Node** value should be specified from the slave node IDs which have already existed. If the value is not one existing node ID or the slave has been disconnected, the PLC will not be able to receive any message, set the **Err** flag to ON and show error code of communication timeout. (Refer to explanation of D6001 for error codes.)

4. The way the EMER instruction reads Emergency messages is the same as Emergency communication method in ES2 operation manual. Select one communication method from them when reading Emergency messages. Two methods cannot be used at the same time.

5. The EMER instruction can read 5 Emergency messages at most. Every time the reading is successful, the **Ok** flag turns on and **Len** displays the total number of messages which are read. You can evaluate how many consecutive words are occupied by **Dest** based on the length. Every message uses 4 words. The data are stored in the order from lower 8 bits to

higher 8 bits. The storage format is shown as below. (E.g. **Dest** is D10, **Len** is 2 which is the number of messages stored in D5.)

D device no.	Value
D5	2

D device no.	Higher 8 bits	Lower 8 bits
D10	The second byte in the first message	The first byte in the first message
D11	The forth byte in the first message	The third byte in the first message
D12	The sixth byte in the first message	The fifth byte in the first message
D13	The eighth byte in the first message	The seventh byte in the first message
D14	The second byte in the second message	The first byte in the second message
D15	The forth byte in the second message	The third byte in the second message
D16	The sixth byte in the second message	The fifth byte in the second message
D17	The eighth byte in the second message	The seventh byte in the second message

API	Mnemonic	Operands	Function	Controllers
339	ZRNM	**Node, Mode, Ok, Err**	Setting the homing mode for Delta servo drive	ES2-C

Type / OP	Bit Devices				Word devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		ZRNM: 9 steps
Node					*	*							*				
Mode					*	*							*				
Ok			*														
Err			*														

PULSE	16-bit	32-bit
	ES2-C	

Operands

Node : Specified node ID

Mode : Homing mode code

Ok : Completion flag

Err : Error flag

Explanation

1. ZRNM instruction is supported for firmware V3.60 or later. And it can work with Delta special mode.

2. The INITC instruction must be complete before this instruction is executed.

3. The value of **Node** is in the range of 1~8 (exclusive to servo node IDs). If the setting value exceeds the range, the PLC will not perform the action of the homing mode and set the **Err** flag to ON. (Refer to explanation of D6001 for error codes.)

4. The ZRNM instruction can set the homing mode of only one drive every time. If multiple instructions are enabled simultaneously, the PLC will take priority to perform the instruction which is enabled earlier.

5. The ZRNM instruction is executed to send the command when it is enabled. If the instruction is disabled before the **Ok** flag is on, the PLC will not set the **Ok** flag to ON.

6. **Mode** sets a homing mode. If the setting value exceeds the range, the PLC will still send the command and the server itself will decide whether to receive the command or not. The setting mode is the homing mode that ASDA servo parameter P5-04 corresponds to.

The setting value of Delta servo homing mode is a hex value. The value is defined as the format of 0xWZYX. See the explanation of respective codes as below.

Homing mode code	Range	Function and code description	Remark
W	0 ~ 1	Select the final position where the servo stops. 0 = The servo leaves the original point, decelerates and stops and then automatically returns to the real original point. 1 = After leaving the original point, decelerating and stopping, the servo will not perform any action any more.	
Z	0 ~ 1	Handling mechanism when the limit is encountered. 0 = Output stops. 1 = Output is conducted in the reverse direction.	
Y	0 ~ 2	Z pulse signal setting (used for X code 0~8) 0 =Look for Z pulse when coming back. Do not look for Z phase when going forward. 1 =Go forward to Z pulse. Do not look for Z pulse when coming back. 2 =Do not look for Z pulse. Z pulse signal handling method (applicable to X code: 9~A) 0 =Look for Z pulse when coming back. 1 =Do not look for Z pulse both when coming back and going forward.	
X	0 ~ A	Homing method: 0~8 0 = Homing in the forward direction; **PL** is the original point	

		1 = Homing in the reverse direction; **NL** is the original point. 2 = Homing in the forward direction; **ORGP:** OFF > ON, as the original point. 3 = Homing in the reverse direction; **ORGP:** OFF > ON, as the original point. 4 = Homing in the forward direction; look for Z pulse and regard it as the original point. 5 = Homing in the reverse direction; look for Z pulse and regard it as the original point. 6 = Homing in the forward direction; **ORGP:** ON > OFF, as the original point. 7 = Homing in the reverse direction; **ORGP:** ON > OFF, as the original point. 8 = Current position is the original point. Homing method: 9~A 9 = Homing in the forward direction; the collision point is the original point. A = Homing in the reverse direction; the collision point is the original point.	

API	Mnemonic	Operands	Function	Controllers			
				ES2-C	SS2	SA2/ SE	SX2
340	CANRS	**S_1, S_2, S_3, D_1, D_2**	User-defined CAN communication sending and receiving				

Type / OP	Bit Devices				Word devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		CANRS: 11 steps
S_1					*	*							*				
S_2													*				
S_3													*				
D_1													*				
D_2			*														

PULSE				16-bit				32-bit			
ES2/ EX2/ EC5	SS2	SA2/ SE	SX2	ES2-C	SS2	SA2/ SE	SX2	ES2/ EX2/ EC5	SS2	SA2/ SE	SX2

Operands

S_1 : Communication mode setting

S_2 : Communication ID (MsgID) and data length

S_3 : Starting device where sent source data are stored

D_1 : Starting device where received data are stored

D_2 : Communication completion flag

Explanation

1. The CANRS instruction is applicable to PLCs with CAN BUS communication port, e.g. ES2-C and PLC that connects with let-side communication modules, e.g. DVPCOPM-SL: when it acts as Master and no slaves are configured.
2. There is no limit to the number of times of using the instruction. But only one CAN communication command is allowed to be sent every time. If one command is being sent or received currently, the next CANRS instruction cannot be enabled. And PLC executes the instruction that is being scanned first.
3. The CANRS instruction can use CAN BUS 2.0A (ID 11-bits) (Arbitration) and 2.0B (ID 29-bits) protocols. The default is 2.0B (M1620=OFF). If 2.0A is needed, you can set M1620 to ON when the PLC runs for the first time. Note: this communication protocol can only be set once when switching Stop to Run.
4. When it is set in Master mode, you can use M1621. The default is M1621=OFF (Master mode), and it will send and then receive. When M1621=ON (Slave mode), it will receive and then send during communication.
5. **S_1** sets the communication port number. When DVPCOPM-SL is installed on the left-side of the PLC as the first module, its number is K100; the second one is K101; the eighth one is K107 and so on. If the PLC CPU is ES2-C, its built-in communication port number is K0.
6. **S_2** is the ID of the transmitted message and data length. According to 2.0A or 2.0B protocol, the transmitted data automatically occupies D buffer registers.

   When 2.0A is selected, **S_2** is 11 bits of ID code with the following data transmission format.

S_2 No.	**S_2**	**S_2**+1
Description	Msg. ID	Data Length

When 2.0B is selected, **S_2** (Lo-word) and **S_2**+1 (Hi-word) are both 29 bits of ID code.

S_2 No.	**S_2**	**S_2**+1	**S_2**+2
Description	Msg. ID（Lo-word）	Msg. ID（Hi-word）	Data Length

7. The length of the transmitted message should be in the range of K0~K8 with the unit of byte (8bits). If the setting value (<0 or >8) exceeds the range, the instruction will run at the minimum value 0 or the maximum 8. If the length of the transmitted message is 0, the communication mode will automatically change into the slave mode to receive messages without sending out any data. The mode can be used to monitor the communication packet.
8. **S_3** is the starting device where transmitted data are stored and only the following 8 bits of data are used.

   For example, 4 messages are transmitted with D10 as the starting device. See the data transmission sequence as below.

S_3 No.	D10	D11	D12	D13
Description	Data1	Data2	Data3	Data4

9. If **S_1** is the master mode in which the master will wait to receive data after sending data or the slave mode, the received data will be directly stored in the device specified by **D_1**. D100 is specified by D_1 Here See the stored content format.

   2.0A mode setting:

D_1 No.	D100	D101	D102 ~ D109 (Lower 8 bits)
Description	Msg. ID	Data Length	Data1 ~ Data8

   2.0B mode setting

D_1 No.	D100	D101	D102	D103 ~ D110 (Lower 8 bits)
Description	Msg. ID (Lo-word)	Msg. ID (Hi-word)	Data Length	Data1 ~ Data8

   Note: If the Msg. ID to be received need be specified at the stage of receiving data, set the value of **D_1** beforehand based on the 2.0A/2.0B mode. If the Msg. ID is not specified, please clear the value of **D_1** to 0 before receiving data.

10. If **S_1** is the master broadcast mode, the received data will be stored in the device specified by **D_1**. D100 is specified by **D_1** here. See the storage format as below.

Selecting 2.0A mode: (Here is the introduction of receiving data from 2 slaves. For other data, please increase the Device number specified by **D_1**)

Response sequence	**Data from the first slave**			**Data from the second slave**		**Data from the third slave**
D_1 No.	D100	D101	D102 ~ D109 (Lower 8 bits)	D110	D111~D119	D120~129
Description	Msg. ID	Data Length	Data1 ~ Data8	Msg. ID	Length, Data	ID, Length, Data

Selecting 2.0B mode: (Here is the introduction of receiving data from 1 slaves. For other data, increase the number of **D_1**)

Response sequence	Data from the first slave				Data from the second slave
D_1 No.	D100	D101	D102	D104 ~ D111 (Lower 8 bits)	D112~D122
Description	Msg. ID (Lo-word)	Msg. ID (Hi-word)	Data Length	Data1 ~ Data8	ID, Length, Data

NOTE: if the Msg. ID of the next slave is 0, it indicates there is no data to be received.

11. When the instruction is set to the slave mode and set to receive after sending (M1621=ON, M1622=OFF), the Msg. ID of **D_1** is the receiving condition on ID. Therefore, if there is no requirements on the receivers, use the broadcast mode instead. When the receiver's ID is met with the set ID, the instruction sends data and after sending is complete, a completion flag will be set in **D_2**. If the sending data length is 0, the instruction does not send data and set a completion flag in **D_2**.
12. When the instruction is set to the slave mode and set to receive only (M1621=ON, M1622=ON), the receiving mode will be broadcast. This mode can be ended when timeout (D1177) occurs (M1623=ON) or when the value in D1175 is exceeding 100 packet limit (M1623=ON). If you still need to receive data when this mode is ended, you can stop executing this instruction for a scan cycle and start this mode again. Every time you reset this mode, the receiving log in D1175 will be cleared.
13. **D_2** is communication completion flag and only M device can be used. When the completion flag is ON, it indicates receiving is complete. The completion flag can be set to ON when the instruction is scanned and the communication is complete. From the status of the completion flag, you can tell if the communication is complete. The status of this flag will be clear each time this instruction is executed. You do not need to clear its status.
14. When the instruction is set to the master mode (M1621=OFF), it is recommended to use it to work with D1177 to set the communication timeout. If the communication packet has not been received fully within the specified period of time, the M1623 will be ON. The setting range for timeout is 0-3000 (default 200) and the unit is ms. If the receiving timeout time is set to 0, it indicates that the communication timeout is not limited and the status can be applied to the slave mode.

15. Descriptions on the Flags / Devices

Flags / Devices	Default	Descriptions
M1620	OFF	OFF→ CAN V2.0B protocol ON → CAN V2.0A protocol
M1621 / M1622	OFF/OFF	OFF/OFF → master mode: waiting to receive after sending; if you only need to send data, you can stop executing this instruction in the next scan. After sending is done, the slave response time should be longer than a scan cycle. OFF/ON → master mode: after sending in broadcast mode, receives data from multiple slaves until timeout occurs. ON/OFF → slave mode: sending data, after receiving is done. ON/ON → slave mode: only receiving in broadcast mode without responses
M1623	OFF	ON: communication error; PLC clears this flag when you start the instruction again.
D1175	0	The accumulated packet number (slave number) in the broadcast mode; this number will be accumulated during execution. You can use this number when the completion flag is ON. Up to 100 slaves can be counted, when exceeding 100, the program does not save and stops counting.
D1177	200	Timeout setting; the unit is ms. When the value is set to 0, it indicates this function is disabled until this instruction stops executing. When the mode is in master broadcast, the timeout value cannot be 0. If the timeout value is 0, the system automatically adjusts this value to 200. When timeout occurs, it indicates the broadcast communication is over.

16. The instruction supports the following series and firmware versions

Series	12SA2/ 20SX2	12SE	26SE	32ES2-C	COPM-SL
FW Version	V3.02	V1.88	V2.02	V3.60	V1.40

17. Here is the CAN BUS format and every bit of content for Msg. ID is explained as below.
As 2.0A protocol is selected and the value of **S_2** is H0123, the Msg. ID content is shown in the following table.

Bit No.	15 ~ 11	10 ~ 8	7 ~ 4	3 ~ 0
S_2 value (16bits)	-	1	2	3

As 2.0B protocol is selected, the value of **S_2** is set to H1234 (Lo-word) and **S_2**+1 is H0567 (Hi-word), the Msg. ID content is shown in the following table.

Bit No.	31 ~ 29	28	27 ~ 24	23 ~ 20	19 ~ 16	15 ~ 0
S_2 value (32bits)	-	0	5	6	7	1234

Example 1

System set: DVP12SA211T + DVPCOPM-SL

Mode: Master mode (receiving after sending)

MBB device Diagnostic description as below

Example of Diagnostic Session:

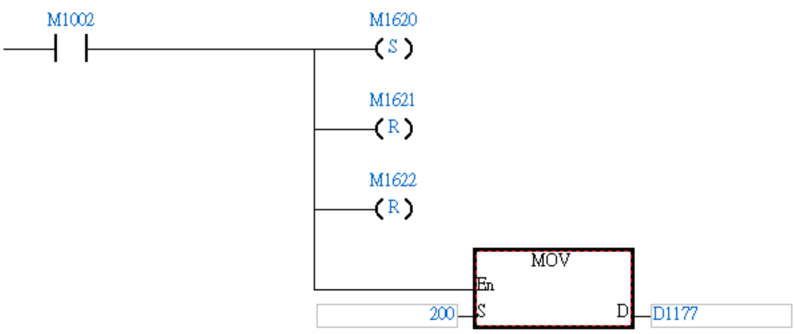
Following is an example of a diagnostic session to write the heartbeat address to 192 (CSM Address).

#	MsgId	Dir	B0	B1	B2	B3	B4	B5	B6	B7	Description
1	0x700	TX	05	2E	FD	01	00	C0	--	--	DID Write Request
2	0x709	RX	03	6E	FD	01	--	--	--	--	Positive Response

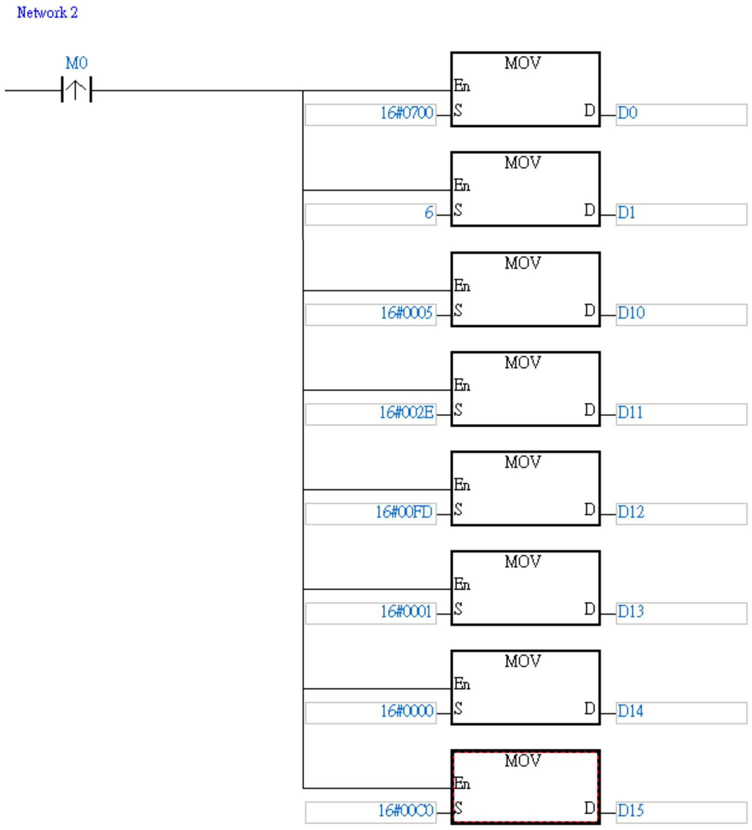
PLC program design:

Step 1) SET M1620 → 2.0A protocol

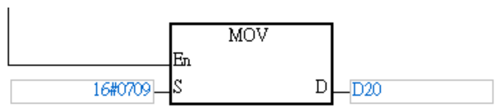
Step 2) RST M1621 & M1622 → Master mode; receiving after sending; set timeout to 200 ms



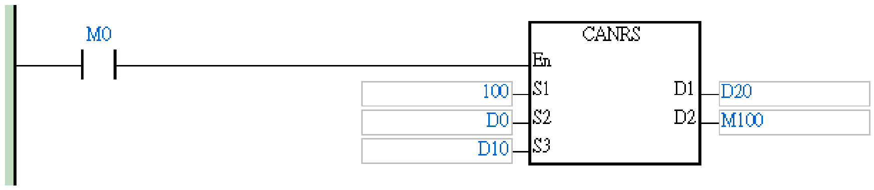
Step 3) LDP M0 → set up MsgId (0x700), data length and data



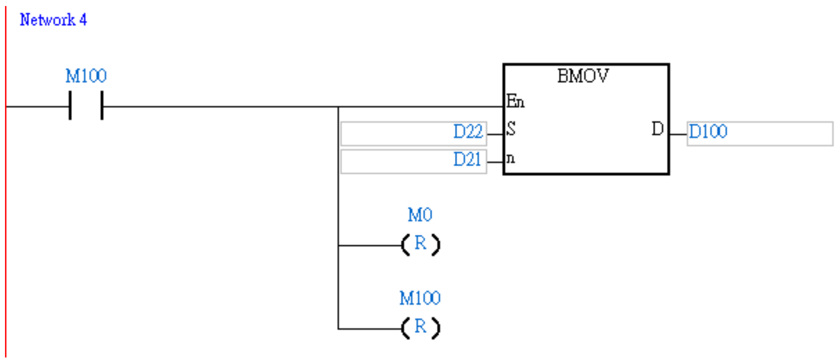
Step 4) LDP M0 → Msg. ID: 0x709



Step 5) LD M0 → use CANRS instruction to set the first left-side module COPM-SL to send data



Step 6) after receiving data is complete, M100 will be ON; stop executing CANRS instruction (RST M0).



Example 2

System set: DVP12SA211T + DVPCOPM-SL

Mode: Master mode (receiving data from all slaves after sending data in broadcast mode)

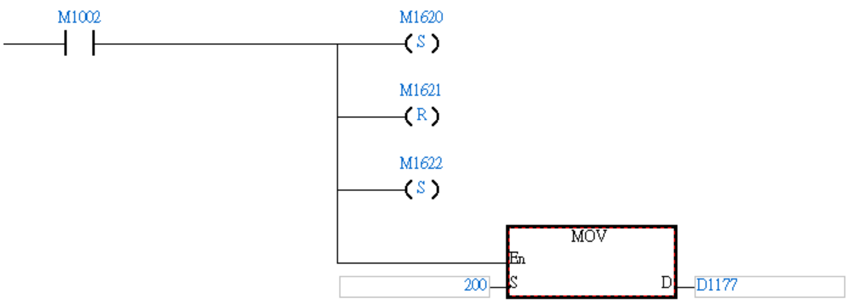
Communication packets:

Identifier	Type	Length	Data
50	Standard	4	11 00 00 00
201	Standard	8	12 FE 86 A4 89 08 87 78
200	Standard	8	12 80 8F 11 8F 18 8F 18
202	Standard	8	12 7D 8E BB 8E F0 8E D8
301	Standard	8	89 08 88 50 87 C8 86 A8
300	Standard	8	8F 1C 8F 14 8F 1C 8F 14
302	Standard	8	8E DC 8E BC 8E F4 8E E4
400	Standard	8	8F 14 8F 1C 8F 1C 8F 1C
401	Standard	8	86 A0 87 08 86 F8 87 48
402	Standard	8	8E DC 8E E4 8E EC 8E E4
500	Standard	8	00 00 00 00 00 00 4F 74
501	Standard	8	00 00 00 00 00 00 4B 98
502	Standard	8	00 00 00 00 00 00 4F 54

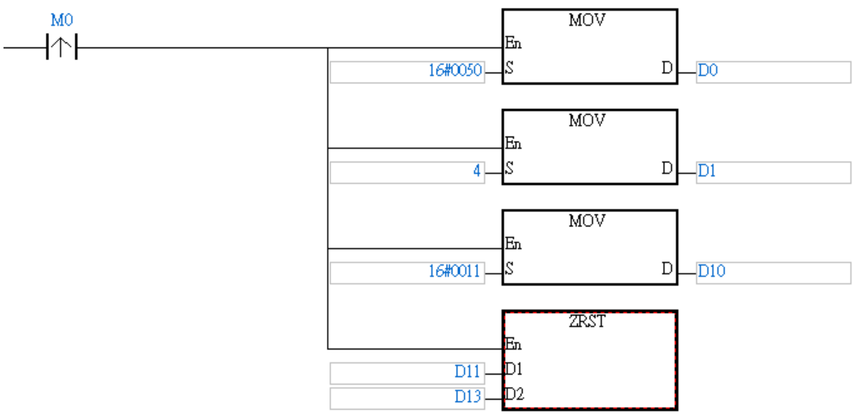
PLC program design:

Step 1) SET M1620 → 2.0A protocol

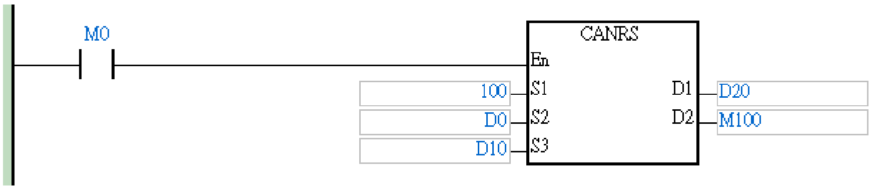
Step 2) RST M1621 and SET M1622 → Master mode; receiving packets from all slaves after sending in broadcast mode; set timeout to 200 ms; if no packets is received in a period of 200 ms, the communication is over.



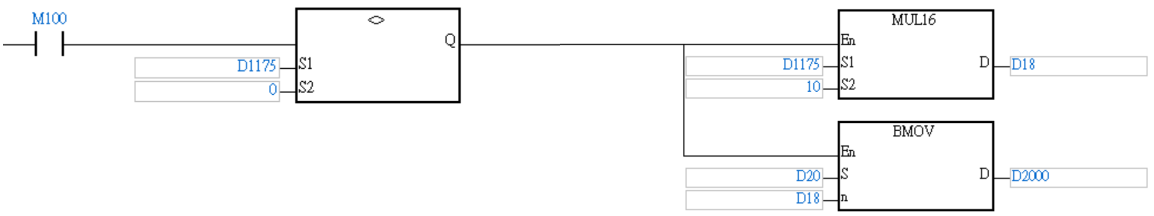
Step 3) LDP M0 → set up MsgId (0x050), data length and data



Step 4) LD M0 → use CANRS instruction to set the first left-side module COPM-SL to send data



Step 5) after receiving data is complete, M100 will be ON; check if the value in D1175 is NOT 0. When there is any value in D1175 other than zero, it indicates D20 has received responses from the slaves.



Example 3

System set: DVP12SA211T + DVPCOPM-SL

Mode: Master mode (receiving after sending)

Example31

System set: DVP12SA211T + DVPCOPM-SL

Mode: Slave mode (receiving first, if the set ID is met, it responds to master)

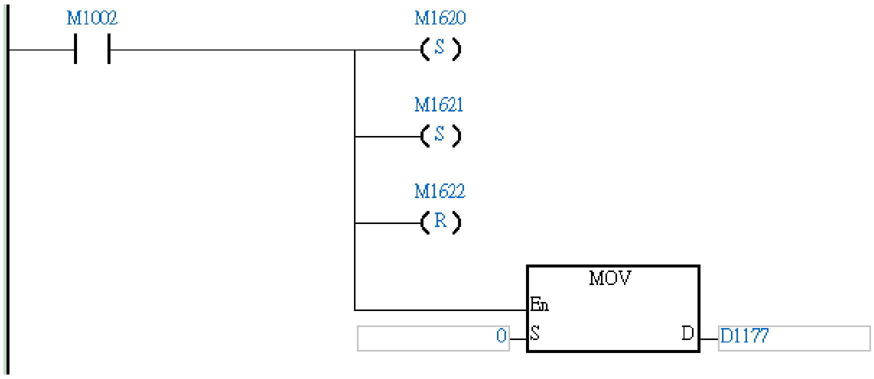
Slave ID is 0x0012 and when the packet contents are in hexadecimal format:

Identifier	Type	Length	Data	Description
012	standard	1	04	Master sending contents
012	standard	4	11 22 33 44	Slave's responses

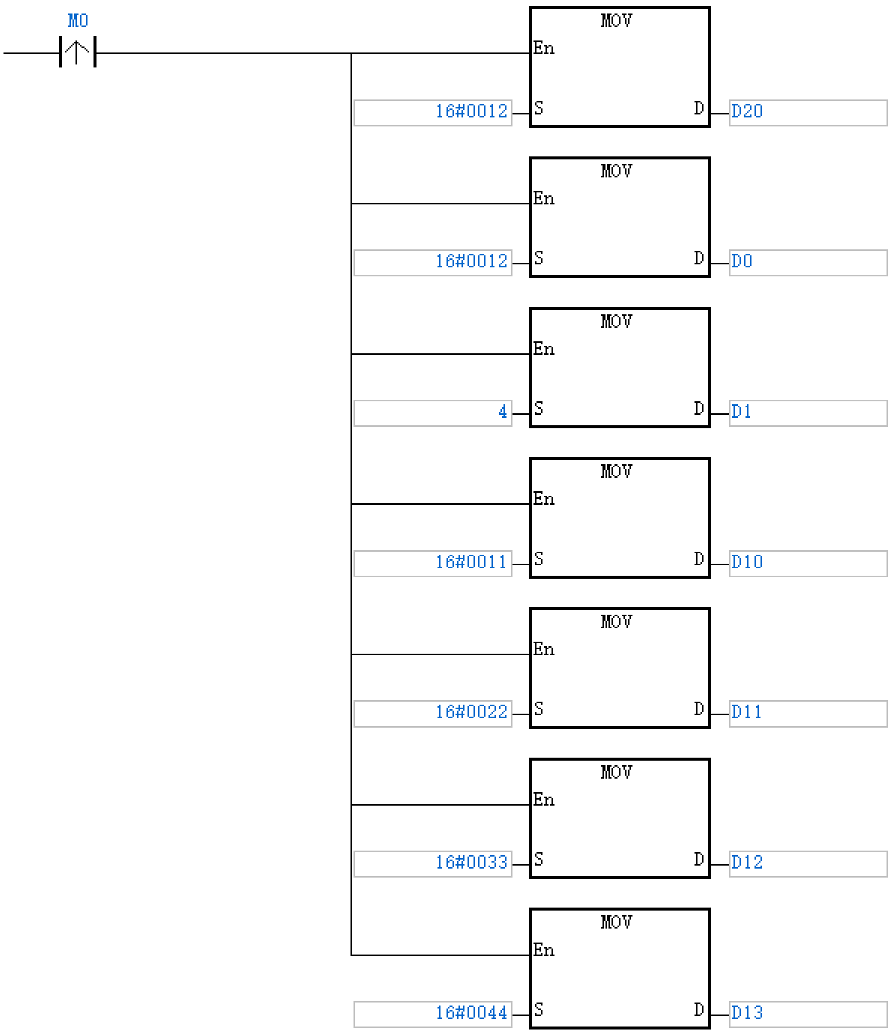
PLC program design:

Step 1) SET M1620 → 2.0A protocol

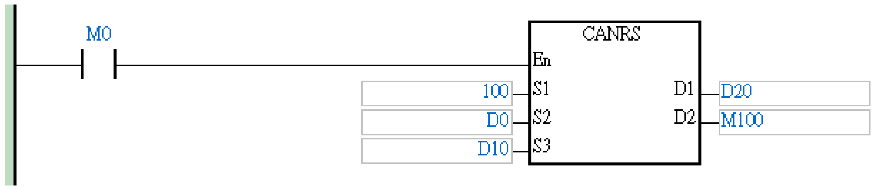
Step 2) RST M1621 and SET M1622 → Slave mode; receiving data from all slaves, if the set ID is met, it responds to master. In receiving mode, the timeout function is not available.



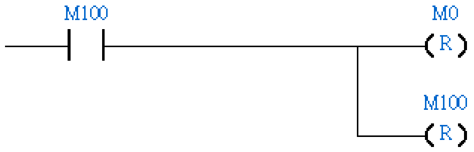
Step 3) LDP M0 → set up MsgId (0x012) and the responses



Step 4) LD M0 → use CANRS instruction to set the first left-side module COPM-SL to respond



Step 5) If M100 is ON, stop executing CANRS instruction.



Note: If Master is going to send data again, you can start executing another CANRS instruction when M100 is ON. Or enter a new ID in D20 and start executing CANRS instruction again.

API	Mnemonic	Operands	Function	Controllers
342	COPRW	**S_1, S_2, S_3, S_4, S_5, D_1**	Read and write CANopen communication data	ES2-C

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	COPRW: 13 steps
S_1					*	*										
S_2					*	*							*			
S_3					*	*							*			
S_4					*	*							*			
S_5													*			
D_1			*													

PULSE	16-bit	32-bit
	ES2-C	

Operands

S_1 : Station address of servo

S_2 : Request code

S_3 : Index

S_4 : Sub-index

S_5 : Read/write device

D : Device

D_1 : Communication completion flag

Explanation

1. It is not available for pulse type instructions. Do not use pulse type contact.

2. For firmware V3.48 or later, it can work with Delta special mode. The range of **S_1** is 1–8. If the setting value is exceeding this range, an error occurs and M1067 will be set to ON, D1067 = 0x0E1A.

3. For firmware V3.60 or later, it can work with Delta special mode and CANopen DS301 mode. This instruction reads and writes CANopen communication data to the servo at the address specified in **S_1**. The range of **S_1** is 1–127. If the value is out of range (<1 or >127), the minimum or maximum value is automatically processed by the instruction as the value of **S_1**.

4. **S_2** can only specify four types of request codes, as shown in the following table.

H23	Writing the 4-byte data
H2B	Writing the 2-byte data
H2F	Writing the 1-byte data
H40	Reading the data. The data length is contained in the SDO response message.

5. For **S3** and **S4**, refer to the object dictionary in the Delta servo operation manual.

6. The definition of **S5** is based on the request code. If the request code is H23, H2B or H2F, **S5** acts as an initial device for the origin. If the request code is H40, **S5** acts as an initial device for the target.

7. You should execute the COPRW instruction only after the INITC instruction is complete in case the parameters are overwritten by the INITC instruction.

8. Any error occurs during operation, M1616 will be set to ON and the servo drive number that shows error will be stored in D6000, error codes in D6001 and STEP that when error occurs in D6002.

   Note: When you use the COPRW instruction, you must edit the process for dealing with communication errors in order to avoid invalid communication occurring as a result of unexpected communication errors.

9. The diagram below shows the timing of the COPRW instruction.

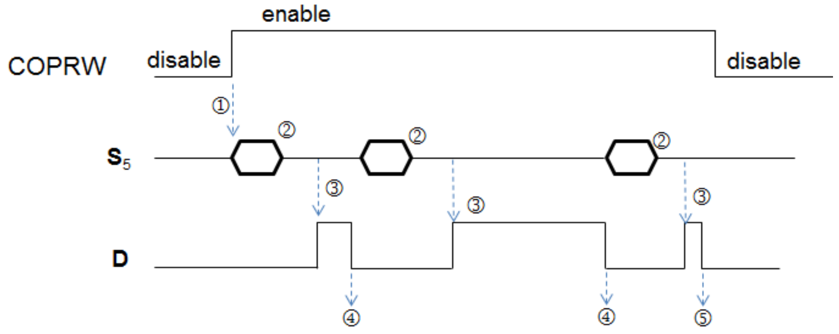
   ① When you enable the COPRW instruction for the first time, the instruction sends the command code immediately if no other CANopen communication is using it.

   ② The instruction sends the command code.

   ③ The code has been sent and the finish flag is set to ON.

   ④ You modify the next data to be sent out. The next command code is sent out immediately after the finish flag is set to OFF.

   ⑤The code has been sent and the COPRW instruction is disabled.



10. Most of the parameters in Delta ASDA are displayed in the decimal format. You can convert the parameters into index addresses, see the example below. 0 is a fixed value for the sub index address.

    Example: The index address of PX-YY=0x2000 +（X << 8）+ YY

    P2-10 = 0x2000 +（0x0002 << 8）+ 0x000A = 0x220A

    P5-04 = 0x2000 +（0x0005 << 8）+ 0x0004 = 0x2504

    P1-44 = 0x2000 +（0x0001 << 8）+ 0x002C = 0x212C

11. Most of the parameters in Delta inverter are also displayed in the decimal format. Use the following formula to convert the parameters.

    Example: The index address of PXX-YY=0x2000 + XX (hexadecimal);
    The sub index address is YY+1 (hexadecimal)

    The index address of P10-15 = 0x2000 + 0x000A = 0x200A
    The sub index address is 0x0F+1= 0x10

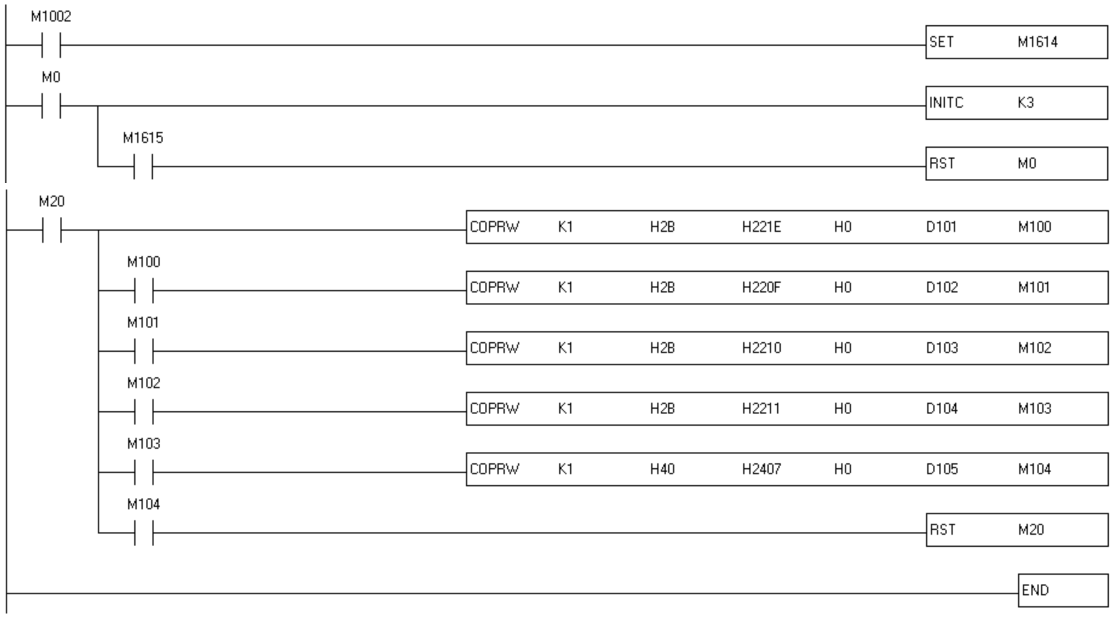
3

Example

1. When M0 changes from OFF to ON, the INITC instruction starts to initialize the servos at station addresses 1–3, until M1615 is ON.

2. When M20 changes from OFF to ON, the PLC writes the 2-byte data in D100-D104, and reads the value of P4-07 and stores the value in D105, using the COPRW instruction. When the writing is complete, M100-M104 is ON.

Parameters	Request code	Device for storage
P2-30	H2B_Write	D101
P2-15	H2B_Write	D102
P2-16	H2B_Write	D103
P2-17	H2B_Write	D104
P4-07	H40_Read	D105


M1002
SET M1614
M0
INITC K3
M1615
RST M0
M20
COPRW K1 H2B H221E H0 D101 M100
M100
COPRW K1 H2B H220F H0 D102 M101
M101
COPRW K1 H2B H2210 H0 D103 M102
M102
COPRW K1 H2B H2211 H0 D104 M103
M103
COPRW K1 H40 H2407 H0 D105 M104
M104
RST M20
END

3.6.23 Module Instructions

API	Mnemonic		PULSE	Function	Applicable to				STEPS	
	16 bits	32 bits			ES2 EX2	SS2	SA2 SE	SX2	16-bit	32-bit
344	–	DPUCONF	–	Setting output control parameters of PU module	✓	-	-	-	-	49
345	PUSTAT	–	–	Reading PU module output state	✓	-	-	-	17	-
346	–	DPUPLS	–	PU module pulse output (no acceleration)	✓	-	-	-	-	29
347	–	DPUDRI	–	Relative position output of PU module (with acceleration and deceleration)	✓	-	-	-	-	29
348	–	DPUDRA	–	Absolute addressing output of PU module (with acceleration and deceleration)	✓	-	-	-	-	29
349	–	DPUZRN	–	PU module homing	✓	-	-	-	-	33
350	–	DPUJOG	–	PU module jog output	✓	-	-	-	-	25
351	–	DPUMPG	–	PU module MPG output	✓	-	-	-	-	41
352	–	DPUCNT	–	High-speed counter function of PU module	✓	-	-	-	-	33
353	PUX	–	✓	Setting PU module input point mode	✓	-	-	-	15	-
354	–	DPULS	✓	Setting PU module software limits	✓	-	-	-	-	29

API	Mnemonic		Operands	Function	Controllers
344	D	PUCONF	Module~ErrCode	Setting output control parameters of PU module	ES2/EX2, SS2, SA2 SE, SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
																DPUCONF: 49 steps (To be used in ISPSoft)
Module					*	*							*			
Axis					*	*							*			
Mode					*	*							*			
SSpeed					*	*							*			
Atime					*	*							*			
Dtime					*	*							*			
MSpeed					*	*							*			
Z_no					*	*							*			
Offset					*	*							*			
Done		*	*	*												
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Axis**: Output axis number **Mode**: Output mode

SSpeed: Speed for starting/ ending frequency **Atime**: Acceleration time

Dtime: Deceleration time **MSpeed**: Maximum output frequency

Z_no: Number of Z phases to look for after returning to the original point

Offset: Specify the number of outputs after returning to the original point

Done: Completion flag **Error**: Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. The timing to set this instruction is when En changes from OFF to ON. WPLSoft does NOT support this instruction.

2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.

3. **Axis** sets the output axis number for the specified PU module. The setting values 1~2 represent the axis1~axis2 output of the specified PU module respectively. If the PU module has no corresponding axis number for output, the error flag **Error** will change to ON. See the following combination of axis numbers and corresponding output points of PU modules.

PU module name	Axis 1	Axis 2
DVP02PU-E2	Y0 / Y1	Y2 / Y3

4. **Mode** sets the output mode of an output axis and the setting values are explained in the following table.

Output mode value	Description	Remark
0	Single-point pulse output (An even-number point for output only)	E.g. Y0 or Y2 for output
1	Pulse (An even-number point) + direction (An odd-number point)	E.g. Y0 is for the pulse and Y1 is for the direction. Y1: ON, negative direction; Y1: OFF, positive direction
2	CW (An even-number point) + CCW (An odd-number point)	E.g. Y0 is for CW (positive direction) and Y1 is for CCW (negative direction)
3	Phase A (An even-number point) + Phase B (An odd-number point)	E.g. Y0 is for phase A and Y1 is for phase B. When phase A is leading phase B: positive direction; when phase B is leading phase A: negative direction
Others	Automatically switch to mode 1 (default value)	

5. **SSpeed~ Offset**

See the explanation of the following non-latched parameters and setting values. If the setting values exceed the range, the instruction will automatically be executed at the minimum or maximum value.

Parameter	Function	Range	Default	Remark
SSpeed	Starting/ending frequency	0 ~ 10,000 (Unit: Hz)	100	
Atime	Acceleration time	0 ~ 10,000 (Unit: ms)	100	
Dtime	Deceleration time	0 ~ 10,000 (Unit: ms)	100	

MSpeed	Maximum output frequency	100 ~ 200,000 (Unit: Hz)	100K	A 32-bit value
Z_no	Number of Z phase signals to seek after returning to the origin.	-100 ~ 100 (Unit: times)	0	0: disabled
Offset	Outputs the offset position after the homing is finished and Z phase seeking is done.	-10,000 ~ 10,000 (Unit: pulses)	0	0: disabled

6. **Done**, an output of the specified PU module has been set as the completion flag. When **Done** is On, it indicates that the parameter setting is successful. You can continue to perform positioning output based on the On state of the completion flag. The clearing of the **Done** flag need be conducted by manual. The **Done** flag changes to ON only when the setting is completed.

7. **Error**, an output of the specified PU module is a parameter error flag. Most parameter ranges are filtered automatically by the PLC. Thus if the error flag is ON, it means that there is no specified PU module or the PU module number is wrong or the output axis number is incorrect.

8. The instruction is a pulse instruction. Even if the A contact is adopted as the condition contact, PU module parameters are also set only when the instruction is started. Therefore, if a parameter value is to be updated, restart the instruction to make the parameter set again.

9. Since the set parameters are delivered through the module communication command, confirm the state of the output **Done** or **Error** before a parameter value is modified and then proceed with relevant operations.

10. **ErrCode** shows error codes. See the description as follows.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.

Programming Example: Refer to the description of DPUDRI instruction (API 347) for more information.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2	SS2	SA2 SE	SX2
345	PUSTAT	Module~ErrCode	Reading PU module output state				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
Module					*	*							*			PUSTAT: 17 steps (To be used in ISPSoft)
Axis					*	*							*			
ZeroS	*	*	*	*									*			
C_Posi													*			
Execute		*	*	*												
Pause		*	*	*												
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Axis**: Output axis number **ZeroS**: Clear present output position to 0 **C_Posi**: Current output position **Execute**: Execution flag P**ause**: Pause flag **Error**: Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. The timing to set this instruction is when En changes from OFF to ON. WPLSoft does NOT support this instruction.

2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.

3. **Axis** sets the output axis number for the specified PU module. The setting values 1~2 represent the axis1~axis2 output of the specified PU module respectively. If the PU module has no corresponding axis number for output, the error flag **Error** will change to ON. See the following combination of axis numbers and corresponding output points of PU modules.

4. **ZeroS** clears the present output position to 0. If the present axis position is to be cleared to 0, set **ZeroS** from OFF to ON when the instruction is started.

5. **C_Posi** sets the present position of the output axis for the specified PU module. The parameter value is a latched value and stored in the PU module.

6. **Execute** is an read-only flag which means the output axis of the specified PU module is outputting or not. When **Execute** is On, it means the output is being conducted. When **Execute** is Off, it means the output axis is unused and can accept the next output command.
7. **Pause** is an read-only flag to control the output axis of the specified PU module to pause its output. When **Pause** is On, it means the output is paused, the present velocity is 0 and the present output has not reached the specified target output position. If you restore the output, the flag will be cleared automatically.

   Note: While **Pause** is On, **Execute** is constantly On as well.
8. **Error** is an read-only error flag which means an error occurs during the reading of the specified PU module. Refer to the explanation of error codes in **ErrCode**.
9. After the PUSTAT instruction gives the pause command, the flags **Execute, Pause and Error** become read-only flags and at the moment, their states cannot be modified. The **Execute, Pause and Error** flags can be set or cleared only when the PUSTAT instruction is turned off.
10. For PU module state, check out the data exchange function of the special extension module through SM1183. Refer to Section 2.16 Additional Remarks on Special Auxiliary Relays and Special Data Registers in this manual for details.
11. **ErrCode** shows error codes and the explanations are seen in the following table.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.
16#1403	There is no such output axis number in the PU module.

Programming Example: Refer to the description of DPUDRI instruction (API 347) for more information.

API	Mnemonic		Operands	Function	Controllers
346	D	PUPLS	Module~ErrCode	PU module pulse output (no acceleration)	ES2/EX2 SS2 SA2 SE SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
Module					*	*							*			DPUPLS: 29 steps (To be used in ISPSoft)
Axis					*	*							*			
TarPulse					*	*							*			
TarSpeed					*	*							*			
Done		*	*	*												
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Axis**: Output axis number **TarPulse**: Target number of output pulses **TarSpeed**: Target output frequency **Done**: Completion flag **Error:** Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. When En setting is set to ON, this instruction would be effective. Once the setting changes to OFF, Output pulse would be terminated immediately. WPLSoft does NOT support this instruction.
2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.
3. **Axis** sets the output axis number for the specified PU module. The setting values 1~2 represent the axis1~axis2 output of the specified PU module respectively. If the PU module has no corresponding axis number for output, the error flag **Error** will change to ON.
4. **TarPulse** sets the number of output pulses. The pulse number is a positive signed 32-bit value. When the value is 0, it means the output is always being performed, the number of output pulses is not limited and the output is not stopped until the instruction is disabled. When the value is less than 0, the PLC automatically uses 2's complement to transform the value into a positive integer as the number of output pulses.

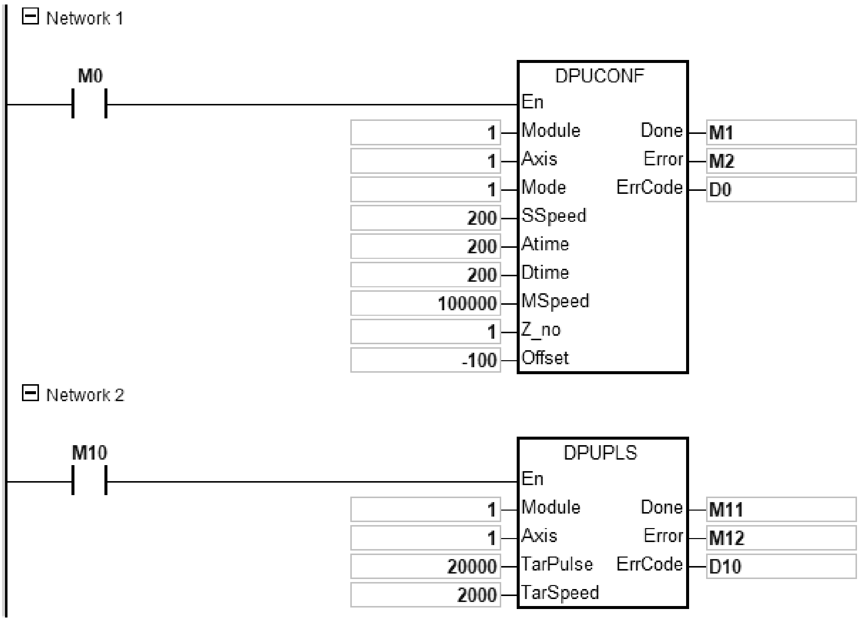
5. **TarSpeed** sets the target output speed (Unit: Hz). The input value is a signed 32-bit value within the range of -200,000 (-200K) ~ 200,000 (200K). You can modify the target frequency any time after the instruction is enabled and the PU module will automatically switch to the newly set target frequency after outputting a full pulse.
6. When **TarSpeed** is a positive number (>0), it means that the "positive direction" output point is Off. When **TarSpeed** is a negative number (<0), it means that the "negative direction" output point is On. When **TarSpeed** is 0, it means that the output will be paused after the being executed pulse is output fully.
7. The instruction does not support the function of acceleration and deceleration. Use the DPUDRI instruction instead if you need the function of acceleration and deceleration.
8. The instruction can be used for the speed change. While the instruction is being executed, you can change the value of **TarSpeed** so as to change the output speed. When the setting value exceeds the maximum frequency, the instruction would be executed at the maximum frequency. But changing the speed would not change the direction. If the direction is to be changed, set the value of **TarSpeed** to 0 first and then modify the target speed.
9. When the outputs have reached the pulse number specified by **TarPulse**, the **Done** flag changes to ON. The **Done** flag need be cleared by manual. The instruction sets the completion flag to ON only when the output is completed.
10. The instruction can be used with the software and hardware limit points. When the limits are triggered, the output stops immediately and the **Error** flag changes to ON.
11. If any error occurs as the instruction is in process of the output, the **Error** flag changes to ON. Refer to the error codes **ErrCode** shows for the trouble shooting.

    The error codes that **ErrCode** shows are listed in the following table.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.
16#1403	There is no such output axis number in the PU module.
16#1405	The output axis specified by the PU module is outputting data. It is not allowed to specify the output repeatedly.
16#1406	PU module stops Output pulse when the positive limit is reached.
16#1407	PU module stops Output pulse when the negative limit is reached.

Programming Example:

1. When M0 is ON, the DPUCONF instruction for axis 1 is executed to modify the parameters by setting **Mode** to 1 (Pulse Y0 + direction Y1), **SSpeed** to 200Hz, **Atime** to 200ms, **Dtime** to 200ms and **MSpeed** to 100kHz. After the output of **Done** is completed, M1 is ON.
2. When M10 is ON, the DPUPLS instruction for axis 1 starts to output 20,000 pulses from Y0 at the frequency of 2KHz (without acceleration and deceleration). Y1 is OFF, which indicates the positive direction and M11 is ON after the pulse output is finished.



API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
347	D	PUDRI	Module~ErrCode	Relative position output of PU module (with acceleration and deceleration)				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
Module					*	*							*			DPUDRI: 29 steps (To be used in ISPSoft)
Axis					*	*							*			
RTarPosi					*	*							*			
TarSpeed					*	*							*			
Done		*	*	*												
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Axis**: Output axis number **RTarPosi**: Number of output pulses for relative positioning **TarSpeed**: Target output frequency **Done**: Completion flag **Error**: Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. When En setting is set to ON, this instruction would be effective. Once the setting changes to OFF, Output pulse would be terminated immediately. WPLSoft does NOT support this instruction.

2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.

3. **Axis** sets the output axis number for the specified PU module. The setting values 1~2 represent the axis1~axis2 output of the specified PU module respectively. If the PU module has no corresponding axis number for output, the error flag **Error** will change to ON.

4. **RTarPosi** sets the position for relative positioning. The pulse number is a signed 32-bit value. When the value is greater than 0, the output will go in the positive direction (and the direction output point is off). When the value is less than 0, the output will go in the negative direction (and the direction output point is on). When the value is 0, the output completion flag **Done** changes to ON.

5. **TarSpeed** sets the target output frequency (Unit: Hz). The frequency value is a positive signed 32-bit integer. When the value is less than 0, the instruction will automatically use 2's complement to transform the value into a positive integer. When the value is 0, the instruction will notify the module to enter the pause mode. The actual output is decelerated at the deceleration rate till the output speed is equal to 0 and the pause flag changes to ON. Refer to PUSTAT instruction for more details.

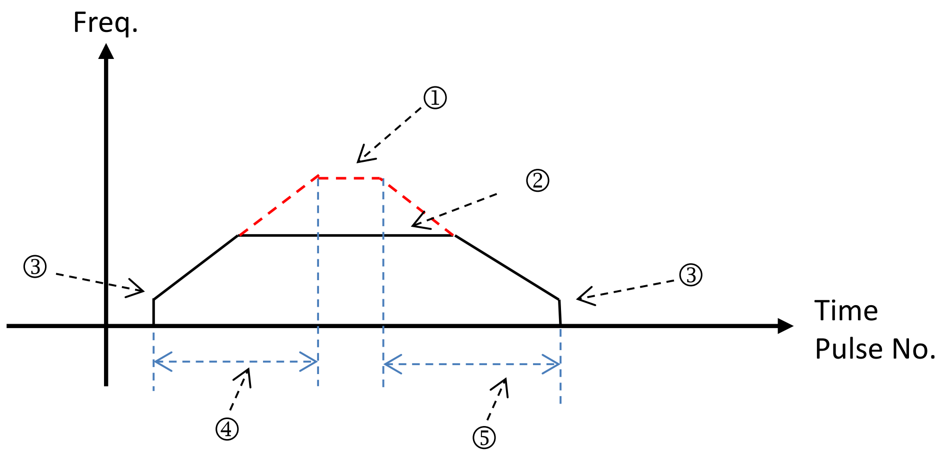
   See the setting range of **TarSpeed** for the module in the following table.

Module Name	TarSpeed Setting Range
DVP02PU-E2	-200,000（-200K）~ 200,000（200K）

6. After the output is started, the target frequency is allowed to change any time. In the actual frequency change, the PLC will automatically change the frequency based on the set acceleration and deceleration rate in the DPUCONF instruction. When the modified speed exceeds the allowed maximum frequency, the output will be performed at the maximum frequency.
7. When the outputs have reached the pulse number for relative positioning specified by **RTarPosi**, the **Done** flag changes to ON. The **Done** flag need be cleared by manual. The instruction sets the completion flag to ON only when the output is completed.
8. The instruction can be used with the software and hardware limit points. The output stops immediately and the **Error** flag changes to ON when the limits are triggered.
9. If any error occurs as the instruction is in process of the output, the **Error** flag changes to ON. Refer to the error codes that **ErrCode** shows for the trouble shooting.
10. The error codes that **ErrCode** shows are listed in the following table.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.
16#1403	There is no such output axis number in the PU module.
16#1405	The output axis specified by the PU module is outputting data. It is not allowed to specify the output repeatedly.
16#1406	PU module stops Output pulse when the positive limit is reached.
16#1407	PU module stops Output pulse when the negative limit is reached.

11. Illustration of the acceleration and deceleration curve of the DPUDRI instruction



① : Maximum output frequency value. Refer to the setting in the DPUCONF instruction for the parameter setting. Alternatively, set the parameter value through HWCONFIG.

②: The target frequency specified by the PU module output instruction. The target frequency output must not exceed the maximum output frequency. If the maximum output frequency is exceeded, the maximum output frequency is regarded as the output frequency.

③: Starting/ending output frequency value. Refer to the setting in the DPUCONF instruction for the parameter setting. Alternatively, set the parameter value through HWCONFIG.

④: The acceleration time value. Refer to the setting in the DPUCONF instruction for the parameter setting. Alternatively, set the parameter value through HWCONFIG.

⑤: The deceleration time value. Refer to the setting in the DPUCONF instruction for the parameter setting. Alternatively, set the parameter value through HWCONFIG.

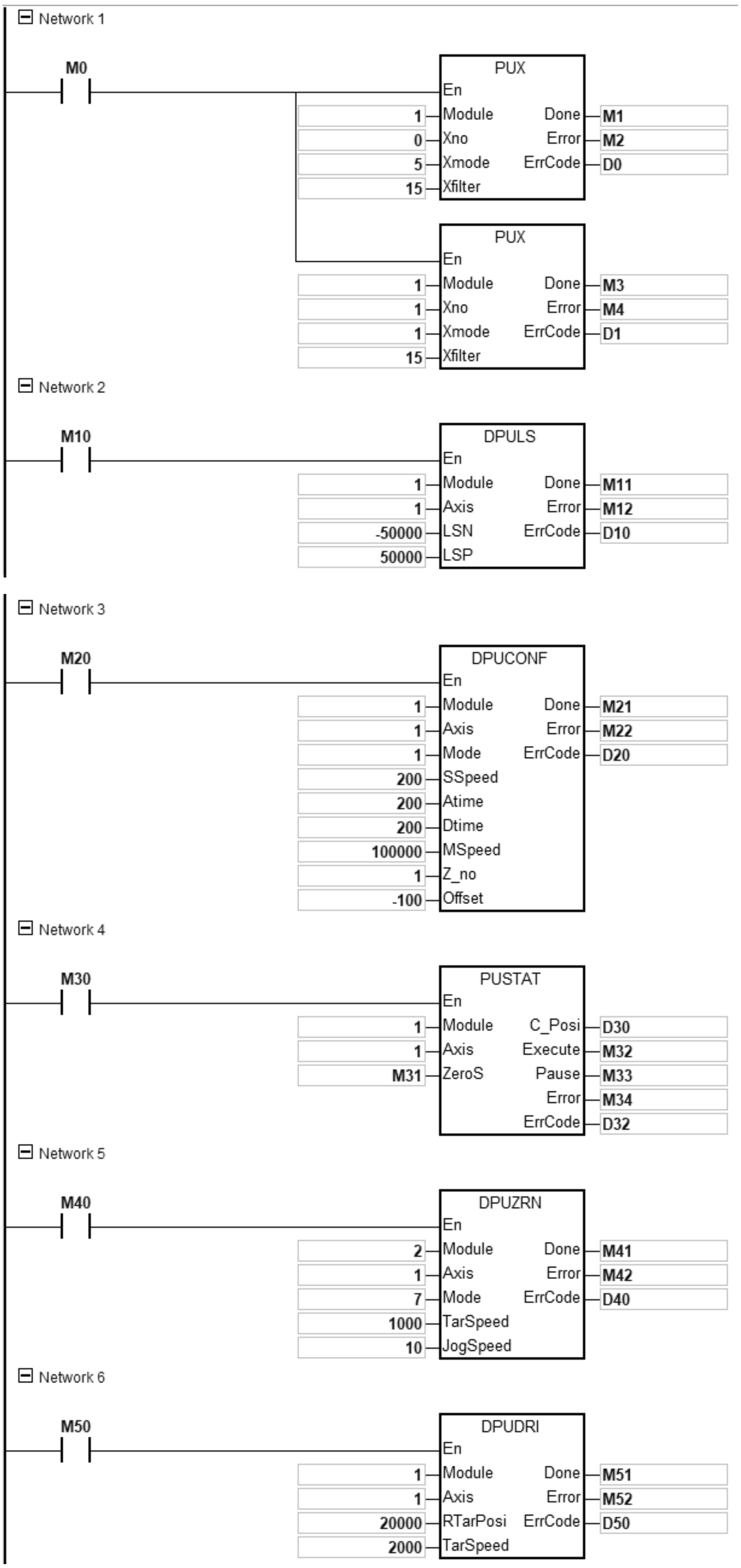
The acceleration and deceleration that the PU module controls is performed according to the fixed slope. So the actual acceleration time and deceleration time change based on the output target frequency. The formula for calculation of acceleration rate and deceleration rate are respectively shown as follows.

(Max. output frequency - starting frequency)/acceleration time;

(Max. output frequency - ending frequency)/deceleration time.

Programming Example:

1. When M0 is ON, the PUX instruction is executed to set the input parameters of DVP02PU-E2 module with “Axis 1, DOG, Rising-edge triggered” for input point X0, “Axis 1, Z phase, Rising-edge triggered” for input point X1 and “15m” for X0 / X1 filter time. When the output of **Done** is completed, M1/M3 changes to ON.
2. When M10 is ON, the DPULS instruction for axis 1 is executed to set the software limit points with “-50000” for LSN and “50000” for LSP. When the output of **Done** is completed, M11 changes to ON.
3. When M20 is ON, the DPUCONF instruction for axis 1 is executed to change the parameters by setting **Mode** to 1 (Pulse Y0+ Direction Y1), **SSpeed** (start speed) to 200Hz, **Atime** (acceleration time) to 200ms, **Dtime** (deceleration time) to 200ms, **MSpeed** (maximum speed) to 100kHz, Z_NO (Number of Z phases to look for after returning to the home position) to 1 and **Offset** (number of outputs after homing is finished) to -100. And M21 changes to ON as the output of Done is completed.
4. When M30=ON, the PUSTAT instruction for axis 1 is executed to read the PU module output state. To clear current output position, you can set M31 to ON so that the current position of axis 1 (rising-edge triggered) in D30 would be cleared to 0.
5. When M40 is ON, the DPUZRN instruction for axis1 starts to perform homing and the PUSTAT instruction displays the current position in D30. The output point Y0 outputs pulses at the frequency of 1kHz and the search for the home starts in positive direction. Once the near home signal (DOG) is reached and X0 is ON, the axis starts to decelerate and then moves at the **Jogspeed** of 100Hz in the negative direction. When X0=OFF, the axis moves in the positive direction to search for Z phase until the first rising-edge triggered signal at X1 (Z phase) is detected, then it moves toward negative direction after 100 output pulses are completed. Finally, M41 changes to ON after the output of **Done** is finished.
6. When M50 is ON, the DPUDRI instruction for axis 1 starts to perform relative positioning output. The PUSTAT instruction displays the current position in D0 and the output point Y0 outputs 20,000 pulses at the frequency of 2kHz (relative addressing). Y1 is OFF, which indicates that the direction is positive and the PUSTAT instruction displays the current position in D30. Finally, M51 changes to ON after the output of **Done** is completed.


Network 1
M0
PUX
En
1 Module Done M1
0 Xno Error M2
5 Xmode ErrCode D0
15 Xfilter
PUX
En
1 Module Done M3
1 Xno Error M4
1 Xmode ErrCode D1
15 Xfilter
Network 2
M10
DPULS
En
1 Module Done M11
1 Axis Error M12
-50000 LSN ErrCode D10
50000 LSP
Network 3
M20
DPUCONF
En
1 Module Done M21
1 Axis Error M22
1 Mode ErrCode D20
200 SSpeed
200 Atime
200 Dtime
100000 MSpeed
1 Z_no
-100 Offset
Network 4
M30
PUSTAT
En
1 Module C_Posi D30
1 Axis Execute M32
M31 ZeroS Pause M33
Error M34
ErrCode D32
Network 5
M40
DPUZRN
En
2 Module Done M41
1 Axis Error M42
7 Mode ErrCode D40
1000 TarSpeed
10 JogSpeed
Network 6
M50
DPUDRI
En
1 Module Done M51
1 Axis Error M52
20000 RTarPosi ErrCode D50
2000 TarSpeed

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
348	D	PUDRA	Module~ErrCode	Absolute addressing output of PU module (with acceleration and deceleration)				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
Module					*	*							*			DPUDRA: 29 steps (To be used in ISPSoft)
Axis					*	*							*			
ATarPosi					*	*							*			
TarSpeed					*	*							*			
Done		*	*	*												
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Axis**: Output axis number **ATarPosi**: Number of output pulses for absolute addressing **TarSpeed**: Target output frequency **Done**: Completion/pause flag **Error**: Error flag **ErrCode**: Error code

Explanations:

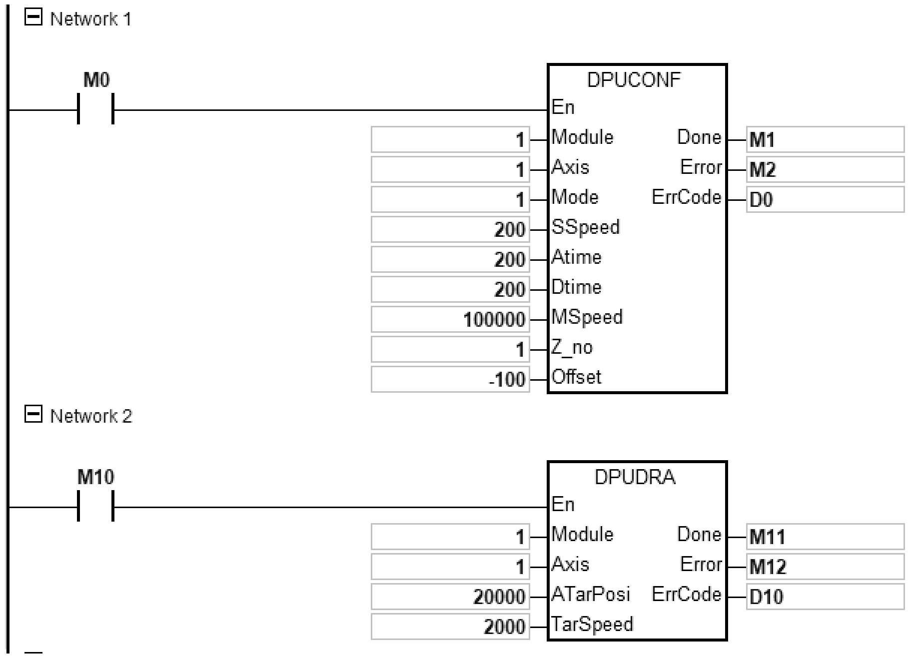
1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. When En setting is set to ON, this instruction would be effective. Once the setting changes to OFF, Output pulse would be terminated immediately. WPLSoft does NOT support this instruction.

2. **Axis** sets the output axis number for the specified PU module. The setting values 1~2 represent the axis1~axis2 output of the specified PU module respectively. If the PU module has no corresponding axis number for output, the error flag **Error** will change to ON.

3. **ATarPosi** is the position for absolute addressing. The input pulse number is a signed 32 bit value. The PU module will automatically compare it with the present position. If the comparison result is greater than 0, the output will be conducted in the positive direction (and the direction output point is off). If the comparison result is less than 0, the output will be conducted in the negative direction and the direction output point is on). When the value is 0, the instruction sets the **Done** flag to ON.
4. Refer to the DPUDRI instruction for the explanation of other parameters.

Programming Example:

1. When M0 is ON, the DPUCONF instruction for axis 1 is executed to modify the parameters by setting **Mode** to 1 (Pulse Y0+ Direction Y1), **SSpeed** (the speed for starting) to 200Hz, **Atime** (acceleration time) to 200ms, **Dtime** (deceleration time) to 200ms and **MSpeed** (maximum output frequency) to 100kHz. And M1 changes to ON as the output of **Done** is completed.
2. When M10 is ON, the DPUDRA instruction for axis 1 is executed to output pulses from Y0 at the frequency of 2kHz until the current position reaches 20,000 (absolute addressing). When Y1 is OFF, the direction is positive. And M11 changes to ON as the output of **Done** is completed.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
349	D	PUZRN		Module~ErrCode	PU module homing				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
Module					*	*							*			DPUZRN: 33 steps (To be used in ISPSoft)
Axis					*	*							*			
Mode					*	*							*			
TarSpeed					*	*							*			
JogSpeed					*	*							*			
Done		*	*	*												
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Axis**: Output axis number **Mode**: Homing mode selection

TarSpeed: Maximum output frequency for the homing **JogSpeed**: The jog frequency for the homing **Done**: Completion flag **Error**: Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. When En setting is set to ON, this instruction would be effective. Once the setting changes to OFF, Output pulse would be terminated immediately. WPLSoft does NOT support this instruction.

2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.

3. **Axis** sets the output axis number for the specified PU module. The setting values 1~2 represent the axis1~axis2 output of the specified PU module respectively. If the PU module has no corresponding axis number for output, the error flag **Error** will change to ON.

4. **Mode** sets a homing mode. The explanation of modes is shown in the following table.

Mode value	Function	Use matching input points by PUX instruction	Remark
0	Directly clear the current position to 0.	None	
1	The axis starts to go toward the negative direction and then stops after leaving the DOG point position	DOG	
2	The axis starts to go toward the positive direction and then stops after leaving the DOG point position	DOG	
3	After Mode 1 is finished, seek the set number of Z phases.	DOG and Z phase input	Use DPUCONF instruction to set up the number of Z phases.
4	After Mode 2 is finished, seek the set number of Z phases.	DOG and Z phase input	
5	After Mode 1 is finished, output the offset position.	DOG	Use DPUCONF instruction to set up the offset position.
6	After Mode 2 is finished, output the offset position.	DOG	
7	After Mode 1 is finished, seek the set number of Z phases and then output the offset position.	DOG and Z phase input	Use DPUCONF instruction to set up the number of Z phases and offset position.
8	After Mode 2 is finished, seek the set number of Z phases and then output the offset position.	DOG and Z phase input	
255	Modify the current output position for the axis.	None	Use the setting value of **TarSpeed**
Others	Reserved		

Note 1: The specified homing behavior may not be realized if the input point for the selected mode is not used together with the PUX instruction. Refer to the following No.12 Homing modes for more information.

5. **TarSpeed** sets the maximum output frequency for the homing. The setting value is a signed 32-bit value. When **Mode** value is between 1~8, the range of the setting value is -200,000 ~ -100 (Hz) and 100 ~ 200,000 (Hz). If **Mode** value is 255, **TarSpeed** value will become the present output position value of the PU module.
6. **JogSpeed** is the jog frequency for reaching the home position and also represents the start/end frequency for homing. The setting value is a signed 16-bit value within the range of 1~10,000 (Hz).
7. When the specified home position is reached during the instruction is executed, the **Done** flag changes to ON. The **Done** flag need be cleared by manual. The instruction sets the completion flag to ON only when the output is completed.
8. The instruction does not support software limit points. It can be used with hardware limit points only. When a hardware limit point is triggered during the output, the **Error** flag will be set to ON.

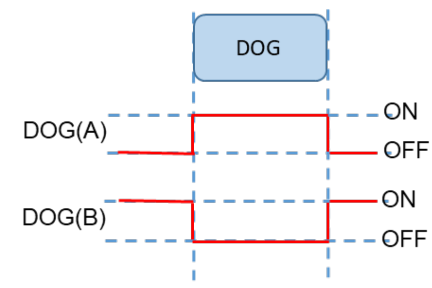
   The following cases and corresponding axis actions occur when a hardware limit point is triggered.

Case	Action
DOG is not entered	The axis stops immediately, then speeds up toward the opposite direction from the frequency specified by JogSpeed until the frequency specified by TarSeed is reached and continues to seek the DOG signal.
At DOG	The axis stops immediately, then moves toward the opposite direction at the frequency specified by JogSpeed and continues to seek the DOG signal.
DOG is moved away from.	The axis stops immediately.

9. If any error occurs as the instruction is in process of the output, the **Error** flag changes to ON. Refer to the error codes that **ErrCode** shows for the trouble shooting.
10. The error codes that **ErrCode** shows are listed in the following table.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.
16#1403	There is no such output axis number in the PU module.
16#1405	The output axis specified by the PU module is outputting data. It is not allowed to specify the output repeatedly.
16#1406	PU module stops Output pulse when the positive limit is reached.
16#1407	PU module stops Output pulse when the negative limit is reached.

11. Explanation of DOG (A) and DOG (B) signals

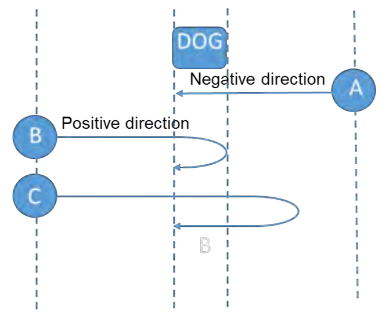
<table>
<tr><th>Contact Type</th><th colspan="2"></th></tr>
<tr><td>DOG (A)</td><td>The DOG signal emerges and the axis enters DOG when the contact switches from OFF to ON.
The DOG signal disappears and the axis leaves DOG when the contact switches from ON to OFF.</td><td rowspan="2">
</td></tr>
<tr><td>DOG (B)</td><td>The DOG signal emerges and the axis enters DOG when the contact switches from ON to OFF.
The DOG signal disappears and the axis leaves DOG when the contact switches from OFF to ON.</td></tr>
</table>

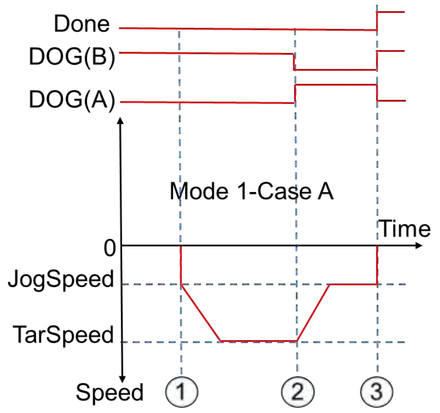
12. Explanation of homing modes

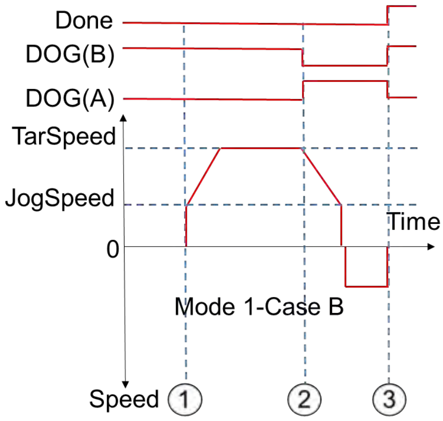
Mode 0：Directly clear the current position to 0.

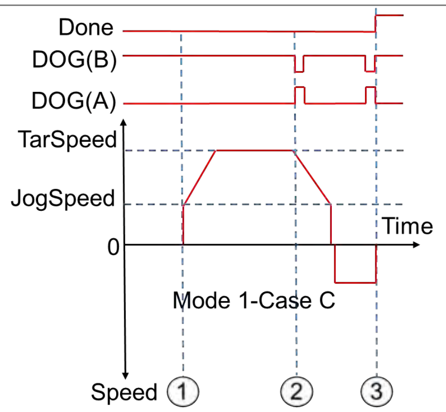
Mode 1：The axis starts to go toward the negative direction and then stops after leaving the DOG point position

Description of cases under mode 1	
Case A	The motion starts in the negative direction. As the DOG point is encountered, The axis starts to decelerate until the JOG speed is reached and then the axis stops while leaving the DOG point.
Case B	The motion starts in the positive direction; the deceleration time is less than the duration for the DOG signal.
Case C	The motion starts in the positive direction; the deceleration time is greater than the duration for the DOG signal.









① The DPUZRN instruction is started; the axis accelerates from the speed specified by JogSpeed to the target speed specified by TarSpeed (positive direction: the value > 0 or negative direction: the value < 0) and then the axis keeps moving at the target speed.

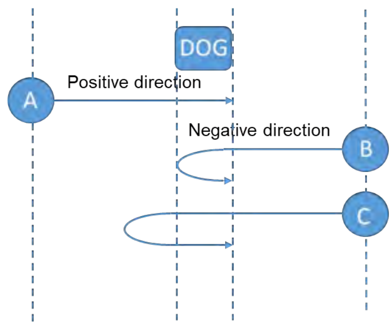
② After the DOG signal appears, the DOG signal is left in the following directions according to the selected mode.

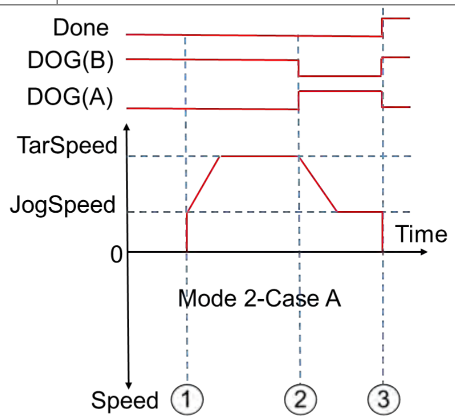
- In the previous direction as Case A shows, the axis decelerates to the JOG speed and then prepares for leaving the DOG signal.
- In the positive direction as CaseB/CaseC shows, the axis decelerates to the JOG speed and then stops. After that, it moves at the JOG speed in the opposite direction and prepares for moving away from the DOG signal.

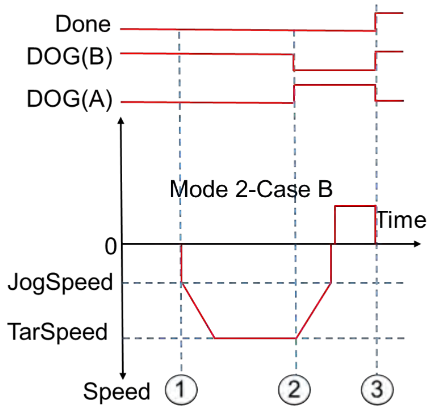
③ The DOG signal is moved away from and meanwhile the axis stops immediately.

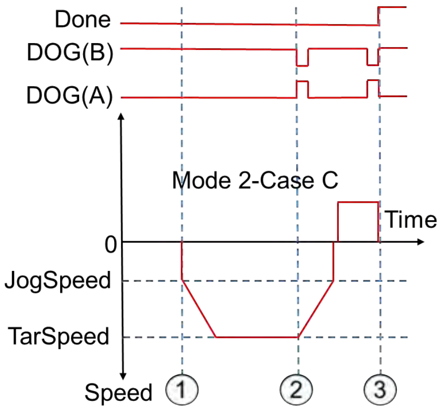
Mode 2：The axis starts to go toward the positive direction and then stops after leaving the DOG point position

Description of cases under mode 2	
Case A	The motion starts in the positive direction. As the DOG point is encountered, the axis starts to decrease its speed until the JOG speed is reached and then stops after leaving the DOG point.
Case B	The motion starts in the negative direction, the deceleration time is less than the duration for the DOG signal.
Case C	The motion starts in the negative direction; the deceleration time is greater than the duration for the DOG signal.









① The DPUZRN instruction is started and the axis accelerates from the speed specified by JogSpeed to the target speed specified by TarSpeed (positive direction: the value > 0 or negative direction: the value < 0) and then the motion continues at the target speed.

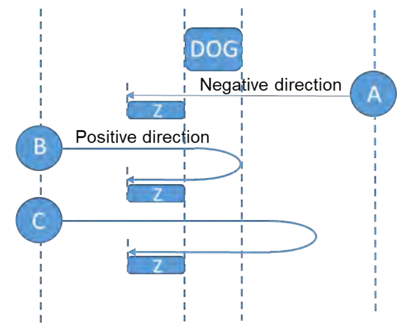
② After the DOG signal appears, the DOG signal is left in the following directions according to the selected mode.

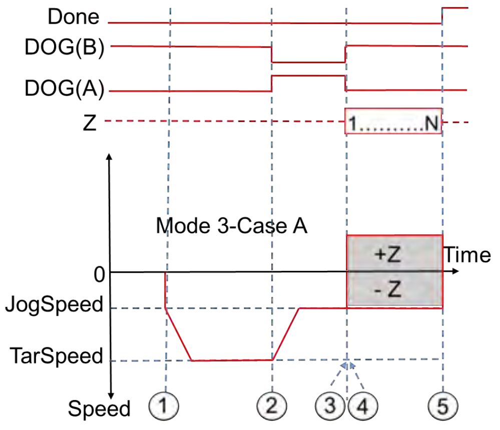
- In the previous direction as Case A shows, the axis decelerates to the JOG speed and then prepares for leaving the DOG signal.
- In the negative direction as CaseB/CaseC shows, the axis decelerates to the JOG speed and then stops. After that, it goes at the JOG speed in the opposite direction and prepares for leaving the DOG signal.

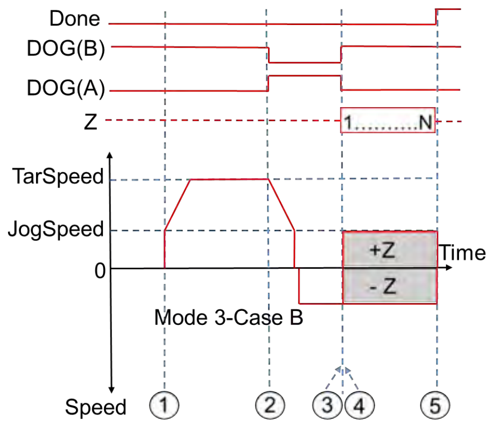
③ The DOG signal is left and meanwhile the axis stops immediately.

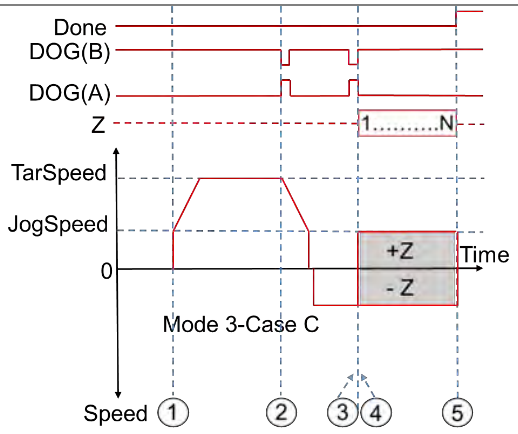
Mode 3：After Mode 1 is finished, seek the set number of Z phases.

Description of cases under mode 3	
Case A	The motion starts in the negative direction. As the DOG point is encountered, the axis starts to decelerate until the JOG speed is reached and then the search for a set number of Z phases begins before the axis stops immediately.
Case B	The motion starts in the positive direction; the deceleration time is less than the duration for the DOG signal.
Case C	The motion starts in the positive direction; the deceleration time is greater than the duration for the DOG signal.









① The DPUZRN instruction is started and the axis accelerates from the speed specified by JogSpeed to the target speed specified by TarSpeed (positive direction: the value > 0 or negative direction: the value < 0) and then the motion continues at the target speed.

② After the DOG signal appears, the DOG signal is left in the following directions according to the selected mode.

- ➢ In the previous direction as Case A shows, the axis decelerates to the JOG speed and then prepares for leaving the DOG signal.
- ➢ In the positive direction as CaseB/CaseC shows, the axis decelerates to the JOG speed and then the axis stops. After that, the axis goes at the JOG speed in the opposite direction and prepares for leaving the DOG signal.

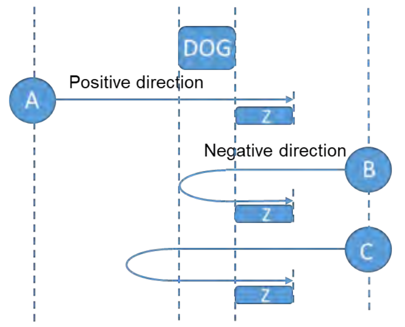
③ After the DOG signal is moved away from, the search for the set number of Z phase pulses begins (positive direction: the value > 0 or negative direction: the value < 0).

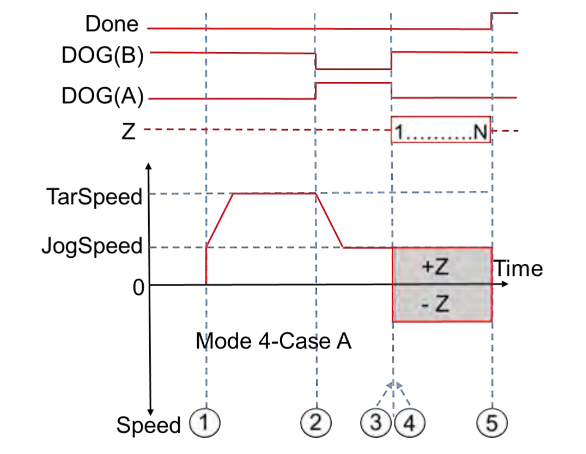
④ The first Z phase pulse is counted from.

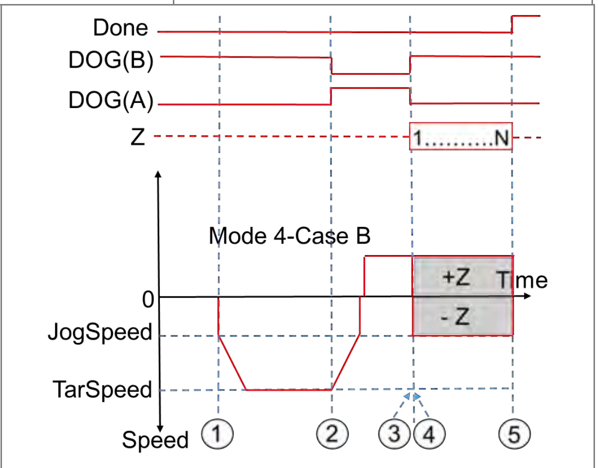
⑤ When counting to the N^{th} Z phase pulse, the axis stops immediately.

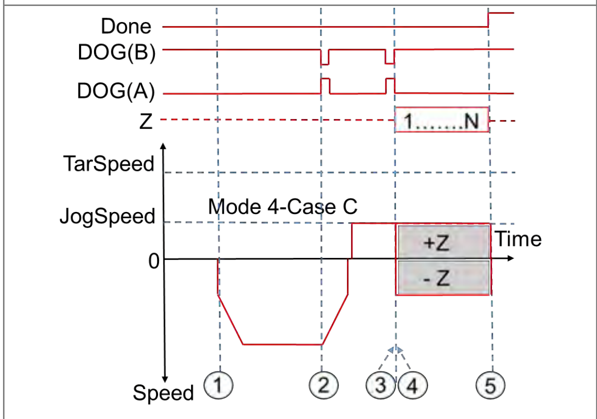
Mode 4：After Mode 2 is finished, seek the set number of Z phases.

Description of cases under mode 4	
Case A	The motion starts in the positive direction. As the DOG point is encountered, the axis starts to decelerate until the JOG speed is reached and then the search for a set number of Z phases begins before the axis stops immediately.
Case B	The motion starts in the negative direction; the deceleration time is less than the duration for the DOG signal.
Case C	The motion starts in the negative direction; the deceleration time is greater than duration for the DOG signal.









① The DPUZRN instruction is started and the axis accelerates from the speed specified by JogSpeed to the target speed specified by TarSpeed (positive direction: the value > 0 or negative direction: the value < 0) and then the motion goes on at the target speed.

② After the DOG signal appears, the DOG signal is left in the following directions according to the selected mode.

- In the previous direction as Case A shows, the axis decelerates to the JOG speed and then prepares for leaving the DOG signal.
- In the negative direction as CaseB/CaseC shows, the axis accelerates to the JOG speed and then the motion stops. After that, the axis continues at the JOG speed in the opposite direction and prepares for leaving the DOG signal.

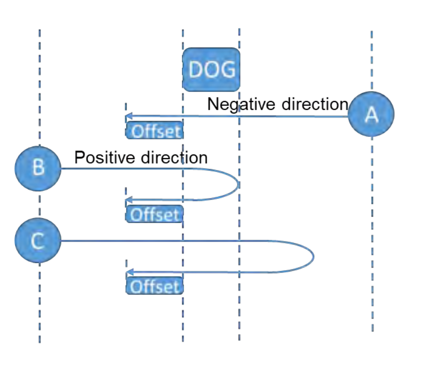
③ After the DOG signal is left, the search for a set number of Z phase pulses starts (positive direction: the value > 0 or negative direction: the value < 0)

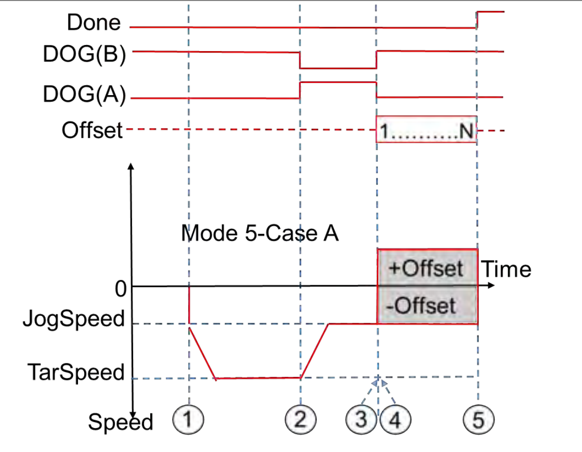
④ The first Z phase pulse is counted from.

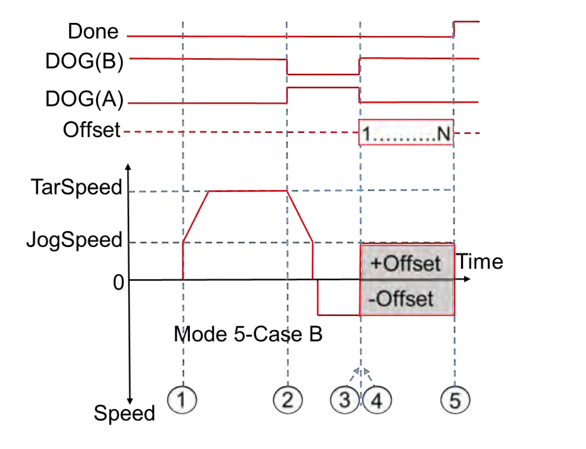
⑤ When counting to the Nth Z phase pulse is completed, the axis stops immediately.

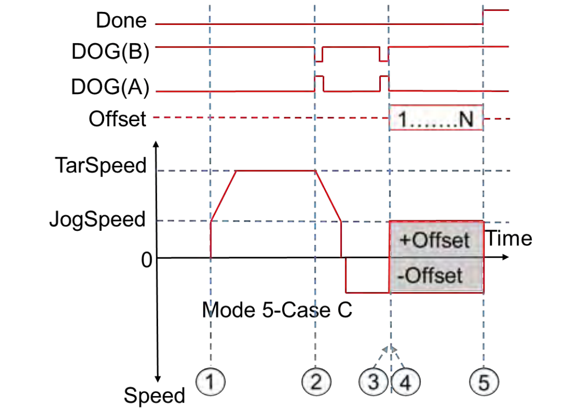
Mode 5：After Mode 1 is finished, output the offset position.

Description of cases under mode 5	
Case A	The motion starts in the negative direction. As the DOG point is encountered, the axis starts to decelerate until the JOG speed is reached, and then the offset position is output. When the offset outputs are completed, the axis stops right away.
Case B	The motion starts in the positive direction; the deceleration time is less than the duration for the DOG signal.
Case C	The motion starts in the positive direction; the deceleration time is greater than the duration for the DOG signal.









① The DPUZRN instruction is started and the axis accelerates from the speed specified by JogSpeed to the target speed specified by TarSpeed (positive direction: the value > 0 or negative direction: the value < 0) and then the motion goes on at the target speed.

② After the DOG signal appears, the DOG signal is left in the following directions according to the selected mode.

- In the previous direction as Case A shows, the axis decelerates to the JOG speed and then prepares for leaving the DOG signal.
- In the positive direction as CaseB/CaseC shows, the axis decelerates to the JOG speed and then stops. After that, the axis continues at the JOG speed in the opposite direction and prepares for leaving the DOG signal.

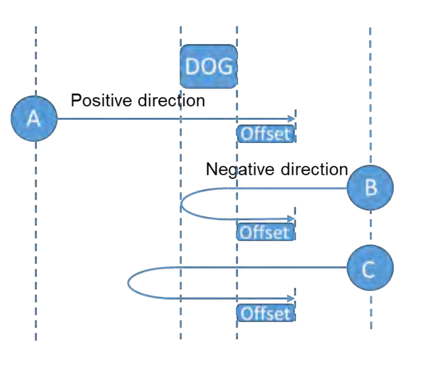
③ After the DOG signal is moved away from, the pulses of the number specified by Offset are output (positive direction: the value > 0 or negative direction: the value < 0).

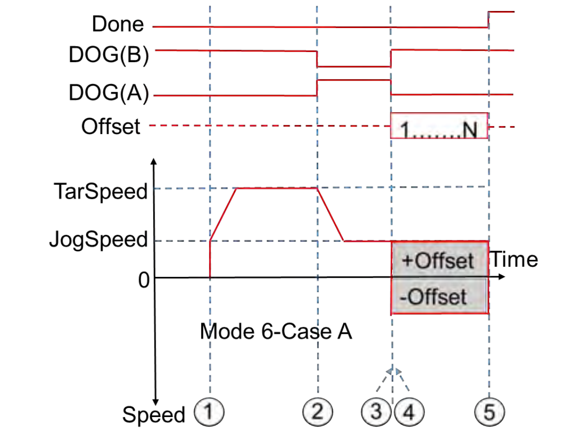
④ The first offset pulse is output.

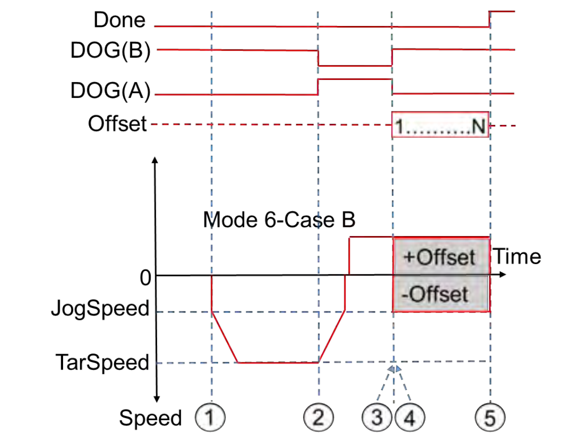
⑤ When the N^{th} offset pulse output is completed, the axis stops immediately.

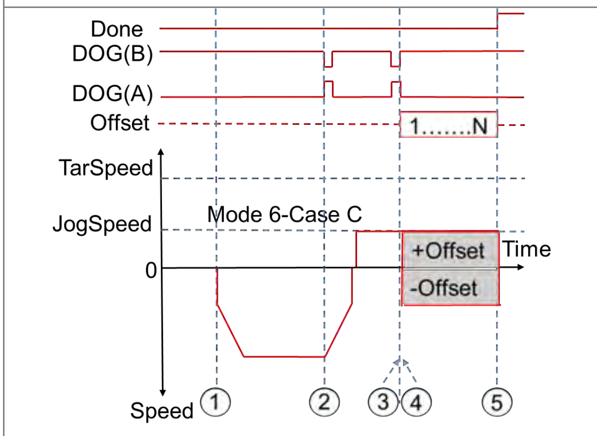
Mode 6：After Mode 2 is finished, output the offset position.

Description of cases under mode 6	
Case A	The motion starts in the positive direction. As the DOG point is encountered, the axis starts to decelerates until the JOG speed is reached and then the offset position is output. When the offset outputs are completed, the axis stops right away.
Case B	The motion starts in the negative direction; the deceleration time is less than the duration for the DOG signal.
Case C	The motion starts in the negative direction; the deceleration time is greater than the duration for the DOG signal.









① The DPUZRN instruction is started and the axis accelerates from the speed specified by JogSpeed to the target speed specified by TarSpeed (positive direction: the value > 0 or negative direction: the value < 0) and then the motion goes on at the target speed.

② After the DOG signal appears, the DOG signal is left in the following directions according to the selected mode.

➢ In the previous direction as Case A shows, the axis decelerates to the JOG speed and then prepares for leaving the DOG signal.

➢ In the negative direction as CaseB/CaseC shows, the axis decelerates to the JOG speed and then it stops. After that, the axis continues at the JOG speed in the opposite direction and prepares for leaving the DOG signal.

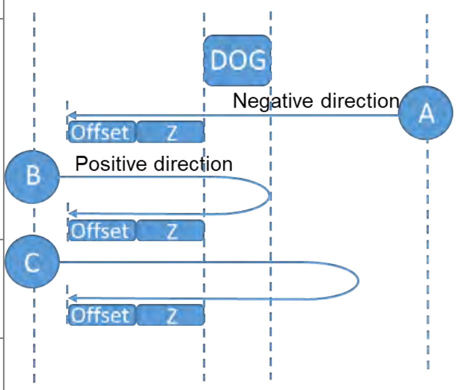
③ After the DOG signal is moved away from, the pulses of the number specified by Offset are output (positive direction: the value > 0 or negative direction: the value < 0).

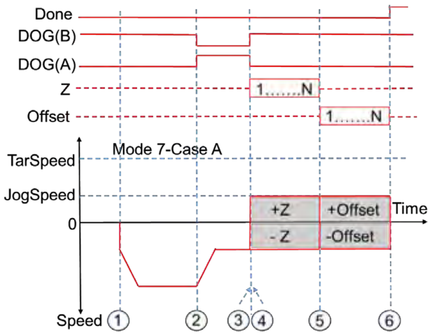
④ The first offset pulse is output.

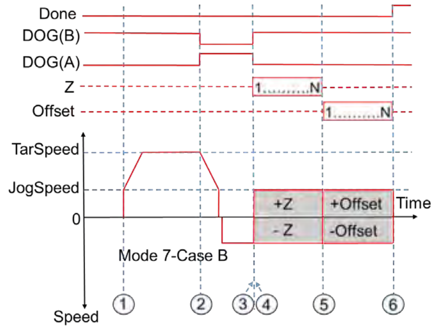
⑤ When the N[th] offset pulse output is completed, the axis stops immediately.

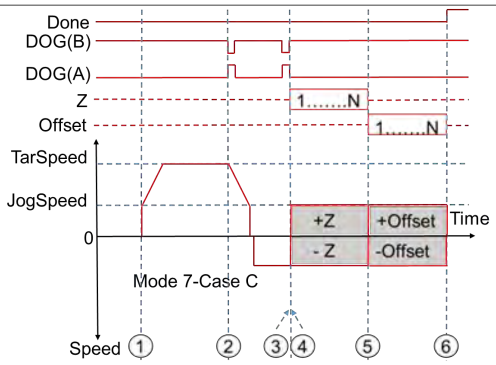
Mode 7：After Mode 1 is finished, seek the set number of Z phases and then output the offset position.

Description of cases under mode 7	
Case A	The motion starts in the negative direction. As the DOG point is encountered, the axis starts to decelerates until the JOG speed is reached and then the search for Z phases starts. After the last Z phase is counted to, the offset position output begins. When offset outputs are completed, the axis stops right away.
Case B	The motion starts in the positive direction; the deceleration time is less than the duration for the DOG signal.
Case C	The motion starts in the positive direction; the deceleration time is greater than the duration for the DOG signal.









① The DPUZRN instruction is started and the motion accelerates from the speed specified by JogSpeed to the target speed specified by TarSpeed (positive direction: the value > 0 or negative direction: the value < 0) and then the motion goes on at the target speed.

② After the DOG signal appears, the DOG signal is left in the following directions according to the selected mode.

- In the previous direction as Case A shows, the axis decelerates to the JOG speed and then prepares for leaving the DOG signal.
- In the positive direction as CaseB/CaseC shows, the axis decelerates to the JOG speed and then stops. After that, the axis continues at the JOG speed in the opposite direction and prepares for moving away from the DOG signal.

③ After the DOG signal is moved away from, the search for a set number of Z phases (positive direction: the value > 0 or negative direction: the value < 0).

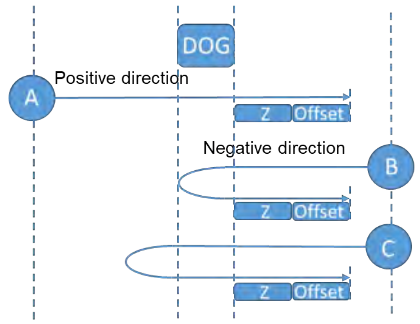
④ The first Z phase pulse is counted from.

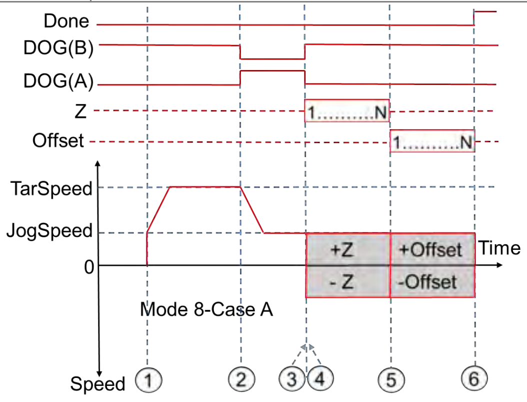
⑤ When counting to the Nth Z phase pulse, the first offset pulse output starts (positive direction: the value > 0 or negative direction: the value < 0).

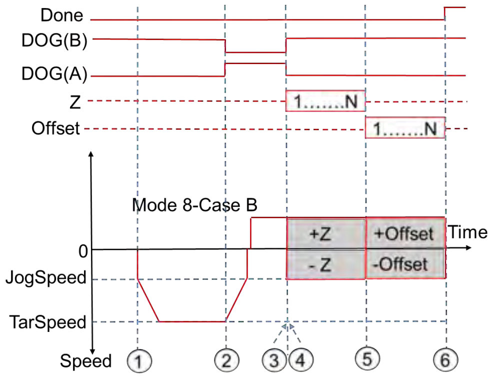
⑥ When the Nth offset pulse output is completed, the axis stops immediately.

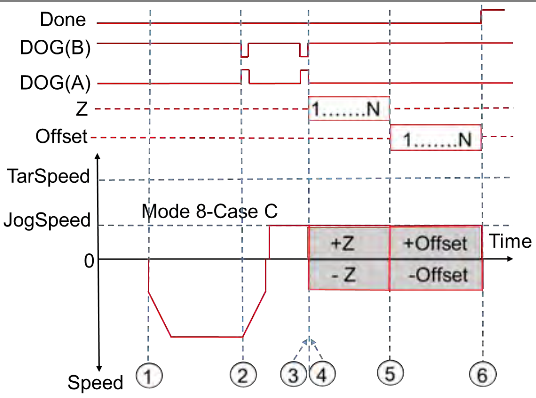
Mode 8：After Mode 2 is finished, seek the set number of Z phases and then output the offset position.

Description of cases under mode 2	
Case A	The motion starts in the positive direction. As the DOG point is encountered, the axis starts to decelerates until the JOG speed is reached and then the search for Z phases starts. After the last Z phase is counted to, the offset position output begins. When offset outputs are completed, the axis stops right away.
Case B	The motion starts in the negative direction; the deceleration time is less than the duration for the DOG signal.
Case C	The motion starts in the negative direction; the deceleration time is greater than the duration for the DOG signal.









① The DPUZRN instruction is started and the axis accelerates from the speed specified by JogSpeed to the target speed specified by TarSpeed (positive direction: the value > 0 or negative direction: the value < 0) and then the motion goes on at the target speed.

② After the DOG signal appears, the DOG signal is left in the following directions according to the selected mode.

- In the previous direction as Case A shows, the axis decelerates to the JOG speed and then prepares for leaving the DOG signal.
- In the negative direction as CaseB/CaseC shows, the axis decelerates to the JOG speed and then stops. After that, the axis continues at the JOG speed in the opposite direction and prepares for moving away from the DOG signal.

③ After the DOG signal is moved away from, the search for a set number of Z phases (positive direction: the value > 0 or negative direction: the value < 0).

④ The first Z phase pulse is counted from.

⑤ When counting to the N^{th} Z phase pulse, the first offset pulse output starts (positive direction: the value > 0 or negative direction: the value < 0).
⑥ When the N^{th} offset pulse output is completed, the axis stops immediately.

Programming Example: Refer to the description of DPUDRI instruction (API 347) for more information.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
350	D	PUJOG	Module~ErrCode	PU module jog output				

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
Module					*	*							*			DPUJOG: 25 steps
Axis					*	*							*			(To be used in ISPSoft)
JogSpeed					*	*							*			
Busy		*	*	*												
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Axis:** Output axis number **JogSpeed**: Jog output frequency

Busy: Output in execution **Error**: Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. When En setting is set to ON, this instruction would be effective. Once the setting changes to OFF, Output pulse would be terminated immediately. WPLSoft does NOT support this instruction.

2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.

3. **Axis** sets the output axis number for the specified PU module. The setting values 1~2 represent the axis1~axis2 output of the specified PU module respectively. If the PU module has no corresponding axis number for output, the error flag **Error** will change to ON.

4. **JogSpeed** sets the jog output frequency. The setting value is a signed 32 bit value within the range of -200,000 (-200K) ~200,000 (200K) (Hz). When the value is greater than 0, the output will go in the positive direction (and the direction output point is off). When the value is less than 0, the output will go in the negative direction (and the direction output point is on). When the value is 0, the output will stop.

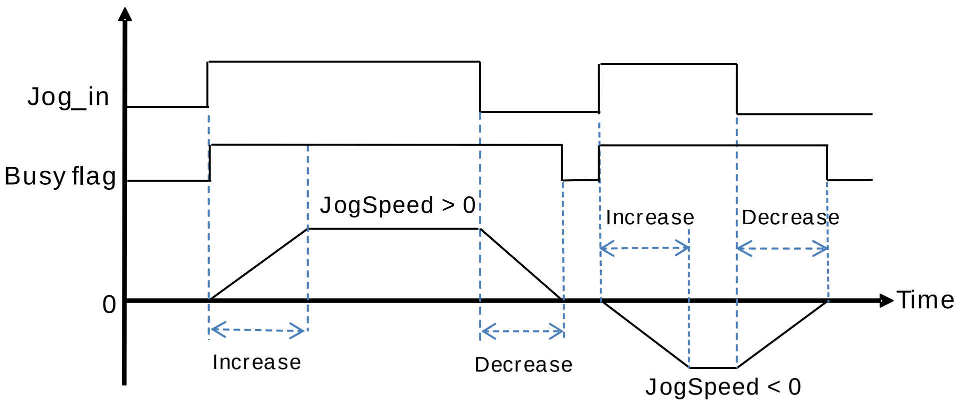
5. The instruction can be used for the speed change. While the instruction is being executed, you can change the value of **TarSpeed** so as to change the output speed. When the setting value

exceeds the maximum frequency, the instruction would be executed at the maximum frequency. But changing the speed would not change the direction. To change the direction, set the value of **TarSpeed** to 0 first and then modify the target speed.

6. The instruction can be used with the software and hardware limit points. When the limits are triggered, the output stops immediately and the **Error** flag changes to ON.
7. If any error occurs as the instruction is in process of the output, the **Error** flag changes to ON. Refer to the error codes that **ErrCode** shows for the trouble shooting.
8. The error codes that **ErrCode** shows are listed in the following table.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.
16#1403	There is no such output axis number in the PU module.
16#1405	The output axis specified by the PU module is outputting data. It is not allowed to specify the output repeatedly.
16#1406	PU module stops Output pulse when the positive limit is reached.
16#1407	PU module stops Output pulse when the negative limit is reached.

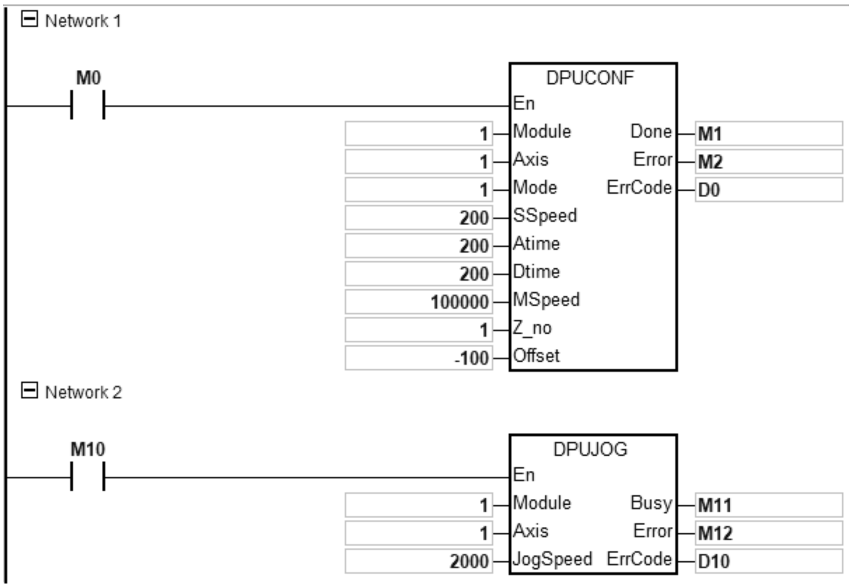
9. See the output timing diagram as below. (Jog_in is the switch to start the instruction and the Busy flag is the **Busy** flag.)



10. After the PUJOG instruction is disabled and the **Busy** flag is off, other output control can be carried out.

Programming Example:

1. When M0 is ON, the DPUCONF instruction for axis 1 is executed to modify the parameters by setting **Mode** to 1 (Pulse Y0 + direction Y1), **SSpeed** to 200Hz, **Atime** to 200ms, **Dtime** to 200ms and **MSpeed** to 100kHz. After the output of **Done** is completed, M1 is ON.
2. When M10 is ON, the DPUJOG instruction for axis 1 starts to perform jog outputs. The pulses are output from Y0 at the frequency of 2KHz. If Y1 is OFF, the direction is positive. And M11 is ON during the instruction execution.



API	Mnemonic		Operands	Function	Controllers			
351	D	PUMPG	Module~ErrCode	PU module MPG output	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
	Bit Devices				Word devices											
Module					*	*							*			DPUMPG: 41 steps (To be used in ISPSoft)
Axis					*	*							*			
InMode					*	*							*			
InPulse													*			
InSpeed													*			
Rate													*			
OPulse													*			
OSpeed													*			
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Axis**: Output axis number **InMode**: Encoder input mode and HYPERLINK "D:\\Manuals\\DVP\\ES2 Operation Manual-Programming\\Editing\\frequency"D:\Manuals\DVP\ES2 Operation Manual-Programming\Editing\frequency HYPERLINK "D:\\Manuals\\DVP\\ES2 Operation Manual-Programming\\Editing\\multiplication"D:\Manuals\DVP\ES2 Operation Manual-Programming\Editing\multiplication for counting **InPulse**: Number of pulses which have been input **InSpeed**: Detected input frequency **Rate**: Input/output rate (floating point number)

OPulse: Number of pulses which have been output **OSpeed**: Frequency at which pulses are being output **Error**: Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. When En setting is set to ON, this instruction would be effective. Once the setting changes to OFF, Output pulse would be terminated immediately. WPLSoft does NOT support this instruction.

2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.

3. **Axis** sets the output axis number for the specified PU module. The setting values 1~2

represent the axis1~axis2 output of the specified PU module respectively. If the PU module has no corresponding axis number for output, the error flag **Error** will change to ON.

4. **InMode** sets the input mode of the encoder source and the frequency HYPERLINK "D:\\Manuals\\DVP\\ES2 Operation Manual-Programming\\Editing\\multiplication"[D:\Manuals\DVP\ES2 Operation Manual-Programming\Editing\multiplication](D:\Manuals\DVP\ES2 Operation Manual-Programming\Editing\multiplication) for counting.

   See the explanation of **InMode** value in the following table.

Value	Input Modes Input mode; set as the following values, otherwise the module will use the defaults to run.
16#0000	Fourfold frequency A/B phase input (default). Phase A leads phase B, indicating counting in the positive direction. Phase B leads phase A, indicating counting in the negative direction.
16#0001	Onefold frequency A/B phase input. Phase A leads phase B, indicating counting in the positive direction. Phase B leads phase A, indicating counting in the negative direction.
16#0002	Twofold frequency A/B phase input. Phase A leads phase B, indicating counting in the positive direction. Phase B leads phase A, indicating counting in the negative direction.
16#0003	Reserved
16#0004	Fourfold frequency A/B phase input Phase A leads phase B, indicating counting in the positive direction. Phase B leads phase A, indicating counting in the negative direction.
16#0005	Pulse + directional input (A+/A-: pulse input; B+/B-: directional input) Phase B ON: counting in the negative direction Phase B OFF: counting in the positive direction Phase A: counting is started by rising-edge trigger.
16#0006	Pulse + directional input (A+/A-: pulse input; B+/B-: directional input) Phase B ON: counting in the positive direction Phase B OFF: counting in the negative direction Phase A: counting is started by rising-edge trigger.
16#0007	Single phase pulse input (A+/A-: pulse input) Phase A: counting is started by rising-edge trigger.
Others	Reserved

5. **InPulse** displays the number of already input pulses, which is a signed 32-bit value. Every time the instruction is started, the PU module will automatically clear the value to 0 and then starts counting.
6. **InSpeed** displays the already detected input frequency which is a 32-bit value. The basic time for the frequency detection is 20ms. Therefore, the detected input frequency is 0 if there is no counting value within 20ms. If there is a counting value within 20ms, the output starts at the minimum frequency of 50Hz. Even if **OSpeed** value is lower than 50Hz through the

Rate-value-based conversion, the output is still conducted at 50Hz.

7. **Rate** is the input / output rate and the value is a floating point number. The number of actual output pulses and frequency are respectively equal to the input pulse number and frequency multiplied by the rate value.

   For example: The input frequency is 100Hz and rate is 0.5. So the output frequency is 100x0.5=50Hz. If the maximum output frequency after conversion exceeds 100KHz, the output frequency is limited to 100KHz.

   Note: The long-time maximum frequency output may lead to the fact that as the MPG has stopped running, the number of outputs is still increased and the output need keep going until it is complete.

8. **OPulse** shows the number of pulses which have been output. **OSpeed** displays the frequency at which the output is being conducted. They are signed 32-bit values.

9. When the **DPUMPG** instruction is disabled, check the frequency at which the output is being conducted and see if it has reached 0. If the instruction is disabled before the frequency reaches 0, the PU module will stop the output immediately and the output of the pulses which are counted based on the conversion rate will not continue any more.

10. The error codes that **ErrCode** shows are listed in the following table.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.
16#1403	There is no such output axis number in the PU module.
16#1405	The output axis specified by the PU module is outputting data. It is not allowed to specify the output repeatedly.
16#1406	PU module stops Output pulse when the positive limit is reached.
16#1407	PU module stops Output pulse when the negative limit is reached.

11. When the DPUMPG instruction is enabled or disabled, the PLC will have to notify the module to enable or disable the high-speed counter function. Thus the instruction can not be used with API352 DPUCNT together. Otherwise it may occur that the two instructions enable or disable the counting of the module with each other.

API	Mnemonic		Operands	Function	Controllers			
					ES2/EX2	SS2	SA2 SE	SX2
352	D	PUDCNT	Module~ErrCode	High-speed counter function of PU module				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DPUCNT: 33 steps (To be used in ISPSoft)
Module					*	*							*			
InMode					*	*							*			
Period					*	*							*			
ZeroS	*	*	*	*									*			
InPulse													*			
InSpeed													*			
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **InMode**: Encoder input mode and frequency multiplication for counting **Period**: Period time for capturing the frequency **ZeroS**: Clear the counter to 0 **InPulse**: Number of pulses which have been input **InSpeed**: Number of pulses per cycle **Error**: Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. When En setting is set to ON, this instruction would be effective. Once the setting changes to OFF, Output pulse would be terminated immediately. WPLSoft does NOT support this instruction.

2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.

3. **InMode** sets the input mode of the encoder source and the frequency HYPERLINK "D:\\Manuals\\DVP\\ES2 Operation Manual-Programming\\Editing\\multiplication"D:\Manuals\DVP\ES2 Operation Manual-Programming\Editing\multiplication for counting.

   See the explanation of **InMode** value in the following table.

   Note: Phase A leads phase B, indicating counting in the positive direction.

Phase B leads phase A, indicating counting in the negative direction

Value	**Input Modes** Input mode; set as the following values, otherwise the module will use the defaults to run.
16#0000	Reserved
16#0001	Onefold frequency A/B phase input
16#0002	Twofold frequency A/B phase input
16#0003	Reserved
16#0004	Fourfold frequency A/B phase input (default)
16#0005	Pulse + directional input (A+/A-: pulse input; B+/B-: directional input) Phase B ON: counting in the negative direction Phase B OFF: counting in the positive direction Phase A: counting is started by rising-edge trigger.
16#0006	Pulse + directional input (A+/A-: pulse input; B+/B-: directional input) Phase B ON: counting in the positive direction Phase B OFF: counting in the negative direction Phase A: counting is started by rising-edge trigger.
16#0007	Single phase pulse input (A+/A-: pulse input) Phase A: counting is started by rising-edge trigger.
Others	Reserved

4. **Period** is the setting value of a cycle time for capturing the frequency within the range of 10ms ~ 1000ms. If the setting value exceeds the range, the maximum value or minimum value will be automatically taken as the setting value by the PLC.

5. **ZeroS** clears the present output position to 0. If the present axis position is to be cleared to 0, set **ZeroS** from OFF to ON when the instruction is started.

6. **InPulse** is the number of already input pulses, which is a signed 32-bit value. The counting value is a latched value. If the value need be cleared to 0, just set **ZeroS** from Off to ON while the instruction is running.

7. **InSpeed** displays the counting value for every **Period** time, which is a signed 32-bit value. If you need convert it into the value with the unit of Hz, use the calculation formula for conversion by yourself.

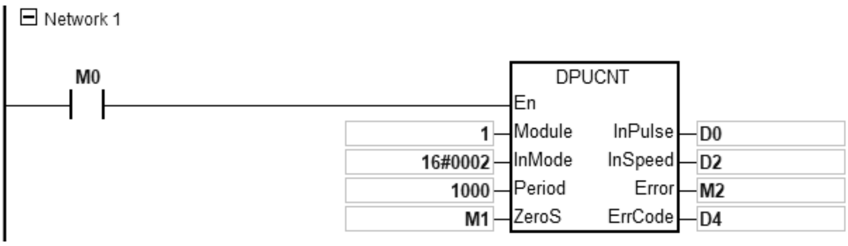
8. The error codes that **ErrCode** shows are listed in the following table.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.

9. When the DPUCNT instruction is enabled or disabled, the PLC will have to notify the module to enable or disable the high-speed counter function. Thus the instruction can not be used with API1409 DPUMPG together. Otherwise it may occur that the two instructions enable or disable the counting of the module with each other.

Programming Example

1. When M0 is ON, the DPUCNT instruction is executed and **InMode** is set to "twofold frequency A/B phase input".
2. When M1 is ON, the counted number of pulses on axis 1 in **InPulse** is cleared.
3. When the input number of pulses is 100 and frequency is 10Hz, **InPulse** and **InSpeed** show 200 pulses and 20Hz respectively.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
353		PUX	P	Module~ErrCode	Setting PU module input point mode				

Type / OP	Bit Devices X	Y	M	S	Word devices K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	Program Steps
Module					*	*							*			PUX, PUXP: 15 steps (To be used in ISPSoft)
Xno					*	*							*			
Xmode					*	*							*			
Xfilter					*	*							*			
Done		*	*	*												
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Xno**: Input point number **Xmode:** Input point mode

Xfilter: Input point filter time **Done**: Completion flag **Error**: Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. When En setting is set to ON, this instruction would be effective. Once the setting changes to OFF, Output pulse would be terminated immediately. WPLSoft does NOT support this instruction.

2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.

3. **Xno** sets the input point number for the PU module with one of the input values 0~4 which respectively represent the input points X0~X4. If there is no correspoing input point in the PU module, the error flag **Error** will change to ON. The value in **Xno** and the corresponding input point are listed in the following table.

Setting in Xno	0	1	2	3	4	Others
DVP02PU-E2	X0	X1	X2	X3	X4	No input point

4. **XMode** selects an input mode for input points. Setting values are explained in the following table:

Setting in XMode	DVP02PU-E2
0	General input (Default)
1	Axis 1, Z phase, Rising-edge triggered
2	Axis 2, Z phase, Rising-edge triggered
3	Axis 1, Z phase, Falling-edge triggered
4	Axis 2, Z phase, Falling-edge triggered
5	Axis 1, DOG, Rising-edge triggered
6	Axis 2, DOG, Rising-edge triggered
7	Axis 1, DOG, Falling-edge triggered
8	Axis 2, DOG, Falling-edge triggered
9	Axis 1, LSN, Rising-edge triggered
10	Axis 2, LSN, Rising -edge triggered
11	Axis 1, LSN, Falling-edge triggered
12	Axis 2, LSN, Falling-edge triggered
13	Axis 1, LSP, Rising-edge triggered
14	Axis 2, LSP, Rising-edge triggered
15	Axis 1, LSP, Falling-edge triggered
16	Axis 2, LSP, Falling-edge triggered
Other	Automatically switch to mode 0 (default)

5. **Xfilter** is explained in the following table. The value in **Xfilter** is the default value if the setting is out of the allowed range.

Parameter	Function	Range	Default
Xfilter	Input point filter time	0 ~ 25 (unit:ms)	10

6. **Done**, an output of the specified PU module has been set as the completion flag. When **Done** is On, it indicates that the parameter setting is successful. You can continue to perform positioning output based on the state of the completion flag (ON). The clearing of the **Done** flag need be conducted by manual. The **Done** flag changes to ON only when the setting is completed.

7. **Error**, an output of the specified PU module is a parameter error flag. Most parameter ranges are filtered automatically by the PLC. Thus if the error flag is ON, it means that there is no

specified PU module or the PU module number is wrong or the output axis number is incorrect.

8. The instruction is a pulse instruction. Even if the A contact is adopted as the condition contact, PU module parameters are also set only when the instruction is started. Therefore, if a parameter value is to be updated, restart the instruction to make the parameter set again.

9. Since the set parameters are delivered through the module communication command, confirm the state of the output **Done** or **Error** before a parameter value is modified and then proceed with relevant operations.

10. For the state of PU module input points, check the data exchange function of the special extension module, SM228. Refer to Section 2.2.16 Additional Remarks on Special Auxiliary Relays and Special Data Registers in DVP-ES3 Series Programming Manual for details on SM228.

11. The error codes that **ErrCode** shows are listed in the following table.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.

Programming Example: Refer to the description of DPUDRI instruction (API 347) for more information.

API	Mnemonic			Operands	Function	Controllers			
354	D	PULS	P	Module~ErrCode	Setting PU module software limits	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DPULS, DPULSP: 29 steps (To be used in ISPSoft)
Module					*	*							*			
Axis					*	*							*			
LSN					*	*							*			
LSP					*	*							*			
Done		*	*	*												
Error		*	*	*												
ErrCode													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

Module: Module number **Axis**: Output axis number **LSN**: Software negative limit

LSP: Software positive limit **Done**: Completion flag **Error**: Error flag **ErrCode**: Error code

Explanations:

1. This instruction is available for ES2/EX2 (FW V4.02 or later) and ES2-E (FW V1.48 or later). For ISPSoft, we recommend using software version 3.16 and above. The timing to set this instruction is when En changes from OFF to ON. WPLSoft does NOT support this instruction.

2. **Module** sets the serial number of modules at the right of the PLC. The first one is number 0, the second one is number 1 and so on. Whatever modules at the right of the PLC must be numbered. The maximum number is 8. The instruction is exclusive to the PU modules at the right of the PLC. If the specified module is not a PU module, the error flag **Error** will change to ON.

3. **Axis** sets the output axis number for the specified PU module. The setting values 1~2 represent the axis1~axis2 of the specified PU module respectively. If the PU module has no corresponding axis number for output, the error flag **Error** will change to ON.
   See the following combination of axis numbers and corresponding output points of PU modules.

PU module name	Axis 1	Axis 2
DVP02PU-E2	Y0 / Y1	Y2 / Y3

4. See the explanation of **LSN and LSP** and setting values in the following table. If the setting value is outside the range, the instruction will automatically be executed at the minimum or maximum value.

Parameter	Function	Range	Default	Remark
LSN	Software negative limit	-2,147,483,648 ~ +2,147,483,647	0	Inactive when both are set to 0
LSP	Software positive limit	-2,147,483,648 ~ +2,147,483,647	0	

5. **Done**, an output of the specified PU module has been set as the completion flag. When **Done** is On, it indicates that the parameter setting is successful. You can continue to perform positioning output based on the state of the completion flag (ON). The clearing of the **Done** flag need be conducted by manual. The **Done** flag changes to ON only when the setting is completed.

6. **Error**, an output of the specified PU module is a parameter error flag. Most parameter ranges are filtered automatically by the PLC. Thus if the error flag is ON, it means that there is no specified PU module or the PU module number is wrong or the output axis number is incorrect.

7. The instruction is a pulse instruction. Even if the A contact is adopted as the condition contact, PU module parameters are also set only when the instruction is started. Therefore, if a parameter value is to be updated, restart the instruction to make the parameter set again.

8. Since the set parameters are delivered through the module communication command, confirm the state of the output **Done** or **Error** before a parameter value is modified and then proceed with relevant operations.

9. The error codes that **ErrCode** shows are listed in the following table.

Error code	Description
16#1400	The module does not support the function.
16#1402	There is no response from the module; communication timeout occurs.

Programming Example: Refer to the description of DPUDRI instruction (API 347) for more information.

Communications

4

This chapter introduces information regarding the communications ports of the PLC. Through this chapter, the user can obtain a full understanding about PLC communication ports.

Table of Contents

4.1 Communication Ports

DVP-ES2/EX2/SA2/SE/SX2 offers 3 communication ports (COM1~COM3), and DVP-SS2/EC5 offers 2 COM ports (COM1~COM2). COM ports of the above models support DELTA Q-link communication format on HMI. Refresh rate of HMI can be increased by this function.

COM1: RS-232 communication port. COM1 can be used as master or slave and is the major COM port for PLC programming. (It is not applicable to DVP-SE.)

COM2: RS-485 communication port. COM2 can be used as master or slave. ES2-C V40 or later does NOT support 600 bmp.

COM3 (ES2/EX2/SA2/SE): RS-485 communication port. COM3 can be used as master or slave. (For DVP-ES2-C, COM3 is the CANopen port.)

COM3 (SX2): Conversion from the USB port to RS-232 port. COM3 can be used as slave only. The 3 COM ports on the models mentioned above support Modbus ASCII or RTU communication format.

USB (COM1) (SE): USB communication port. It only can be used as a slave. The communication mode and format can not be modified.

Communication Format:

COM port / Parameter	RS-232 (COM1)	RS-485 (COM2)	RS-485 (COM3)	RS-485 (SX2 COM3)
Baud rate	60~115200 bps	60~921000 bps		
Data length	7~8bits			
Parity	Even / Odd / None parity check			
Length of stop bit	1~2 bits			
Register for Setting	D1036	D1120	D1109	
Retain communication format	M1138	M1120	M1136	
ASCII mode	Available for both Master/Slave			Available for Slave
RTU mode	Available for both Master/Slave			Available for Slave
ASCII/RTU mode selection	M1139	M1143	M1320	
Communication address of Slave	D1121		D1255	
Data length for access (ASCII/RTU)	100 registers			

Default communication settings for all COM ports:

– Modbus ASCII
– 7 data bits
– 1 stop bit
– Odd parity
– Baud rate: 9600

4.2 Communication Protocol ASCII mode

Communication Data Structure

9600 (Baud rate), 7 (data bits), Even (Parity), 1 (Start bit), 1 (Stop bit)

Field name	Content	Explanation
Start bit	STX	Start bit ‘:’ (3AH)
Communication address	ADR 1	Address consists of 2 ASCII codes
	ADR 0	
Command code	CMD 1	Command code consists of 2 ASCII codes
	CMD 0	
Data	DATA (0)	Data content consist of 2n ASCII codes, n≤205
	DATA (1)	
	……….	
	DATA (n-1)	
LRC checksum	LRC CHK 1	LRC checksum consists of 2 ASCII codes
	LRC CHK 0	
Stop bit	END1	Stop bit consists of 2 ASCII codes END1 = CR (0DH), END0 = LF (0AH)
	END0	

Corresponding table for Hexadecimal value and ASCII codes

ASCII	“0“	“1“	“2“	“3“	“4“	“5“	“6“	“7“
Hex	30H	31H	32H	33H	34H	35H	36H	37H
ASCII	“8“	“9“	“A“	“B“	“C“	“D“	“E“	“F“
Hex	38H	39H	41H	42H	43H	44H	45H	46H

4.2.1 ADR (Communication Address)

Valid communication addresses are in the range of 0~254. Communication address equals to 0 means broadcast to all PLCs. PLC will not respond to a broadcast message. PLC will reply a normal message to the master device when communication address is not 0.

Example, ASCII codes for communication address 16 in Decimal. (16 in Decimal = 10 in Hex)
(ADR 1, ADR 0)=’1’,’0’⇨’1’=31H, ‘0’ = 30H

4.2.2 CMD (Command code) and DATA

The content of access data depends on the command code.

Available setting for command code:

CMD(Hex)	Explanation	Device
01 (01 H)	Read status of contact	S, Y, M, T, C
02 (02 H)	Read status of contact	S, X, Y, M,T, C
03 (03 H)	Read content of register	T, C, D
05 (05 H)	Force ON/OFF single contact	S, Y, M, T, C
06 (06 H)	Set content of single register	T, C, D
15 (0F H)	Force ON/OFF multiple contacts	S, Y, M, T, C
16 (10 H)	Set content of multiple registers	T, C, D
17 (11 H)	Retrieve information of Slave	None
23 (17 H)	Simultaneous data read/write in a polling of EASY PLC LINK	None

Example: Read devices T20~T27 (address: H0614~H61B) from Slave ID#01(station number)

PC→PLC

“: 01 03 06 14 00 08 DA CR LF”

Sent massage:

Field name	ASCII	Hex
STX	:	3A
Slave Address	01	30 31
Command code	03	30 33
Starting Address High	06	30 36
Starting Address Low	14	31 34
Number of Points High	00	30 30
Number of Points Low	08	30 38
LRC checksum	DA	44 41
END	CR LF	0D 0A

PLC→PC

“: 01 03 10 00 01 00 02 00 03 00 04 00 05 00 06 00 07 00 08 C8 CR LF”

Responded massage:

Field name	ASCII	Hex
STX	:	3A
Slave Address	01	30 31
Command code	03	30 33
Bytes Count	10	31 30
Data Hi (T20)	00	30 30
Data Lo (T20)	01	30 31
Data Hi (T21)	00	30 30

Field name	ASCII	Hex
Data Lo (T21)	02	30 32
Data Hi (T22)	00	30 30
Data Lo (T22)	03	30 33
Data Hi (T23)	00	30 30
Data Lo (T23)	04	30 34
Data Hi (T24)	00	30 30
Data Lo (T24)	05	30 35
Data Hi (T25)	00	30 30
Data Lo (T25)	06	30 36
Data Hi (T26)	00	30 30
Data Lo (T26)	07	30 37
Data Hi (T27)	00	30 30
Data Lo (T27)	08	30 38
Check sum(LRC)	C8	43 38
END	CR LF	0D 0A

4.2.3 LRC CHK (checksum)

LRC (Longitudinal Redundancy Check) is calculated by summing up the Hex values from ADR1 to last data character then finding the 2's-complement negation of the sum.

Example: Read the content of register at address 0401H. 01H+03H+04H+01H+00+01H = 0AH.
The 2's-complement of 0AH: F6H

Field name	ASCII	Hex
STX	:	3A
Slave Address	**01**	30 31
Command code	**03**	30 33
Starting data address Hi	**04**	30 34
Starting data address Lo	**01**	30 31
Number of data Hi	**00**	30 30
Number of data Lo	**01**	30 31
LRC checksum	F6	46 36
END	CR LF	0D 0A

Exception response:

The PLC is expected to return a normal response after receiving command messages from the master device. The following table depicts the conditions that either a no response or an error response is replied to the master device.

1. The PLC did not receive a valid message due to a communication error; thus the PLC has no response. The master device will eventually process a timeout condition.

2. The PLC receives a valid message without a communication error, but cannot accommodate it, an

exception response will return to the master device. In the exception response, the most significant bit of the original command code is set to 1, and an exception code explaining the condition that caused the exception is returned.

An example of exception response of command code 01H and exception 02H:
Sent message:

Field Name	ASCII	Hex
STX	:	3A
Slave Address	01	30 31
Command code	01	30 31
Starting Address Hi	04	30 34
Starting Address Lo	00	30 30
Number of Points Hi	00	30 30
Number of Points Lo	10	31 30
Error Check (LRC)	EA	45 41
END	CR LF	0D 0A

Feedback message:

Field Name	ASCII	Hex
STX	:	3A
Slave Address	01	30 31
Function	81	38 31
Exception Code	02	30 32
Error Check (LRC)	7C	37 43
END	CR LF	0D 0A

Exception code:	Explanation:
01	Illegal command code: The command code received in the command message is invalid for PLC.
02	Illegal device address: The device address received in the command message is invalid for PLC.
03	Illegal device content: The data received in the command message is invalid for PLC.
07	1. Checksum Error - Check if the checksum is correct 2. Illegal command messages - The command message is too short. - Length command message is out of range.

4.3 Communication Protocol RTU mode

Communication Data Structure

9600 (Baud rate), 8 (data bits), EVEN (Parity), 1 (Start bit), 1 (Stop bit)

START	No data input ≥ 10 ms
Address	Communication Address: the 8-bit binary address
Command code	Command Code: the 8-bit binary address
DATA (n-1)	Data Contents: n × 8-bit BIN data, n≦202
…….	
DATA 0	
CRC CHK Low	CRC Checksum: The 16-bit CRC checksum is composed of 2 8-bit binary codes
CRC CHK High	
END	No data input ≥ 10 ms

4.3.1 Address (Communication Address)

Valid communication addresses are in the range of 0~254. Communication address equals to 0 means broadcast to all PLCs. PLC will not respond to a broadcast message. PLC will reply a normal message to the master device when communication address is not 0.

Example, communication address should be set to 10 (Hex) when communicating with a PLC with address 16 (Dec) (16 in Decimal = 10 in Hex)

4.3.2 CMD (Command code) and DATA

The content of access data depends on the command code. For descriptions of available command codes, please refer to **4.2.2** in this chapter.

Example: read consecutive 8 words from address 0614H~H61B (T20~T27) of PLC Slave ID#1.

PC→PLC

“ 01 03 06 14 00 08 04 80”

Sent message:

Field Name	Example (Hex)
START	No data input ≥ 10 ms
Slave Address	01
Command code	03
Starting Address	06
	14
Number of Points	00
	08
CRC CHK Low	04
CRC CHK High	80
END	No data input ≥ 10 ms

PLC→PC

" 01 03 10 00 01 00 02 00 03 00 04 00 05 00 06 00 07 00 08 72 98"

Feedback message:

Field Name	Example (Hex)
START	No data input ⩾ 10 ms
Slave Address	01
Command code	03
Bytes Count	10
Data Hi (T20)	00
Data Lo (T20)	01
Data Hi (T21)	00
Data Lo (T21)	02
Data Hi (T22)	00
Data Lo (T22)	03
Data Hi (T23)	00
Data Lo (T23)	04
Data Hi (T24)	00
Data Lo (T24)	05
Data Hi (T25)	00
Data Lo (T25)	06
Data Hi (T26)	00
Data Lo (T26)	07
Data Hi (T27)	00
Data Lo (T27)	08
CRC CHK Low	72
CRC CHK High	98
END	No data input ⩾ 10 ms

4.3.3 CRC CHK (check sum)

The CRC Check starts from "Slave Address" and ends in "The last data content." Calculation of CRC:

Step 1: Set the 16-bit register (CRC register) = FFFFH.

Step 2: Operate XOR on the first 8-bit message (Address) and the lower 8 bits of CRC register. Store the result in the CRC register

Step 3: Right shift CRC register for a bit and fill "0" into the highest bit.

Step 4: Check the lowest bit (bit 0) of the shifted value. If bit 0 is 0, fill in the new value obtained at step 3 to CRC register; if bit 0 is NOT 0, operate XOR on A001H and the shifted value and store the result in the CRC register.

Step 5: Repeat step 3 – 4 to finish all operation on all the 8 bits.

Step 6: Repeat step 2 – 5 until the operation of all the messages are completed. The final value obtained in the CRC register is the CRC checksum. Care should be taken when placing the LOW byte and HIGH byte of the obtained CRC checksum.

Calculation example of the CRC Check using the C language:

```
unsigned char* data      ← // index of the command message
unsigned char length     ← // length of the command message
unsigned int crc_chk(unsigned char* data, unsigned char length)
{
  int j;
  unsigned int reg_crc=0Xffff;
  while(length--)
  {
    reg_crc ^= *data++;
    for (j=0;j<8;j++)
    {
      If (reg_crc & 0x01) reg_crc=(reg_crc>>1) ^ 0Xa001; /* LSB(b0)=1 */
      else reg_crc=reg_crc >>1;
    }
  }
  return reg_crc;      // the value that sent back to the CRC register finally
}
```

Exception response:

The PLC is expected to return a normal response after receiving command messages from the master device. The following content depicts the conditions that either no response situation occurs or an error response is replied to the master device.

1. The PLC did not receive a valid message due to a communication error; thus the PLC has no response. In this case, condition of communication timeout has to be set up in the master device
2. The PLC receives a valid message without a communication error, but cannot accommodate it. In this case, an exception response will return to the master device. In the exception response, the most significant bit of the original command code is set to 1, and an exception code explaining the condition that caused the exception is returned.

An example of exception response of command code 01H and exception 02H:

Sent message:

Field Name	Example (Hex)
START	No data input ≥ 10 ms
Slave Address	01
Command code	01
Starting Address	04
	00
Number of Points	00
	10
CRC CHK Low	3C
CRC CHK High	F6
END	No data input ≥ 10 ms

Feedback message:

Field Name	Example (Hex)
START	No data input ≥ 10 ms
Slave Address	01
Function	81
Exception Code	02
CRC CHK Low	C1
CRC CHK High	91
END	No data input ≥ 10 ms

4.4 PLC Device Address

<table>
<tr><th rowspan="2">Device</th><th rowspan="2" colspan="2">Range</th><th colspan="3">Effective Range</th><th rowspan="2">MODBUS Address</th><th rowspan="2">Address</th></tr>
<tr><th>ES2/EX2/ EC5</th><th>SS2</th><th>SA2/SE SX2</th></tr>
<tr><td>S</td><td colspan="2">000~255</td><td rowspan="4">000~1023</td><td rowspan="4" colspan="2">000~1023</td><td>000001~000256</td><td>0000~00FF</td></tr>
<tr><td>S</td><td colspan="2">256~511</td><td>000257~000512</td><td>0100~01FF</td></tr>
<tr><td>S</td><td colspan="2">512~767</td><td>000513~000768</td><td>0200~02FF</td></tr>
<tr><td>S</td><td colspan="2">768~1023</td><td>000769~001024</td><td>0300~03FF</td></tr>
<tr><td>X</td><td colspan="2">000~377 (Octal)</td><td>000~377</td><td colspan="2">000~377</td><td>101025~101280</td><td>0400~04FF</td></tr>
<tr><td>Y</td><td colspan="2">000~377 (Octal)</td><td>000~377</td><td colspan="2">000~377</td><td>001281~001536</td><td>0500~05FF</td></tr>
<tr><td rowspan="2">T</td><td colspan="2">000~255 bit</td><td>000~255</td><td colspan="2">000~255</td><td>001537~001792</td><td>0600~06FF</td></tr>
<tr><td colspan="2">000~255 word</td><td>000~255</td><td colspan="2">000~255</td><td>401537~401792</td><td>0600~06FF</td></tr>
<tr><td>M</td><td colspan="2">000~255</td><td rowspan="6">0000
~
4095</td><td rowspan="6" colspan="2">0000~4095</td><td rowspan="6">002049~003584</td><td>0800~08FF</td></tr>
<tr><td>M</td><td colspan="2">256~511</td><td>0900~09FF</td></tr>
<tr><td>M</td><td colspan="2">512~767</td><td>0A00~0AFF</td></tr>
<tr><td>M</td><td colspan="2">768~1023</td><td>0B00~0BFF</td></tr>
<tr><td>M</td><td colspan="2">1024~1279</td><td>0C00~0CFF</td></tr>
<tr><td>M</td><td colspan="2">1280~1535</td><td>0D00~0DFF</td></tr>
<tr><td>M</td><td colspan="2">1536~1791</td><td rowspan="10">0000
~
4095</td><td rowspan="10" colspan="2">0000~4095</td><td rowspan="10">045057~047616</td><td>B000~B0FF</td></tr>
<tr><td>M</td><td colspan="2">1792~2047</td><td>B100~B1FF</td></tr>
<tr><td>M</td><td colspan="2">2048~2303</td><td>B200~B2FF</td></tr>
<tr><td>M</td><td colspan="2">2304~2559</td><td>B300~B3FF</td></tr>
<tr><td>M</td><td colspan="2">2560~2815</td><td>B400~B4FF</td></tr>
<tr><td>M</td><td colspan="2">2816~3071</td><td>B500~B5FF</td></tr>
<tr><td>M</td><td colspan="2">3072~3327</td><td>B600~B6FF</td></tr>
<tr><td>M</td><td colspan="2">3328~3583</td><td>B700~B7FF</td></tr>
<tr><td>M</td><td colspan="2">3584~3839</td><td>B800~B8FF</td></tr>
<tr><td>M</td><td colspan="2">3840~4095</td><td>B900~B9FF</td></tr>
<tr><td rowspan="2">C</td><td rowspan="2">000~199 (16-bit)</td><td>Bit</td><td>000~199</td><td colspan="2">000~199</td><td>003585~003784</td><td>0E00~0EC7</td></tr>
<tr><td>Word</td><td>000~199</td><td colspan="2">000~199</td><td>403585~403784</td><td>0E00~0EC7</td></tr>
<tr><td rowspan="3">C</td><td rowspan="3">200~255 (32-bit)</td><td>Bit</td><td>200~255</td><td colspan="2">200~255</td><td>003785~003840</td><td>0EC8~0EFF</td></tr>
<tr><td>Word</td><td>200~255</td><td colspan="2">200~255</td><td>401793~401903 (Odd address valid)</td><td>0700~076F</td></tr>
<tr><td>Dword</td><td>200~255</td><td colspan="2">200~255</td><td>403785~403840</td><td>0EC8~0EFF</td></tr>
<tr><td>D</td><td colspan="2">000~255</td><td rowspan="4">0000
~
9999</td><td rowspan="4">0000
~
4999</td><td rowspan="4">0000
~
9999</td><td rowspan="4">404097~405376</td><td>1000~10FF</td></tr>
<tr><td>D</td><td colspan="2">256~511</td><td>1100~11FF</td></tr>
<tr><td>D</td><td colspan="2">512~767</td><td>1200~12FF</td></tr>
<tr><td>D</td><td colspan="2">768~1023</td><td>1300~13FF</td></tr>
</table>

Device	Range	Effective Range			MODBUS Address	Address
		ES2/EX2/EC5	SS2	SA2/SE SX2		
D	1024~1279					1400~14FF
D	1280~1535				405377~408192	1500~15FF
D	1536~1791					1600~16FF
D	1792~2047					1700~17FF
D	2048~2303					1800~18FF
D	2304~2559					1900~19FF
D	2560~2815					1A00~1AFF
D	2816~3071					1B00~1BFF
D	3072~3327					1C00~1CFF
D	3328~3583					1D00~1DFF
D	3584~3839					1E00~1EFF
D	3840~4095					1F00~1FFF
D	4096~4351				436865~440960	9000~90FF
D	4352~4999					9100~91FF
D	4608~4863					9200~92FF
D	4864~5119					9300~93FF
D	5120~5375		N/A			9400~94FF
D	5376~5631					9500~95FF
D	5632~5887					9600~96FF
D	5888~6143					9700~97FF
D	6144~6399	0000 ~ 9999	N/A	0000 ~ 9999	436865~440960	9800~98FF
D	6400~6655					9900~99FF
D	6656~6911					9A00~9AFF
D	6912~7167					9B00~9BFF
D	7168~7423					9C00~9CFF
D	7424~7679					9D00~9DFF
D	7680~7935					9E00~9EFF
D	7936~8191					9F00~9FFF
D	8192~8447				440961~442768	A000~A0FF
D	8448~8703					A100~A1FF
D	8704~8959					A200~A2FF
D	8960~9215					A300~A3FF
D	9216~9471					A400~A4FF
D	9472~9727					A500~A5FF
D	9728~9983					A600~A6FF
D	9984~9999					A700~A70F
D	10000~11999	Applicable to DVP-SE			442769~444768	A710~AEDF

4.5 Command Code

4.5.1 Command Code: 01, Read Status of Contact (Input point X is not included)

Number of Points (max) = 255 (Dec) = FF (Hex)
Example：Read contacts T20~T56 from Slave ID#1
PC→PLC “:01 01 06 14 00 25 BF CR LF”

Sent message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	01
Starting Address Hi	06
Starting Address Lo	14
Number of Points Hi	00
Number of Points Lo	25
Error Check (LRC)	BF
ETX 1	0D (Hex)
ETX 0	0A (Hex)

Assume Number of Points in sent message is **n** (Dec), quotient of **n**/8 is **M** and the remainder is **N.** When **N** = 0, Bytes Count in feedback message will be **M**; when N≠0, Bytes Count will be **M+1**.

PLC→PC “:01 01 05 CD 6B B2 0E 1B D6 CR LF”

Feedback message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	01
Bytes Count	05
Data (Coils T27…T20)	CD
Data (Coils T35…T38)	6B
Data (Coils T43…T36)	B2
Data (Coils T51…T44)	0E
Data (Coils T56…T52)	1B
Error Check (LRC)	E6
END 1	0D (Hex)
END 0	0A (Hex)

4.5.2 Command Code: 02, Read Status of Contact (Input point X is included)

Example: Read status of contact Y024~Y070 from Slave ID#01

PC→PLC “: 01 02 05 14 00 25 BF CR LF”

Sent message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	02
Starting Address Hi	05
Starting Address Lo	14
Number of Points Hi	00
Number of Points Lo	25
Error Check (LRC)	BF
END 1	0D (Hex)
END 0	0A (Hex)

Assume Number of Points in sent message is **n** (Dec), quotient of **n**/8 is **M** and the remainder is **N.** When **N** = 0, Bytes Count in feedback message will be **M**; when N≠0, Bytes Count will be **M+1**.

PLC→PC “: 01 01 05 CD 6B B2 0E 1B E5 CR LF”

Feedback message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	02
Bytes Count	05
Data (Coils Y033…Y024)	CD
Data (Coils Y043…Y034)	6B
Data (Coils Y053…Y044)	B2
Data (Coils Y063…Y054)	0E
Data (Coils Y070…Y064)	1B
Error Check (LRC)	E5
END 1	0D (Hex)
END 0	0A (Hex)

4.5.3 Command Code: 03, Read Content of Register (T, C, D)

Example: Read coils T20~T27 from Slave ID#01

PC→PLC ": 01 03 06 14 00 08 DA CR LF"

Sent message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	03
Starting Address Hi	06
Starting Address Lo	14
Number of Points Hi	00
Number of Points Lo	08
Error Check (LRC)	DA
END 1	0D (Hex)
END 0	0A (Hex)

PLC→PC

":01 03 10 00 01 00 02 00 03 00 04 00 05 00 06 00 07 00 08 B8 CR LF"

Feedback message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	03
Bytes Count	10
Data Hi (T20)	00
Data Lo (T20)	01
Data Hi (T21)	00
Data Lo (T21)	02
Data Hi (T22)	00
Data Lo (T22)	03
Data Hi (T23)	00
Data Lo (T23)	04
Data Hi (T24)	00
Data Lo (T24)	05
Data Hi (T25)	00
Data Lo (T25)	06
Data Hi (T26)	00
Data Lo (T26)	07
Data Hi (T27)	00
Data Lo (T27)	08
Error Check (LRC)	C8
END 1	0D (Hex)
END 0	0A (Hex)

4.5.4 Command Code: 05, Force ON/OFF single contact

The Force data FF00 (Hex) indicates force ON the contact. The Force data 0000 (Hex) indicates force OFF the contact. Also, When MMNN = 0xFF00, the coil will be ON, when MMNN = 0x0000, the coil will be OFF. Other force data is invalid and will not take any effect.

Example: Force coil Y0 ON

PC→PLC “: 01 05 05 00 FF 00 F6 CR LF”

Sent message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	05
Coil Address Hi	05
Coil Address Lo	00
Force Data Hi	FF
Force Data Lo	00
Error Check (LRC)	F6
END 1	0D (Hex)
END 0	0A (Hex)

PLC→PC “: 01 05 05 00 FF 00 F6 CR LF”

Feedback message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	05
Coil Address Hi	05
Coil Address Lo	00
Force Data Hi	FF
Force Data Lo	00
Error Check (LRC)	F6
END 1	0D (Hex)
END 0	0A (Hex)

4.5.5 Command Code: 06, Set content of single register

Example: Set content of register T0: 12 34 (Hex)

PC→PLC “: 01 06 06 00 12 34 AD CR LF”

Sent message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	06
Register Address Hi	06
Register Address Lo	00
Preset Data Hi	12
Preset Data Lo	34
Error Check (LRC)	AD
END 1	0D (Hex)
END 0	0A (Hex)

PLC→PC “: 01 06 06 00 12 34 AD CR LF”

Feedback message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	06
Register T0 Address Hi	06
Register T0 Address Lo	00
Preset Data Hi	12
Preset Data Lo	34
Error Check (LRC)	AD
END 1	0D (Hex)
END 0	0A (Hex)

4.5.6 Command Code: 15, Force ON/OFF multiple contacts

Max contacts/coils available for Force ON/OFF: 255

Example: Set Coil Y007…Y000 = 1100 1101, Y011…Y010 = 01.

PC→PLC “: 01 0F 05 00 00 0A 02 CD 01 11 CR LF”

Sent message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	0F
Coil Address Hi	05
Coil Address Lo	00
Quantity of Coils Hi	00
Quantity of Coils Lo	0A
Byte Count	02
Force Data Hi	CD
Force Data Lo	01
Error Check (LRC)	11
END 1	0D (Hex)
END 0	0A (Hex)

PLC→PC “: 01 0F 05 00 00 0A E1 CR LF”

Feedback message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	0F
Register T0 Address Hi	05
Register T0 Address Lo	00
Preset Data Hi	00
Preset Data Lo	0A
Error Check (LRC)	E1
END 1	0D (Hex)
END 0	0A (Hex)

4.5.7 Command Code: 16, Set content of multiple registers

Example: Set register T0 to 00 0A , T1 to 01 02 .

PC→PLC “: 01 10 06 00 00 02 04 00 0A 01 02 D6 CR LF”

Sent message:

Field Name	ASCII
STX	:
Slave Address	01
Command code	10
Starting Address Hi	06
Starting Address Lo	00
Number of Register Hi	00
Number of Register Lo	02
Byte Count	04
Data Hi	00
Data Lo	0A
Data Hi	01
Data Lo	02
Error Check (LRC)	D6
END 1	0D(Hex)
END 0	0A(Hex)

PLC→PC “: 01 10 06 00 00 02 E7 CR LF”

Feedback message:

Field Name	ASCII
STX	3A
Slave Address	01
Command code	10
Starting Address Hi	06
Starting Address Lo	00
Number of Registers Hi	00
Number of Registers Lo	02
Error Check (LRC)	E7
END 1	0D (Hex)
END 0	0A (Hex)

Memo

Sequential Function Chart

5

This chapter provides information for programming in SFC mode.

Table of Contents

5.1 Step Ladder Instruction [STL], [RET]

Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2	SX2
STL	S0~S1023	Starts STL program	1				

Explanation:

STL Sn constructs a step point. When STL instruction appears in the program, the main program will enter a step ladder status controlled by steps. The initial STL program has to start from S0 ~ S9 as initial step points. The No. of Step points cannot be repeated.

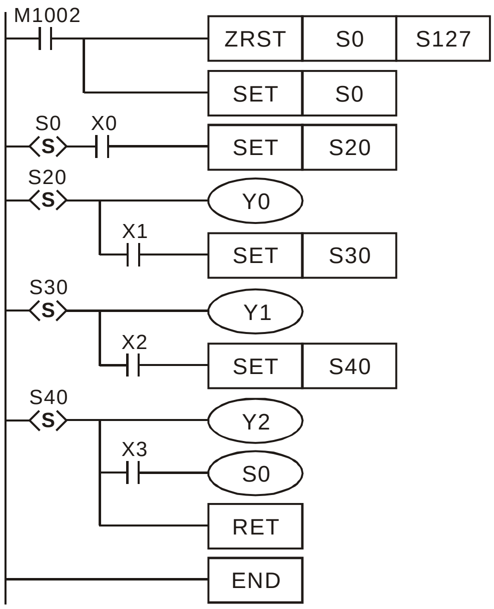
Mnemonic	Operands	Function	Program steps	Controllers			
				ES2/EX2/ EC5	SS2	SA2	SX2
RET	None	Ends STL program	1				

Explanation:

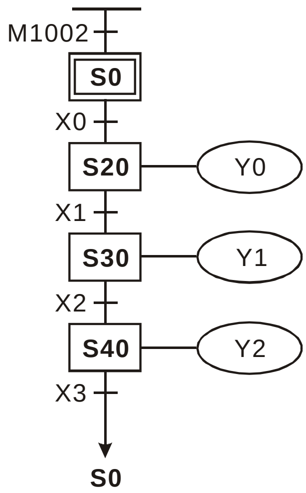
RET instruction indicates the end of a step ladder program starting from S0 ~ S9, i.e. the execution returns to main program after RET is executed. Maximum 10 initial steps (S0 ~ S9) can be applied and every initial step requires a RET instruction as an end of STL program. With the step ladder program composed of STL/RET instructions, SFC can perform a step by step control process.

Program Example:

Step ladder diagram:



SFC:



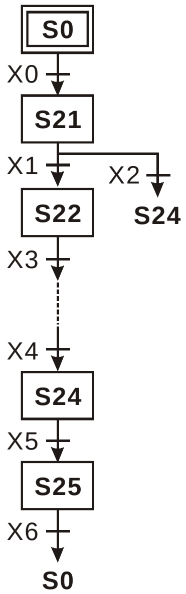
5.2 Sequential Function Chart (SFC)

In the application of automation control, a seamless combination between electrical control and mechanical control is required for completing an automation process. The sequential control of automation process can be divided into several steps (states). Each step is designated with own action and the transition from one step to another generally requires some transition criteria (condition). The action of the previous step finishes as long as all criteria is true. When next step begins, the action of previous step will be cleared. The step-by-step transition process is the concept for designing sequential function chart (SFC).

Features:

1. Users do not have to consider the sequential relationship between outputs as general ladder logic because STL operation process can execute multiple outputs or interlocked outputs automatically. An easy sequential design between the steps is the only thing required to control the machines.
2. The actions in SFC are easy to understand. Also, it's easy to do a trial operation, error detecting or period maintenance.
3. SFC functions as a flow chart. The STL operation works on the internal step relay S, which is also the step points representing each state in SFC. When current step is finished, the program proceeds to the next step according to the transition condition and the desired continuous control purpose can be achieved by this process.
4. Cycle process can be performed. Please refer to the SFC opposite. Initial step S0 transfers to general step S21 by transition condition X0. S21 transfers to S22 or jumps to S24 by the condition X1 and X2. The process finally proceeds to S25 then a single cycle process is completed when S25 returns to S0 with transition condition X6 fulfilled.

SFC:



Explanation on SFC Toolbar Icons in Ladder Editor (WPLSoft)	
	Ladder diagram mode. The icon inserts general ladder diagram before the STL diagram, usually the instructions for initializing the STL program.
	Initial step in SFC. S0 ~ S9.are applicable

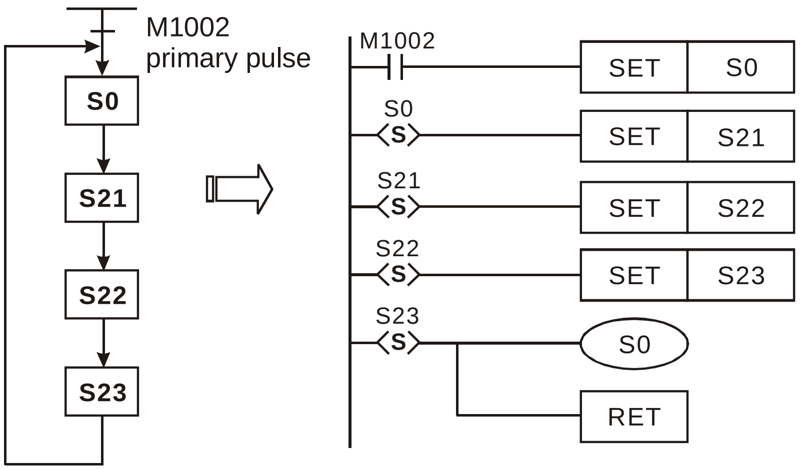
F3	General step. S10 ~ S1023 are applicable.
F4	Step jump. Used for a step to jump to another non-adjacent step. (Jumping up/down to non-adjacent steps in the same sequence, returning to initial step, or jumping among different sequences.)
F5	Transition condition. The transition condition to move between each step point.
F6	Alternative divergence. Alternative divergence is used for a step point to transfer to different corresponding step points by different transition conditions.
F7	Alternative convergence. Alternative convergence is used for two step points or more to transfer to the same step point according to transition condition.
F8	Simultaneous divergence. Simultaneous divergence is used for a step point to transfer to two step points or more by the same transition condition.
F9	Simultaneous convergence. Simultaneous convergence is used for two step points or more to transfer to the same step point with the same transition condition when multiple conditions are fulfilled at the same time.

5.3 The Operation of STL Program

Step ladder diagram (STL) is a programming method for users to write a program which functions similar to SFC. STL provides PLC program designers a more readable and clear programming method as drawing a flow chart. The sequences or steps in the below SFC is quite understandable and can be translated into the ladder diagram opposite.

STL program starts with STL instruction and ends with RET instruction. STL Sn constructs a step point. When STL instruction appears in the program, the main program will enter a step ladder status controlled by steps. RET instruction indicates the end of a step ladder program starting from initial steps S0 ~ S9 and every initial step requires a RET instruction as an end of STL program.

If there is no RET instruction at the end of a step sequence, errors will be detected by WPLSoft.

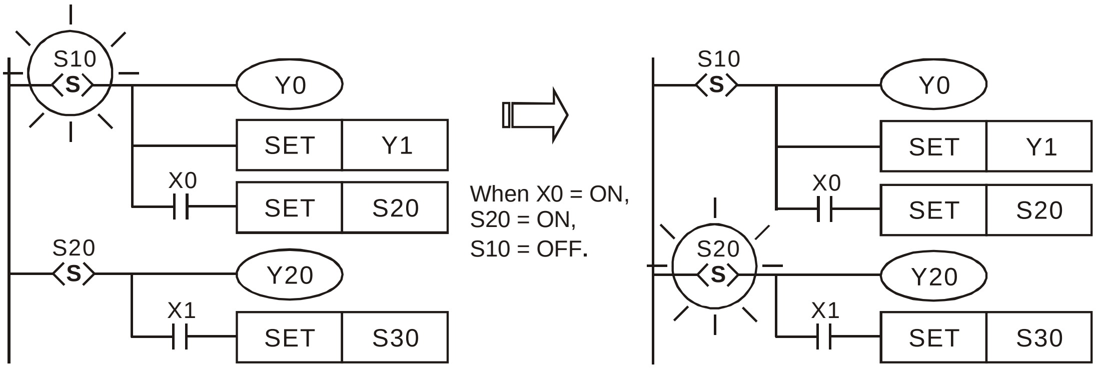


Actions of Step Points:

STL program is composed of many step points, and each step point represents a single task in the STL control process. To perform a sequential control result, every step point needs to do 3 actions.

1. Drive output coils
2. Designate the transition condition
3. Designate which step will take over the control from the current step

Example:



Explanation:

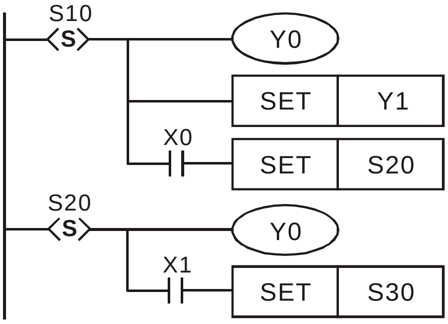
When S10 = ON, Y0 and Y1 will be ON. When X0 = ON, S20 will be ON and Y20 will be ON. When S10 = OFF, Y0 will be OFF but Y1 will still be ON (SET instruction is applied on Y1, so Y1 will be ON and latched.)

STL Transition:

When step point Sn is ON, its following output circuit will be activated. When Sn = OFF, its following output circuit will be OFF. The interval between the activation of the step point and its following output circuit is one scan cycle.

Repeated Usage of Output Coil:

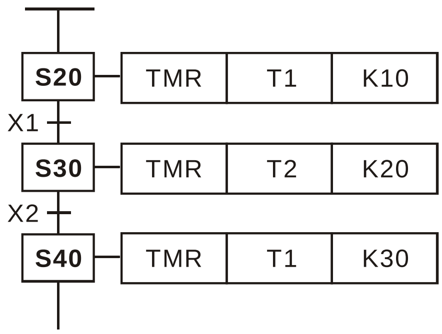
1. Output coils of the same number could be used in different step points.
2. See the diagram opposite. There can be the same output device (Y0) among different steps (sequences). Y0 remains ON when S10 transfers to S20.
3. Y0 will be OFF due to the transition from S10 to S20. However when S20 is ON, Y0 will be ON again. Therefore in this case, Y0 remains ON when S10 transfers to S20.



4. For general ladder diagrams, repeated usages of output coils should be avoided. The No. of output coil used by a step should also avoid being used when the step ladder diagram returns to a general ladder diagram.
5. Note: If the operands are qualified by E or F, the repeated usages of output coils in the steps is no longer allowed, e.g. OUT Y0E1.

Repeated usage of timer:

See the opposite diagram. Timers can only be used repeatedly in non-adjacent steps.

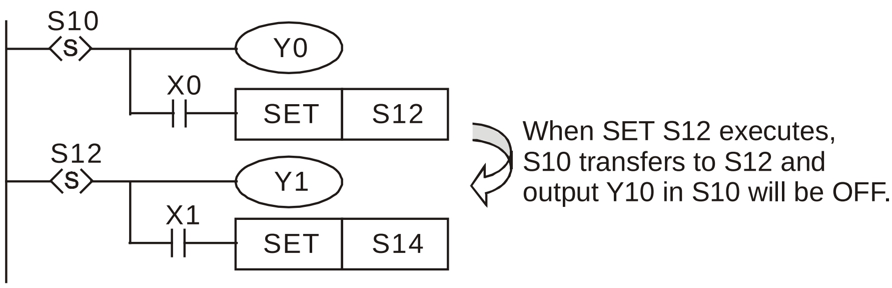


Transfer of Step Points:

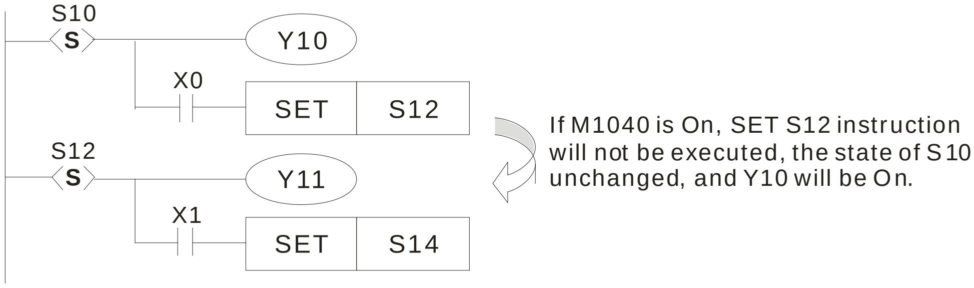
SET Sn and OUT Sn instructions are used to enable (or transfer to) another step. Because there can be many step control sequences (i.e. the initial steps starting with S0 ~ S9) existing in the program. The transfer of a step can take place in the same step sequence, or be transferred to different step sequence. Usages of SET Sn and OUT Sn are different according to the transfer methods. Please see the explanations below

SET Sn

1. Used for driving the next step in the same sequence. After the transition, all output in the previous step will be OFF.



2. If M1014 is used, and it is On, the transfer of the steps will be prohibited, and the states of the steps remain unchanged.



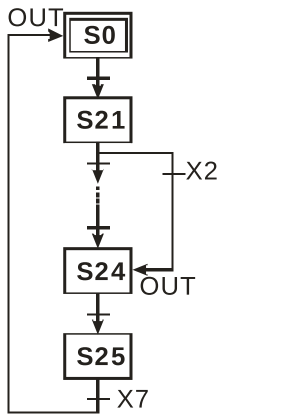
OUT Sn

Used for returning to the initial step in the same step sequence. Also for jumping up/down to non-adjacent steps in the same sequence, or separating steps in different sequences. After the transition, all output from the previous status will be cleared.

① Returning to the initial step in the same sequence.

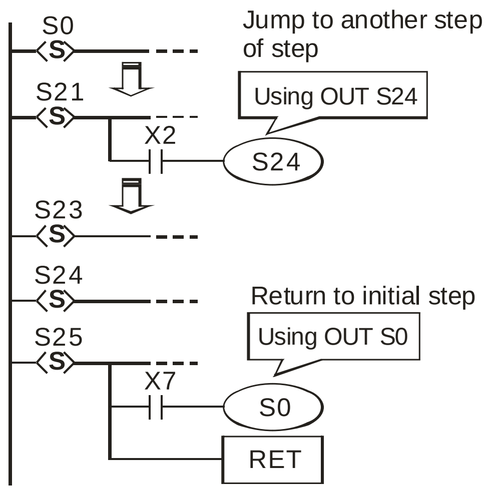
② Jumping up/down to non-adjacent steps in the same sequence.

SFC:



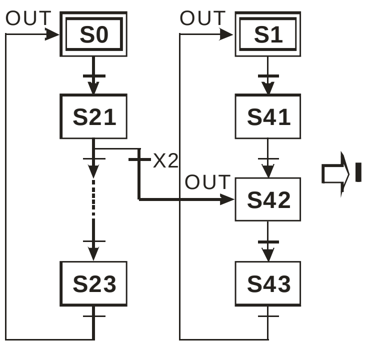
S25 returns to the initial step S0 by using OUT.

Ladder diagram:

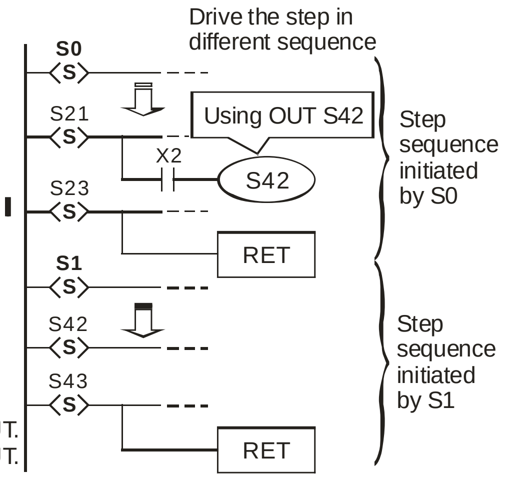


③ Separating steps in different sequences.

SFC:



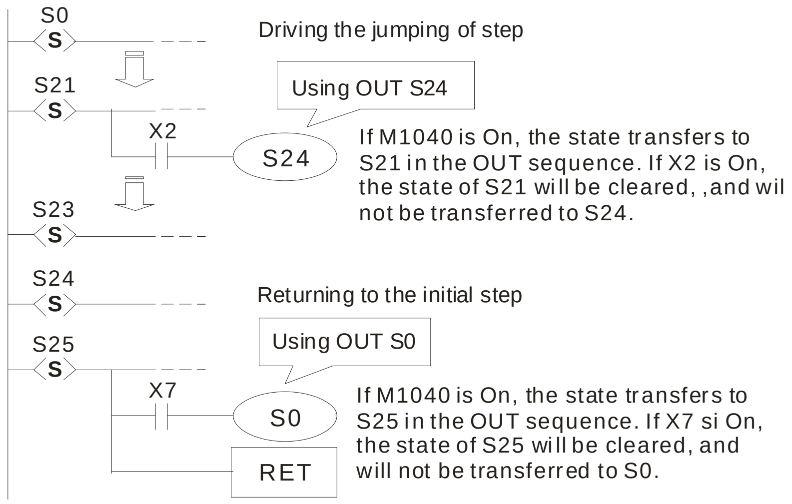
Ladder diagram:



Two different step sequence: S0 and S1
S23 returns to initial step S0 by using OUT.
S43 returns to initial step S1 by using OUT.

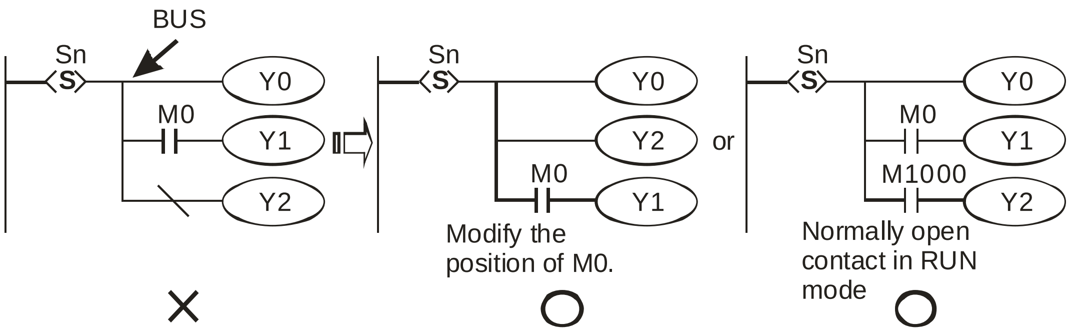
If M1014 is used, and M1040 is On, the steps in the same sequence will be cleared to Off.

Ladder diagram:



Cautions for Driving Output Point:

Once LD or LDI instructions are written into the second line after the step point, the bus will not be able to connect output coils directly otherwise errors will occur when compiling the ladder diagram. The following diagram explains the methods for correcting the ladder ion correct diagram.



Restrictions on Using Certain Instructions:

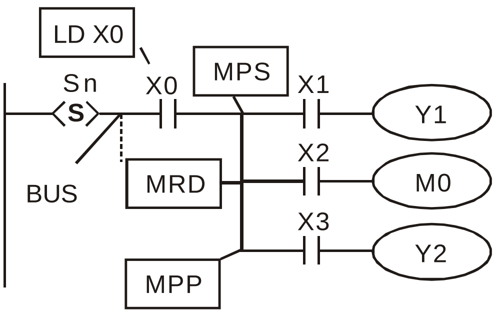
Serial/parallel circuits or instructions in general ladder diagram are also applicable in step points of STL diagram. However, there are restrictions on some of the instructions. Care should be taken when using the instructions listed in the table below.

Basic Instructions Applicable in a Step

Basic instruction / Step point		LD/LDI/LDP/LDF AND/ANI/ANDP/ANDF OR/ORI/ORP/ORF INV/OUT/SET/RST	ANB/ORB MPS/MRD/MPP	MC/MCR
Primary step point/ General step point		Yes	Yes	No
Diverging step point/ Converging step point	General output	Yes	Yes	No
	Step point transfer	Yes	Yes	No

1. DO NOT use MC/MCR instruction in the step.
2. DO NOT use STL instruction in a general subroutine or interruption subroutine.
3. CJ instruction can be used in STL instruction, however this is not recommended because the actions will thus become more complicated.
4. Position of MPS/MRD/MPP instruction:

Ladder diagram:



Instruction code:

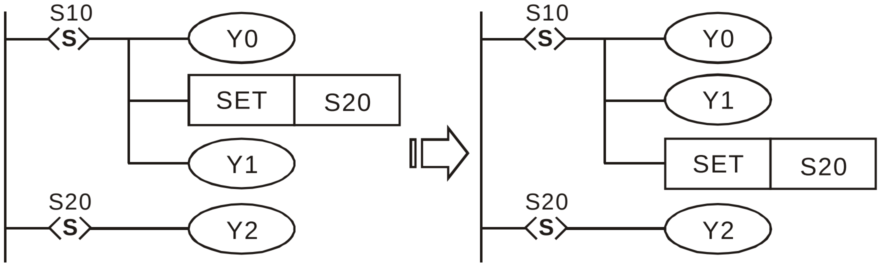
STL	Sn
LD	X0
MPS	
AND	X1
OUT	Y1
MRD	
AND	X2
OUT	M0
MPP	
AND	X3
OUT	Y2

Explanation:

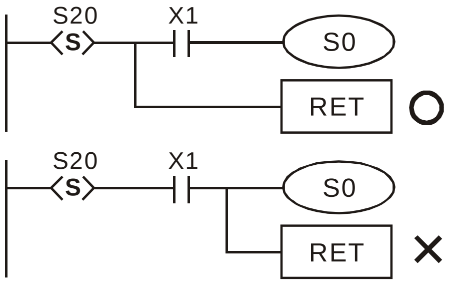
MPS/MRD/MPP instruction cannot be used directly on the new bus. You have to execute LD or LDI instruction first before applying MPS/MRD/MPP.

Other Points to Note:

1. The instruction used for transferring the step (SET S□ or OUT S□) are suggested to be executed after all the relevant outputs and actions in the current step are completed.
   The execution results by the PLC are the same. However, if there are many conditions or actions in S10, it is recommended to modify the diagram in the left into the diagram in the right, which executes SET S20 after all actions are completed. The sequence will be more understandable and clear with this modification.

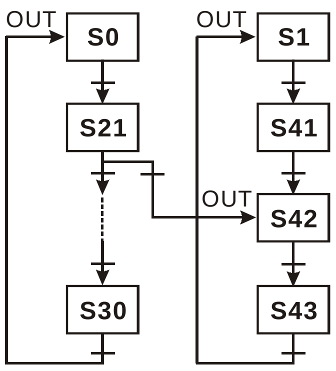


2. As indicated in the below diagram, make sure to connect RET instruction directly after the step point rather than the NO or NC contact.



5.4 Points to Note for Designing a Step Ladder Program

1. The first step in the SFC is called the “initial step", S0 ~ S9. Use the initial step as the start of a sequence and ends with RET instruction.
2. If no STL instruction is in use, step point S can be used as a general-purpose auxiliary relay..
3. When STL instruction is in use, the No. of step S cannot be repeated.
4. Types of sequences:
   - Single sequence: Only one simple sequence without alternative divergence, alternative convergence, simultaneous divergence or simultaneous convergence in the program.
   - Complicated single sequence: Only one sequence with alternative divergence, alternative convergence, simultaneous divergence and simultaneous convergence in the program.
   - Multiple sequences: More than one sequence in a program, maximum 10 sequences, S0 ~ S9.
5. Sequence jump: Multiple sequences are allowed to be written into the step ladder diagram.
   - There are two sequences, S0 and S1. PLC writes in S0 ~ S30 first and S1 ~ S43 next..
   - Users can assign a step in the sequence to jump to any step in another sequence.
   - When the condition below S21 is fulfilled, the sequence will jump to step S42 in sequence S1, which is called “sequence jump.”



6. Restrictions on diverging sequence: Please refer to section **5.5** for examples
   a) Max. 8 step points could be used for single divergence sequence.
   b) Max. 16 step points could be used for the convergence of multiple diverted sequences.
   c) Users can assign a step in the sequence to jump to any step in another sequence.
7. Reset step points and disable outputs
   a) Use the ZRST instruction to reset (turn off) a specified step sequence..
   b) Set ON the flag M1034 to disable Y outputs.
8. Latched step:
   The ON/OFF status of the latched step will be memorized when the power of the PLC is switched off. When the PLC is powered up again, PLC will resume the status before power-off and executes from the interrupted point. Please be aware of the area for the latched steps.

9. Special auxiliary relays and special registers: For more details please refer to **5.6 IST Instruction**.

Device	Description
M1040	Disabling step transition.
M1041	Step transition start. Flag for IST instruction.
M1042	Enabling pulse operation. Flag for IST instruction.
M1043	Zero return completed. Flag for IST instruction.
M1044	Zero point condition. Flag for IST instruction.
M1045	Disabling “all output reset” function. Flag for IST instruction.
M1046	Indicating STL status. M1046 = ON when any step is ON
M1047	Enabling STL monitoring
D1040	No. of the 1st step point which is ON.
D1041	No. of the 2nd step point which is ON
D1042	No. of the 3rd step point which is ON.
D1043	No. of the 4th step point which is ON
D1044	No. of the 5th step point which is ON.
D1045	No. of the 6th step point which is ON
D1046	No. of the 7th step point which is ON.
D1047	No. of the 8th step point which is ON

5.5 Types of Sequences

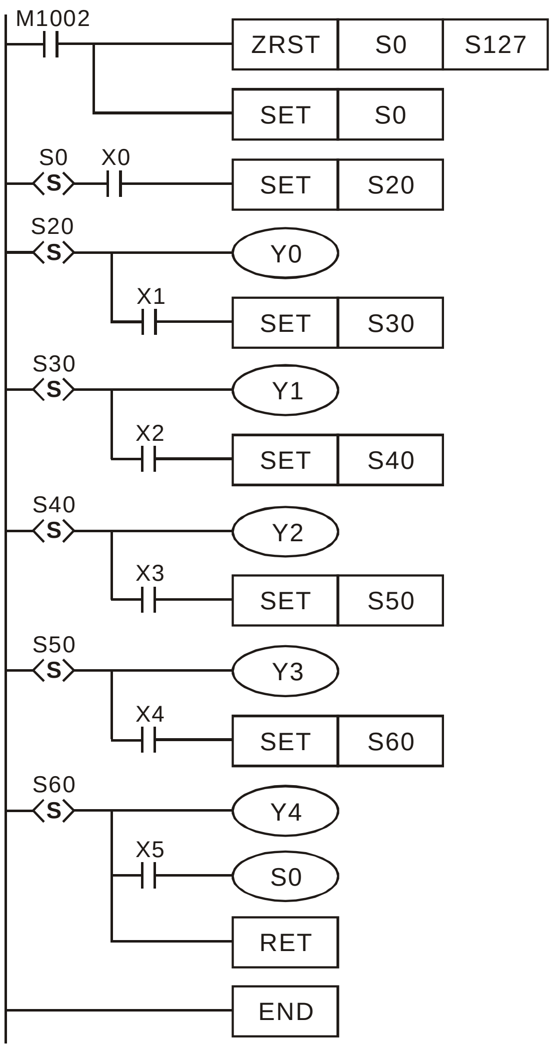
Single Sequence: The basic type of sequence

The first step in a step ladder diagram is called initial step, ranged as S0 ~ S9. The steps following the initial step are general steps numbered as S10 ~ S1023. When IST instruction is applied, S10 ~ S19 will become the steps for zero return operation.

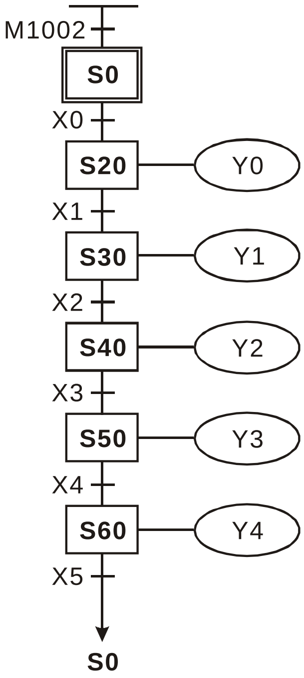
1. Single Sequence without Divergence and Convergence

   After a sequence is completed, the control power on the steps will be transferred to the initial step.

Step Ladder Diagram

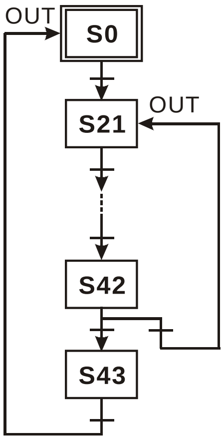


SFC diagram

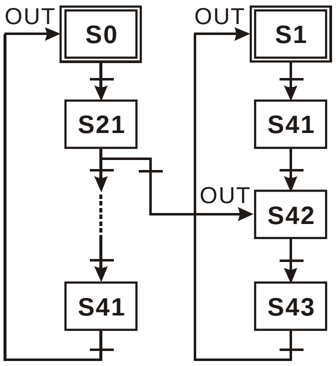


2. Step Jump

a) The control power over the step is transferred to a certain step on top.

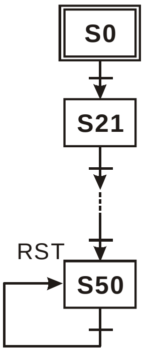


b) The control power over the step is transferred to the step in another sequence.



3. Reset Sequence

As the opposite diagram indicates, S50 will reset itself when the transition condition is fulfilled and the sequence ends here.

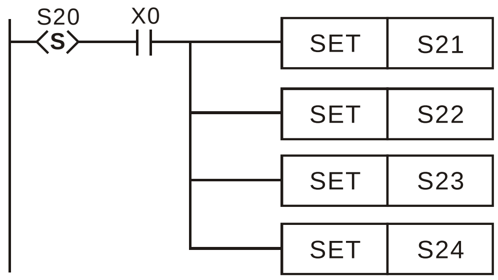


Complicated Single Sequence: Includes simultaneous divergence, alternative divergence, simultaneous convergence and alternative convergence

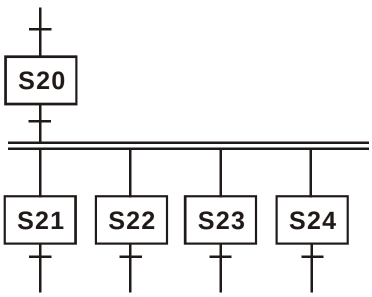
1. Structure of Simultaneous Divergence

   When the condition at the current step is true, the step can be transferred to multiple steps. For example, when X0 = ON, S20 will be simultaneously transferred to S21, S22, S23 and S24.

   Ladder diagram of simultaneous divergence:



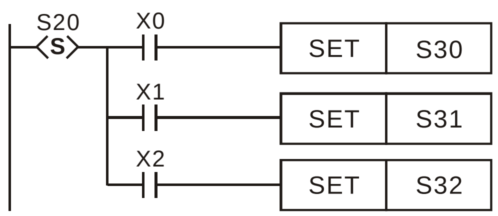
   SFC diagram of simultaneous divergence:



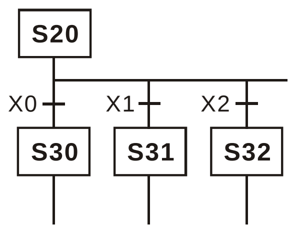
2. Structure of Alternative Divergence

   When the individual condition at the current status is true, the step will be transferred to another individual step. For example, when X0 = ON, S20 will be transferred to S30; when X1 = ON, S20 will be transferred to S31; when X2 = ON, S20 will be transferred to S32.

   Ladder diagram of alternative divergence:



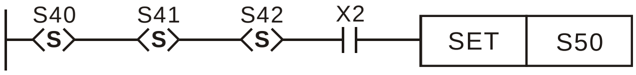
   SFC diagram of alternative divergence:



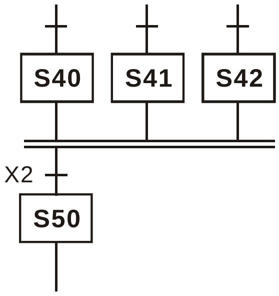
3. Structure of Simultaneous Convergence

   Consecutive STL instructions construct a simultaneous convergence structure. When the transition condition is true after continuous steps, the operation will be transferred to next step. In simultaneous convergence, only when all sequences are completed will the transfer be allowed.

   Ladder diagram of simultaneous convergence:



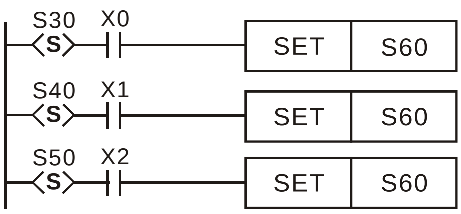
   SFC diagram of simultaneous convergence:



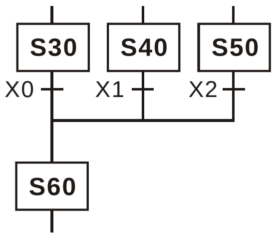
4. Structure of Alternative Convergence

   The following ladder explains the structure of alternative convergence. Program operation will transfer to S60 as long as one of the transition conditions of S30, S40 or S50 is ON.

   Ladder diagram of alternative convergence:

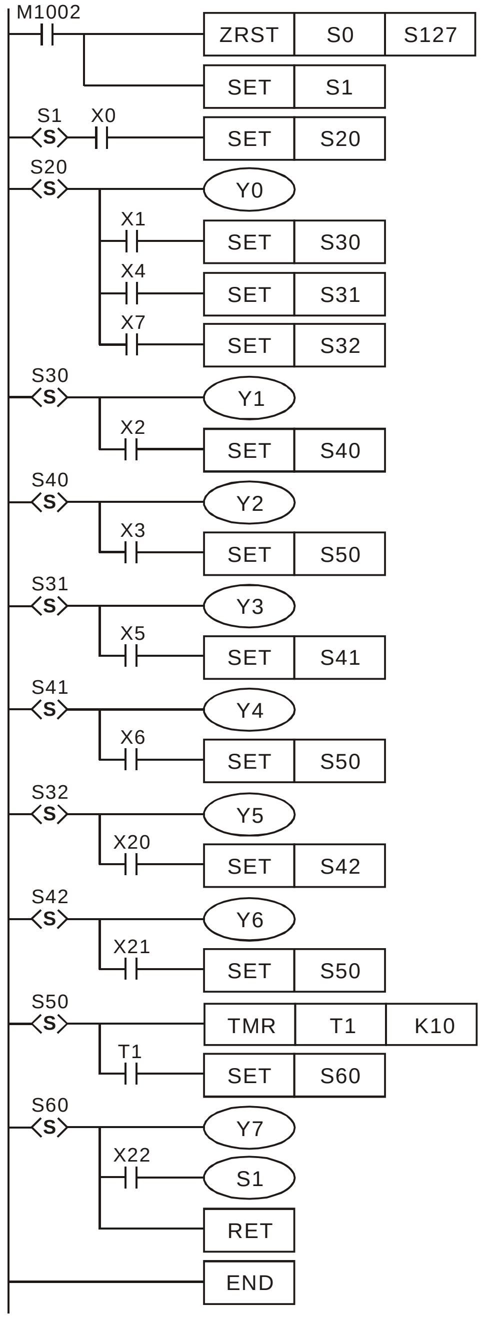


   SFC diagram of alternative convergence:

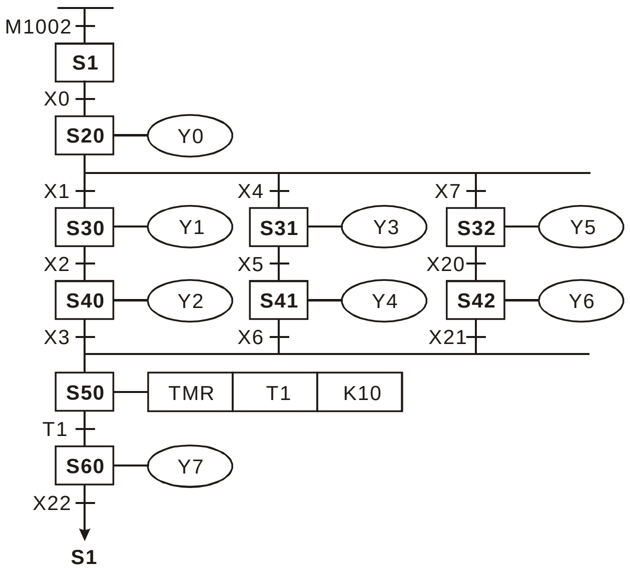


Example of alternative divergence & alternative convergence:

Step Ladder Diagram:

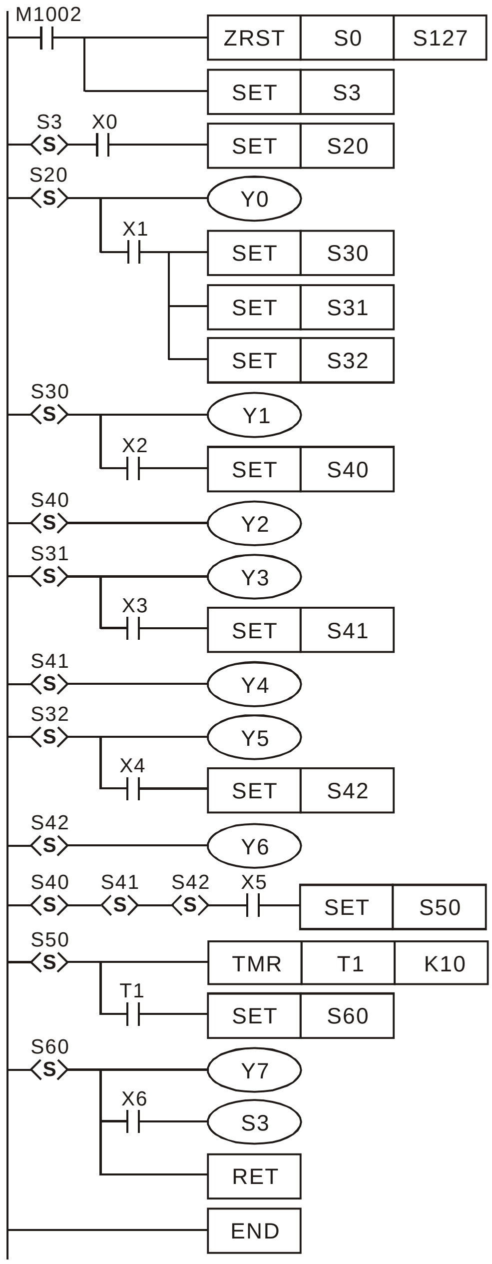


SFC Diagram:

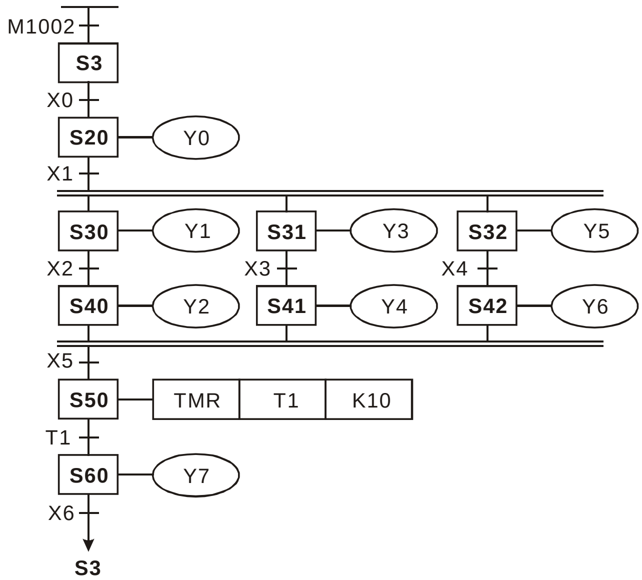


Example of simultaneous divergence & simultaneous convergence:

Step Ladder Diagram:

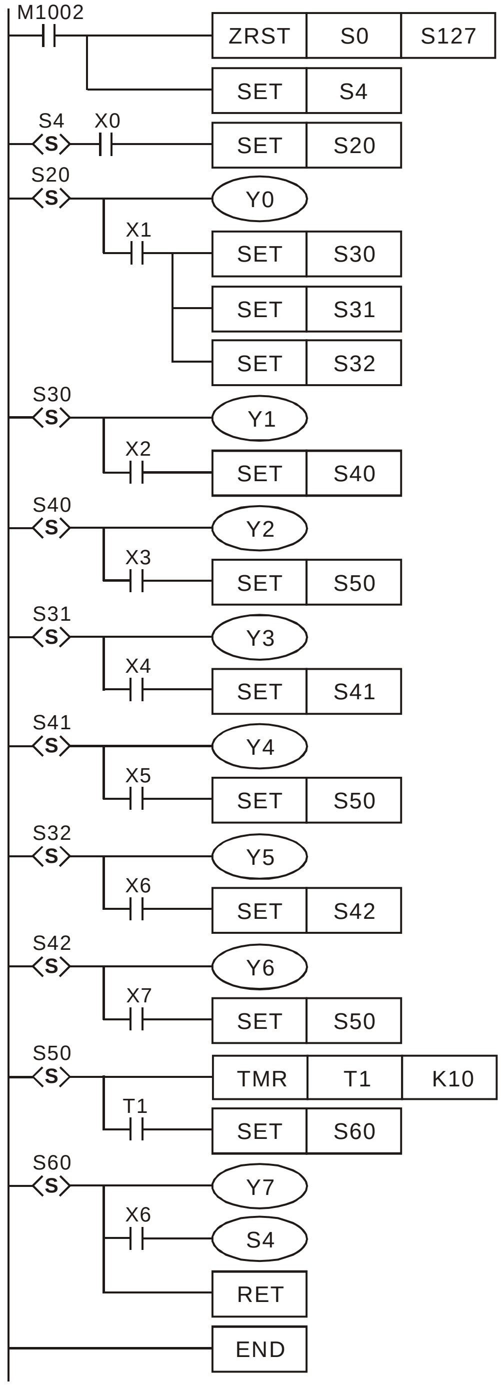


SFC Diagram:

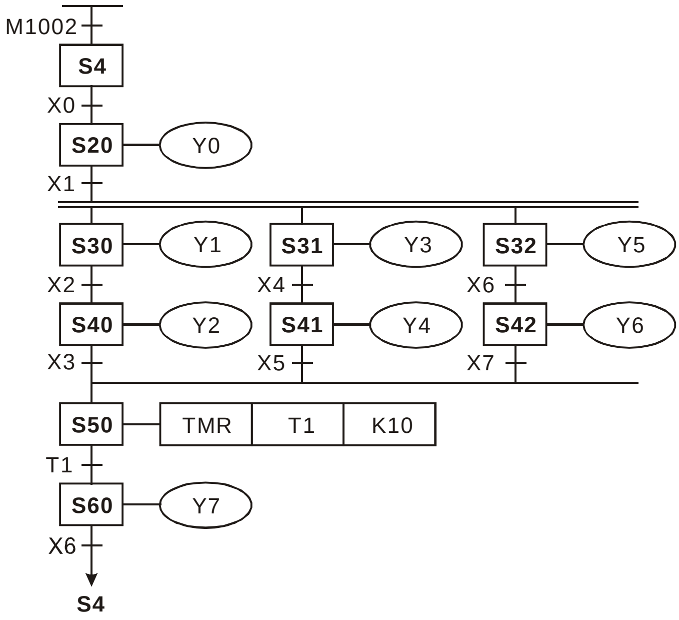


Example of the simultaneous divergence & alternative convergence:

Step Ladder Diagram:

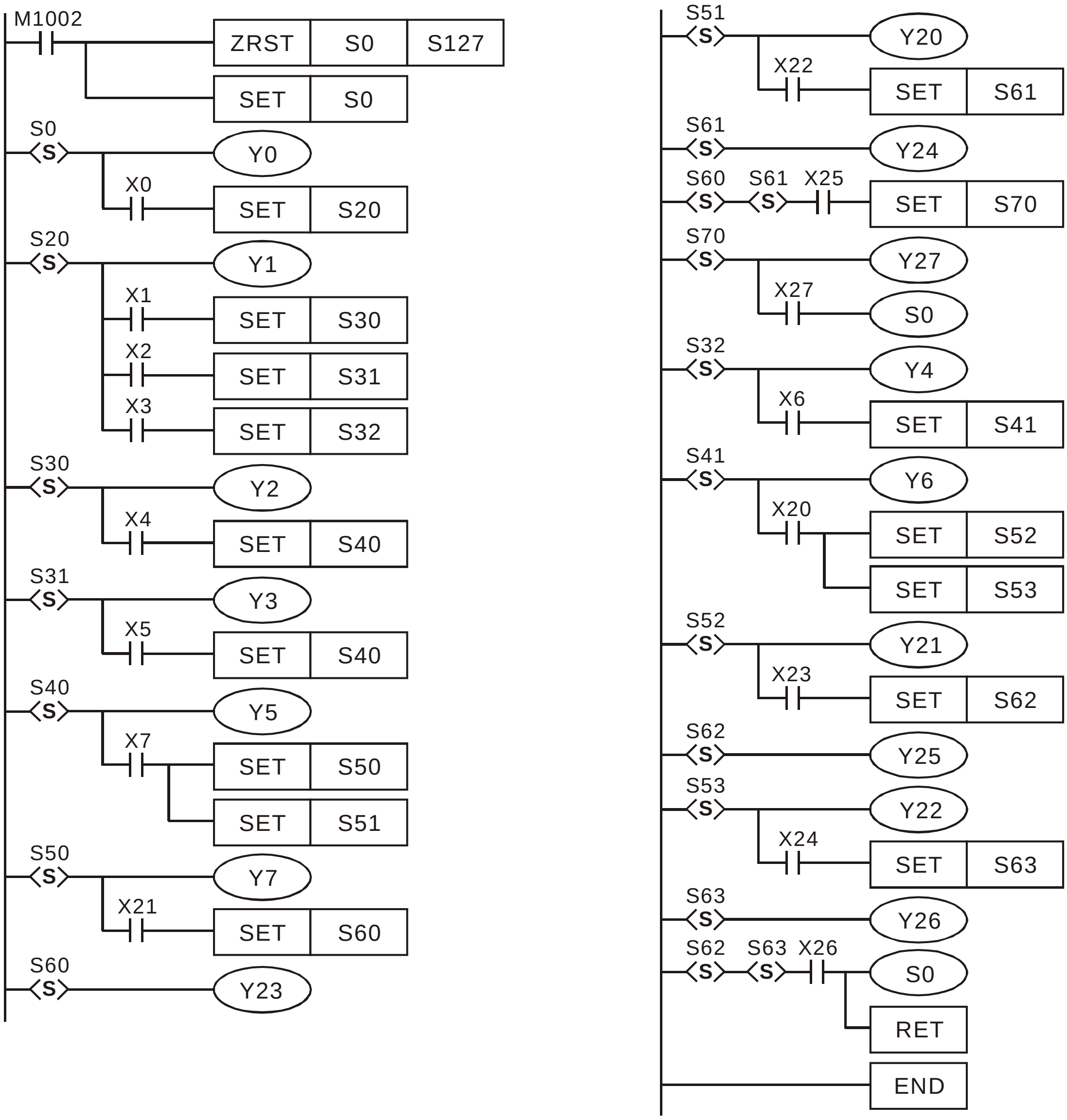


SFC Diagram:

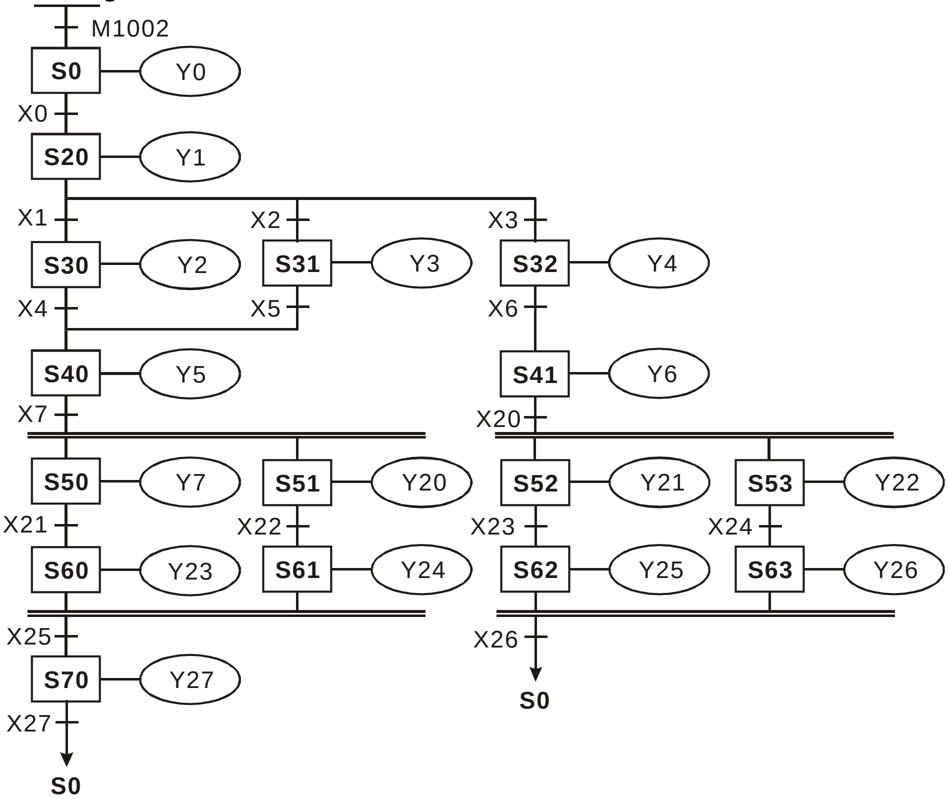


Combination example 1: (Includes alternative divergence/convergence and simultaneous divergence/convergence)

Step Ladder Diagram:

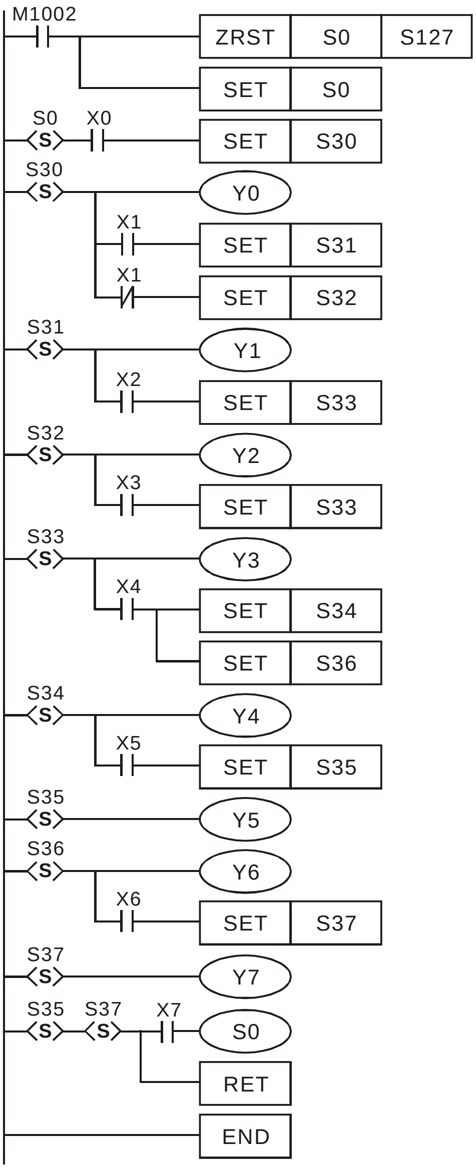


SFC Diagram:

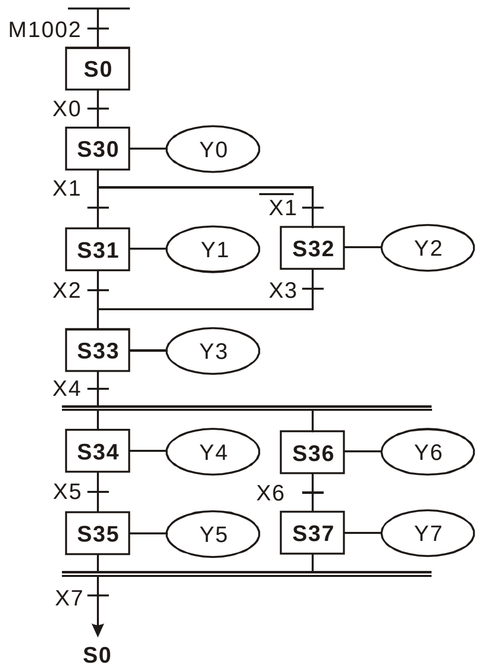

M1002
S0
Y0
X0
S20
Y1
X1
X2
X3
S30
Y2
S31
Y3
S32
Y4
X4
X5
X6
S40
Y5
S41
Y6
X7
X20
S50
Y7
S51
Y20
S52
Y21
S53
Y22
X21
X22
X23
X24
S60
Y23
S61
Y24
S62
Y25
S63
Y26
X25
X26
S70
Y27
S0
X27
S0

Combination example 2: (Includes alternative divergence/convergence and simultaneous divergence/convergence)

Step Ladder Diagram:



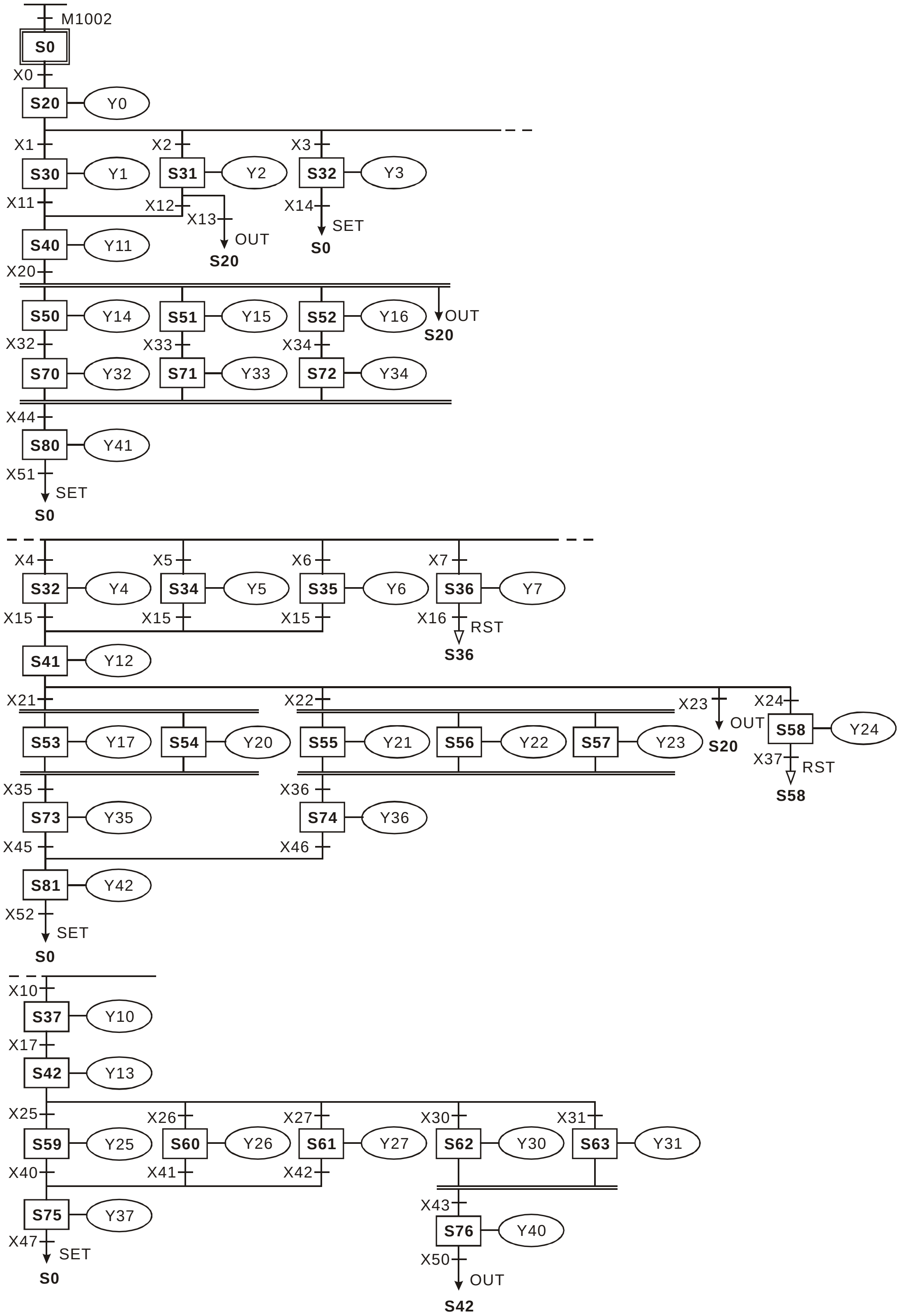
SFC Diagram:



Restrictions on Divergence Sequence:

1. Max. 8 step points could be used for single divergence sequence. As the diagram below, there are maximum 8 diverged steps S30 ~ S37 after step S20.
2. Max. 16 step points could be used for the convergence of multiple diverted sequences. As the diagram below, there are 4 steps diverged after S40, 7 steps diverged after S41, and 5 steps diverged after S42. There are maximum 16 loops in this sequence.
3. Users can assign a step in the sequence to jump to any step in another sequence.

SFC Diagram:


M1002
S0
X0
S20
Y0
X1
X2
X3
S30
Y1
S31
Y2
S32
Y3
X11
X12
X13
X14
SET
OUT
S40
Y11
S20
S0
X20
S50
Y14
S51
Y15
S52
Y16
OUT
S20
X32
X33
X34
S70
Y32
S71
Y33
S72
Y34
X44
S80
Y41
X51
SET
S0
X4
X5
X6
X7
S32
Y4
S34
Y5
S35
Y6
S36
Y7
X15
X15
X15
X16
RST
S36
S41
Y12
X21
X22
X23
X24
OUT
S58
Y24
S53
Y17
S54
Y20
S55
Y21
S56
Y22
S57
Y23
S20
X37
RST
S58
X35
X36
S73
Y35
S74
Y36
X45
X46
S81
Y42
X52
SET
S0
X10
S37
Y10
X17
S42
Y13
X25
X26
X27
X30
X31
S59
Y25
S60
Y26
S61
Y27
S62
Y30
S63
Y31
X40
X41
X42
X43
S75
Y37
S76
Y40
X47
SET
X50
OUT
S0
S42

5.6 IST Instruction

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2/ EC5	SS2	SA2	SX2
60	IST	S D1 D2	Initial State				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	IST: 7 steps
S	*	*	*													
D_1				*												
D_2				*												

PULSE				16-bit				32-bit			
ES2/EX2/ EC5	SS2	SA2	SX2	ES2/EX2/ EC5	SS2	SA2	SX2	ES2/EX2/ EC5	SS2	SA2	SX2

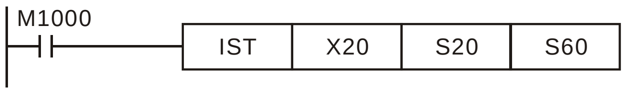
Operands:

S: Source device for assigning pre-defined operation modes (8 consecutive devices). **D_1** The smallest No. of step points in auto mode. **D_2**: The greatest No. of step points in auto mode.

Explanations:

1. The IST is a handy instruction specifically for the initial state of the step ladder operation modes.
2. The range of **D_1** and **D_2** : S20~S911, **D_1** < **D_2**.
3. IST instruction can only be used one time in a program.

Program Example 1:



1. Operation mode:

   S: X20: Individual operation (Manual operation)
   X21: Zero return
   X22: Step operation
   X23: One cycle operation
   X24: Continuous operation
   X25: Zero return start switch
   X26: Start switch
   X27: Stop switch

2. When IST instruction is executed, the following special auxiliary relays will be assigned automatically.

   M1040: Movement inhibited
   M1041: Movement start
   M1042: Status pulse
   M1047: STL monitor enable
   S0: Manual operation/initial state step point
   S1: Zero point return/initial state step point
   S2: Auto operation/initial state step point

3. When IST instruction is used, S10~S19 are occupied for zero point return operation and cannot be used as a general step point. In addition, when S0~S9 are in use, S0 initiates “manual operation mode”, S1 initiates “zero return mode” and S2 initiates “auto mode”. Thus, the three step points of initial state have to be programmed in first priority.
4. When S1 (zero return mode) is initialized, i.e. selected, zero return will NOT be executed if any of the state S10~S19 is ON.

5. When S2 (auto mode) is initialized, i.e. selected, auto mode will NOT be executed if M1043 = ON or any of the state between $\mathbf{D_1}$ to $\mathbf{D_{21}}$ is ON.

Program Example 2:

Robot arm control (by IST instruction):

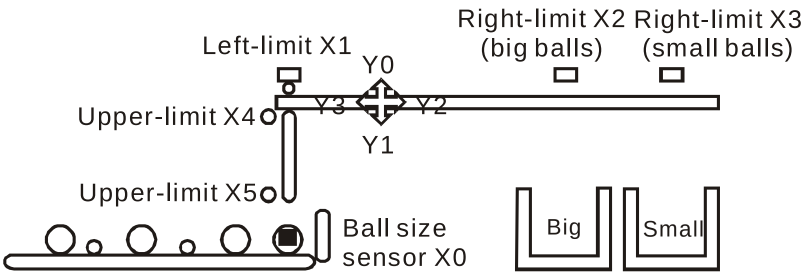
1. Control purpose:

   Select the big balls and small balls and move them to corresponding boxes. Configure the control panel for each operation.

2. Motion of the Robot arm:

   lower robot arm, clip balls, raise robot arm, shift to right, lower robot arm, release balls, raise robot arm, shift to left to finish the operation cycle.

3. I/O Devices



4. Operation mode:

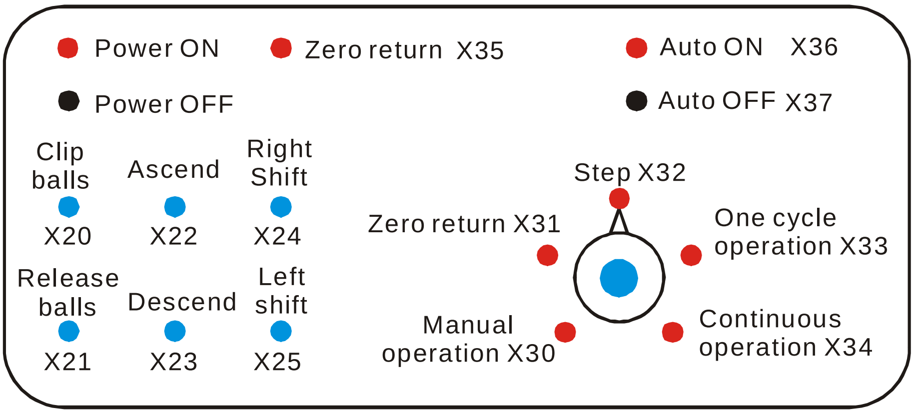
   Single step: Press single button for single step to control the ON/OFF of external load.

   Zero return: Press zero return button to perform homing on the machine.

   Auto (Single step / One cycle operation / Continuous operation):

   - Single step: the operation proceeds with one step every time when Auto ON is pressed.
   - One cycle operation: press Auto ON at zero position, the operation performs one full cycle operation and stops at zero point. If Auto OFF is pressed during the cycle, the operation will pause. If Auto ON is pressed again, the operation will resume the cycle and stop at zero point.
   - Continuous operation: press Auto ON at zero position, the operation will perform continuous operation cycles. If Auto OFF is pressed, the operation will stop at the end of the current cycle.

5. Control panel

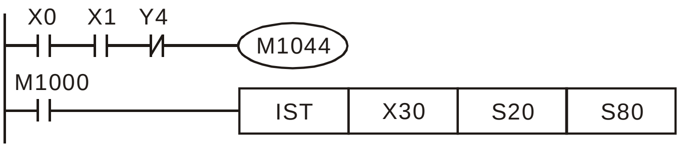


a) X0: ball size sensor.

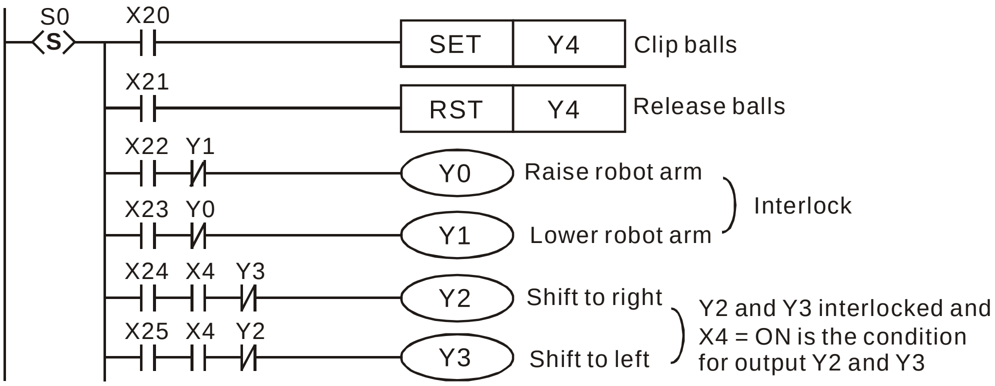
b) X1: left-limit of robot arm, X2: right-limit (big balls), X3: right-limit (small balls), X4: upper-limit of clamp, X5: lower-limit of clamp.

c) Y0: raise robot arm, Y1: lower robot arm, Y2: shift to right, Y3: shift to left, Y4: clip balls.

6. START circuit:

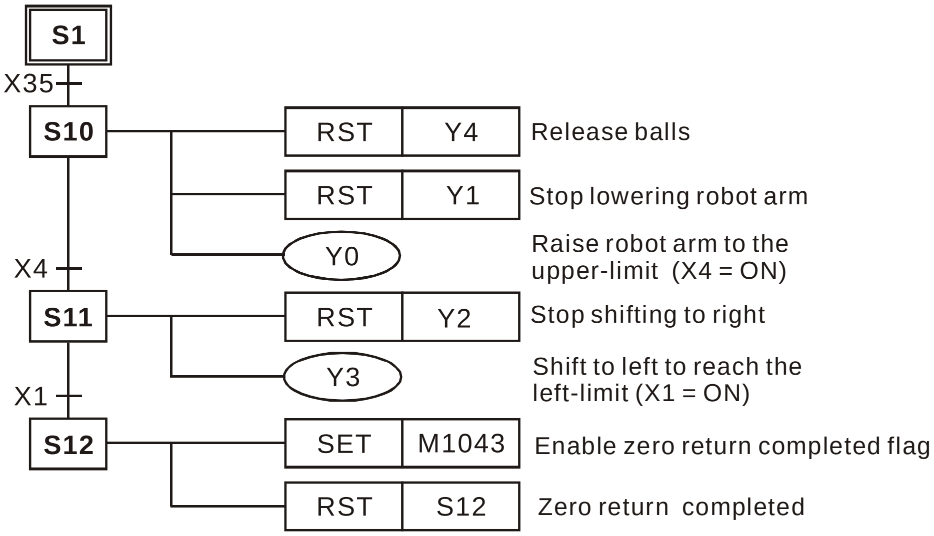


7. Manual mode:

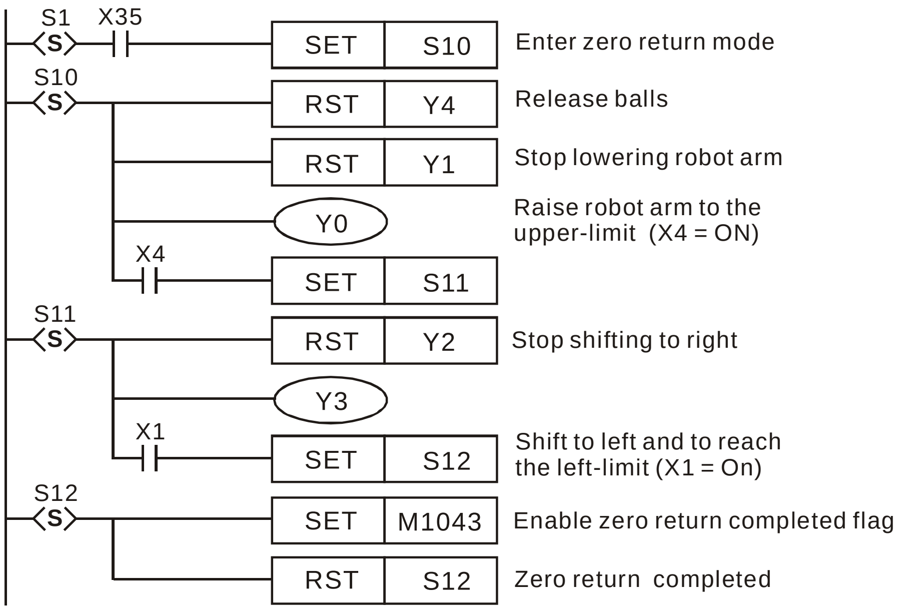


8. Zero return mode:

a) SFC:

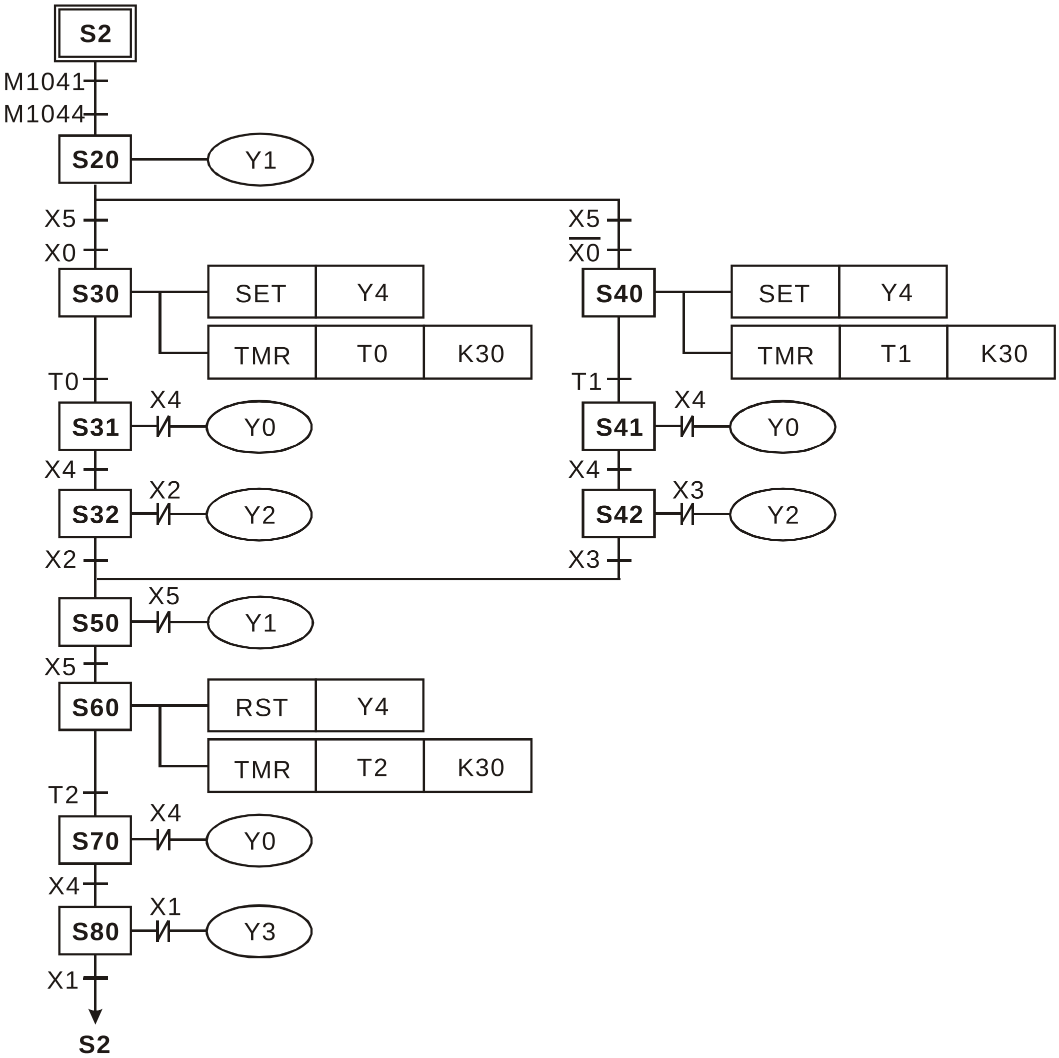


b) Ladder Diagram:

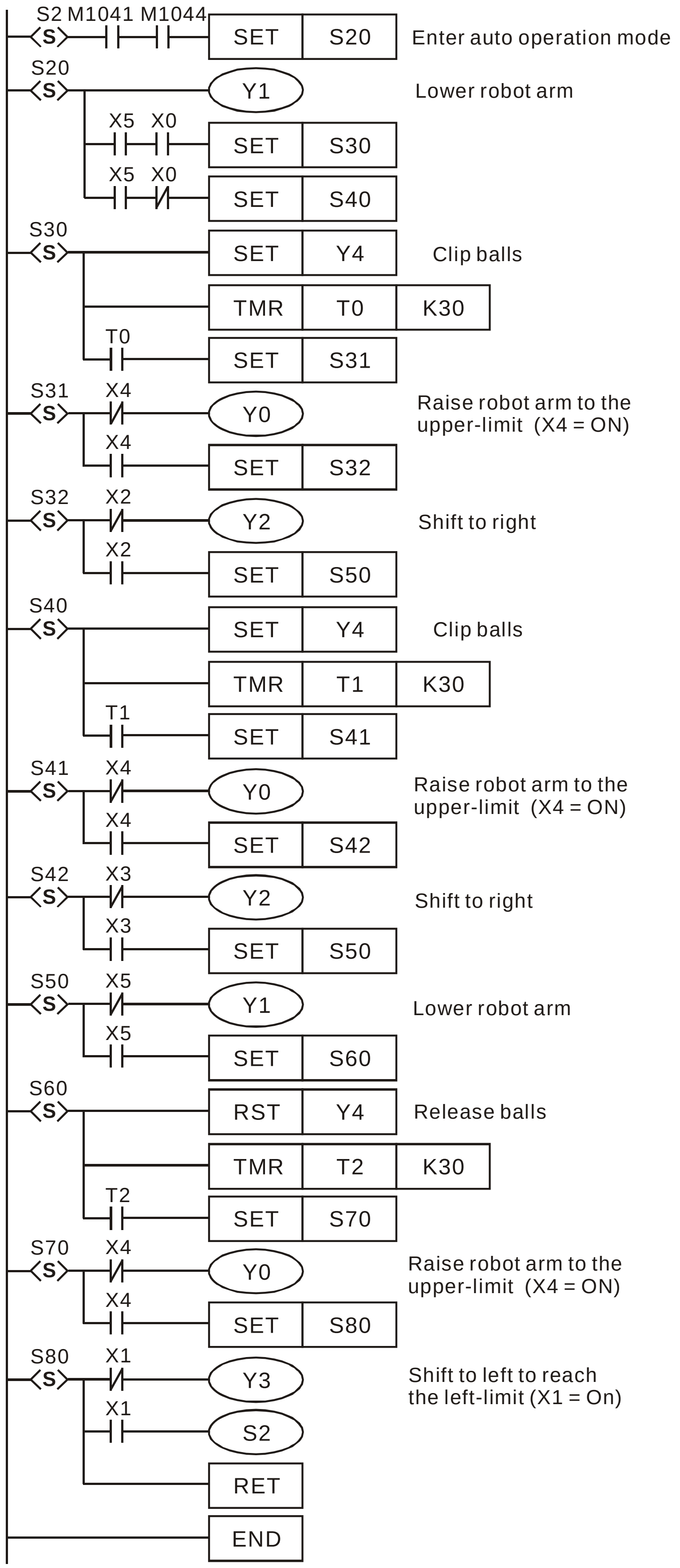


9. Auto operation (Single step / One-cycle operation / continuous operation):

a) SFC:



b) Ladder Diagram:



Troubleshooting

6

This chapter offers error code table and information for troubleshooting during PLC operation.

Table of Contents

6.1 Common Problems and Solutions

The following tables list some common problems and troubleshooting procedures for the PLC system in the event of faulty operation.

System Operation

Symptom	Troubleshooting and Corrective Actions
All LEDs are OFF.	1. Check the power supply wiring. 2. Check if the power supplied to the PLC control units is in the range of the rating. 3. Be sure to check the fluctuation in the power supply. 4. Disconnect the power supply wiring to the other devices if the power supplied to the PLC control unit is shared with them. If the LEDs on the PLC control unit turn ON at this moment, the capacity of the power supply is not enough to control other devices as well. Prepare another power supply for other devices or increase the capacity of the power supply. 5. If the POWER LED still does not light up when the power is on after the above corrective actions, the PLC should be sent back to the dealer or the distributor whom you purchased the product from.
The ERROR LED is ON.	When the execution of the program loop takes longer than the value set in D1000 (scan timeout), the ERROR LED of the PLC will be ON. Solutions: • Check the value in D1008 to see in which Step that a scan timeout occurred. Go to the specific Step to check if that is where the problem is. • Reduce the PLC program or use the instruction WDT to increase the scan timeout time. • If the ERROR LED is ON, download the PLC program and then turn the power of PLC OFF and then ON again to see if the ERROR LED is still ON. • If the ERROR LED is still ON, turn the power of PLC OFF and check if there is any noise interference or foreign object in the PLC.
The ERROR LED blinks rapidly every 0.2 seconds.	When the ERROR LED is blinking rapidly every 0.2, that indicates the built-in DC 24V is not sufficient. Check if the DC 24V power supply is overloaded. Refer to section 6.4 for detailed low voltage handling procedures.

Symptom	Troubleshooting and Corrective Actions
The ERROR LED blinks every 0.5 seconds.	Possible causes: • The downloaded PLC program contains invalid instructions or parameters. • The downloaded PLC program contains illegal program syntax or devices. Solutions: • Check the value in D1004 for the error code and the value in D1137 for the Step where an error occurred. And then check the troubleshoot for the error code in the next section.
The ERROR LED blinks every 0.2 seconds and waits for one second and then starts blinking 0.2 seconds again. The blinking pattern keeps going.	The SRAM (static random-access memory) of the PLC or other critical hardware component is out of order. The PLC CPU cannot read/write data, cannot execute any communication nor operation. Contact Delta technical support team or the distributors for further help.
Diagnosing Input Malfunction	When there are inputs but the input indicator LEDs are OFF, 1. Check the wiring of the input devices. 2. Check that the power is properly supplied to the input terminals. 3. If the power is properly supplied to the input terminal, there is probably an abnormality in the PLC's input circuit. Please contact your dealer. 4. If the power is not properly supplied to the input terminal, there is probably an abnormality in the input device or input power supply. Check the input device and input power supply. When there are no inputs but the input indicator LEDs are ON, 1. Monitor the input condition using a programming tool. If the input monitored is OFF, there is probably an abnormality in the PLC's input circuit. Please contact your dealer. 2. If the input monitored is ON, check the program again. Also, check the leakage current at the input devices (e.g., two-wire sensor) and check for the duplicated use of output or the program flow when a control instruction such as MC or CJ is used. 3. Check the settings of the I/O allocation.

Symptom	Troubleshooting and Corrective Actions
Diagnosing Output Malfunction	When there are no outputs, but the output indicator LEDs are ON, 1. Check the wiring of the loads. 2. Check if the power is properly supplied to the loads. 3. If the power is properly supplied to the load, there is probably an abnormality in the load. Check the load again. 4. If the power is not supplied to the load, there is probably an abnormality in the PLC's output circuit. Pleas contact your dealer. When there are outputs but the output indicator LEDs are OFF, 1. Monitor the output condition using a programming tool. If the output monitored is turned ON, there is probably a duplicated output error. 2. Forcing ON the output using a programming tool. If the output indicator LED is turned ON, go to input condition check. If the output LED remains OFF, there is probably an abnormality in the PLC's output circuit. Please contact your dealer.

6.2 Error code Table (Hex)

After you write the program into the PLC, the illegal use of operands (devices) or incorrect syntax in the program will result in flashing of ERROR indicator and M1004 = ON. In this case, you can find out the cause of the error by checking the error code (hex) in special register D1004. The address where the error occurs is stored in the data register D1137. If the error is a general loop error, the address stored in D1137 will be invalid.

Error code	Description	Solution
0001	Operand bit device S exceeds the valid range	Check D1137 (Error step number) Re-enter the instruction correctly
0002	Label P exceeds the valid range or duplicated	
0003	Operand KnSm exceeds the valid range	
0102	Interrupt pointer I exceeds the valid range or duplicated	
0202	Instruction MC exceeds the valid range	
0302	Instruction MCR exceeds the valid range	
0401	Operand bit device X exceeds the valid range	
0403	Operand KnXm exceeds the valid range	
0501	Operand bit device Y exceeds the valid range	
0503	Operand KnYm exceeds the valid range	
0601	Operand bit device T exceeds the valid range	
0604	Operand word device T register exceeds limit	
0801	Operand bit device M exceeds the valid range	
0803	Operand KnMm exceeds the valid range	
0B01	Operand K, H available range error	
0D01	DECO operand misuse	
0D02	ENCO operand misuse	
0D03	DHSCS operand misuse	
0D04	DHSCR operand misuse	
0D05	PLSY operand misuse	
0D06	PWM operand misuse	
0D07	FROM/TO operand misuse	
0D08	PID operand misuse	
0D09	SPD operand misuse	
0D0A	DHSZ operand misuse	
0D0B	IST operand misuse	
0E01	Operand bit device C exceeds the valid range	
0E04	Operand word device C register exceeds limit	
0E05	DCNT operand CXXX misuse	

Error code	Description	Solution
0E18	BCD conversion error	Check the D1137 (Error step number) Re-enter the instruction correctly
0E19	Division error (divisor=0)	
0E1A	Device use is out of range (including index registers E, F)	
0E1B	Negative number after radical expression	
0E1C	FROM/TO communication error	
0F04	Operand word device D register exceeds limit	
0F05	DCNT operand DXXX misuse	
0F06	SFTR operand misuse	
0F07	SFTL operand misuse	
0F08	REF operand misuse	
0F09	Improper use of operands of WSFR, WSFL instructions	
0F0A	Times of using TTMR, STMR instruction exceed the range	
0F0B	Times of using SORT instruction exceed the range	
0F0C	Times of using TKY instruction exceed the range	
0F0D	Times of using HKY instruction exceed the range	
1000	ZRST operand misuse	
10EF	E and F misuse operand or exceed the usage range	
2000	Usage exceed limit (MTR, ARWS, TTMR, PR, HOUR)	

Error code	Description	Solution
C400	An unrecognized instruction code is being used	A circuit error occurs if a combination of instructions is incorrectly specified. Select programming mode and correct the identified error.
C401	Loop Error	
C402	LD / LDI continuously use more than 9 times	
C403	MPS continuously use more than 9 times	
C404	FOR-NEXT exceed 6 levels	
C405	STL / RET used between FOR and NEXT SRET / IRET used between FOR and NEXT MC / MCR used between FOR and NEXT END / FEND used between FOR and NEXT	
C407	STL continuously use more than 9 times	
C408	Use MC / MCR in STL, Use I / P in STL	
C409	Use STL/RET in subroutine or interrupt program	
C40A	Use MC/MCR in subroutine Use MC/MCR in interrupt program	
C40B	MC / MCR does not begin from N0 or discontinuously	
C40C	MC / MCR corresponding value N is different	

Error code	Description	Solution
C40D	Use I / P incorrectly	
C40E	IRET doesn't follow by the last FEND instruction SRET doesn't follow by the last FEND instruction	
C40F	PLC program and data in parameters have not been initialized	
C41B	Invalid RUN/STOP instruction to extension module	
C41C	The number of input/output points of I/O extension unit is larger than the specified limit	
C41D	Number of extension modules exceeds the range	
C41F	Failing to write data into memory	
C440	Hardware error in high-speed counter	
C441	Hardware error in high-speed comparator	
C442	Hardware error in MCU pulse output	
C443	No response from extension unit	
C450	The analog-to-digital/digital-to-analog function of the MCU fails.	
C4EE	No END command in the program	
C4FF	Invalid instruction (no such instruction existing)	
C430	Error occurs while the left-side module is being initialized	Check if the module is connected securely? If there is any abnormality on the connection? If none of the above, replace with a new module.
C437	Error occurs while checking the memory of the left-side module	
C438	Error occurs while checking the model code of the left-side module	
C41C	The number of the extended X and Y points detected has exceed the supported quantity.	Make sure the number of the X and Y points of the extension DIO module is within the allowable range.
C421	The PLC identification code is not matched, when using the backup file from the memory card to restore the PLC program.	Make sure the PLC identification code in the backup file is the same as the PLC CPU has set.
C422	The PLC password is not matched, when using the backup file from the memory card to restore the PLC program.	Make sure the PLC password in the backup file is the same as the PLC CPU has set.
C425	The checksum or the PLC program syntax is not matched, when using the backup file from the memory card to restore the PLC program.	Make another backup file of the PLC program.

Error code	Description	Solution
C426	The PLC parameters in the backup file from the memory is destroyed and cannot be used to restore the PLC program.	Make another backup file of the PLC program.
C451	The program in the old firmware version is not encrypted. If upgrading to the new firmware version directly, an error will occur.	Restore to the default settings and Power-OFF the PLC CPU and then Power-ON again. After that upgrade to the new firmware.
C452	PLC program - MCU unique ID is not matched. Note: PLC checks if the program can be stored. This error indicates the program cannot be stored in the built-in memory.	Download the PLC program and then Power-OFF and Power-ON again. If this error persists, it means the PLC built-in memory is broken. Replace with a new PLC.
C453	PLC program - checksum error Note: It may indicate the PLC self-check area of the memory is broken or an unexpected connection loss occurred during on-line editing or any incident that prevents all programs to be updated.	
C461	The module is detected as abnormal. (noise interferences)	1. Check the module connection. 2. Replace with a new module.
C462		
C463	The number of X and Y points of the extended modules and the number of the special modules are not consistent between two module detections.	
C464	The module is detected as abnormal. (The number of X and Y points of the extended modules and the number of the special modules are not detected.)	
C465	The detected numbers of X and Y points of the extended modules and the number of the special modules are not consistent with the set values in D1248 (X points of extension module), D1254 (Y points of extension module) and D1385 (number of special modules).	1. Check the module connection. 2. Check the values set in D1248, D1254 and D1385.

6.3 Error Detection Devices

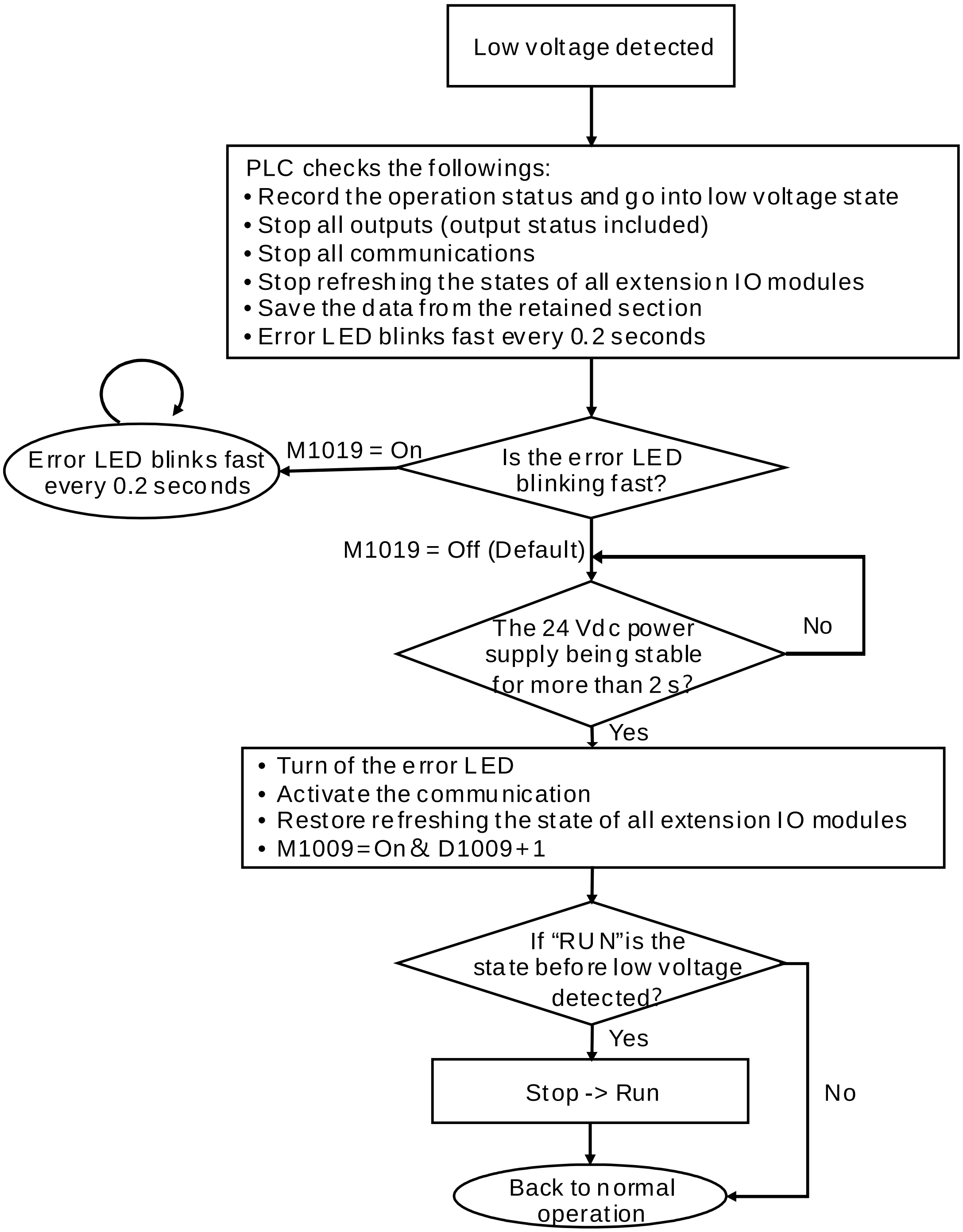
Error Check Devices	Description	Drop Latch	STOP → RUN	RUN → STOP
M1067	Program execution error flag	None	Reset	Latch
M1068	Execution error latch flag	None	Latch	Latch
D1067	Algorithm error code	None	Reset	Latch
D1068	Step value of algorithm errors	None	Latch	Latch

Device D1067 Error Code	Description
0E18	BCD conversion error
0E19	Division error (divisor=0)
0E1A	Floating point exceeds the usage range
0E1B	The value of square root is negative
0E1C	Instructions FROM/TO communication error

6.4 Low Voltage Handling Procedures

The low voltage means When the 24 Vdc power supply is lower than 17.8 V (error margin: approximately 2%) and lasts for more than 10 ms.

Once low voltage detected, the PLC CPU performs the following handling process flow:



CANopen Function and Operation 7

This chapter explains the functions of CANopen and the usage.

Table of Contents

7.1 The Introduction of CANopen

- Due to the simple wiring, immediate communication, strong debugging ability, stable communication, and low cost, the CANopen network is widely used in fields such as industrial automation, automotive industry, medical equipment industry, and building trade.
- The CAN port, which conforms to the basic communication protocol of CANopen DS301, is built in the PLC, can work in a master mode or a slave mode.
- This chapter explains the functions of CANopen. The functions are mainly controlled by the special auxiliary relay M1349. If M1349 is ON, the CANopen functions are enabled. If M1349 is OFF, the CANopen functions are disabled. In a master mode, the CANopen functions can support slave 1~slave 16.
- The CANopen network configuration software for DVP-ES2-C is CANopen Builder. The CANopen station address and the communication rate are set by means of this software. The programming software for DVP-ES2-C is WPLSoft or ISPSoft.
- This chapter mainly focuses on the CANopen functions. If users do not understand the professional terms mentioned in the introduction of the functions, they can refer to section 7.3 for more information.

7.1.1 The Description of the CANopen Functions

- **If the CAN port functions as a master, it has the following functions.**
  - It support the standard CANopen protocol DS301 V4.02.
  - It supports the NMT (network management object) service.
    - It supports the NMT state control.
      The NMT state control can be used to control the state of a slave in the CANopen network.
    - It supports the NMT error control.
      The NMT error control is used to detect the disconnection of a slave. The NMT error control can be classified into two types, i.e. Heartbeat and Node Guarding. The PLC supports Heartbeat, but do not support Node Guarding.
  - It supports the PDO (process data object) service.
    - The PDO message is used to transmit the immediate input data and output data.
    - It supports 64 RxPDO at most, and 390 bytes at most.
    - It supports 64 TxPDO at most, and 390 bytes at most.
    - The PDO transmission type: The synchronous mode, and the asynchronous mode
  - It supports the SDO (service data object) service.
    - The SDO can be used to read the parameter from a slave, write the parameter into a slave, or configure the parameter for a slave.
    - It supports the standard SDO transmission mode.
    - It supports the automatic SDO functions. Twenty pieces of data at most can be written into a slave.
    - It supports the use of the SDO service in a PLC ladder diagram to read the data from a slave or write the data into a slave.
  - It supports the service of reading the emergency from a slave.
    - The service of reading the emergency from a slave can be used to read an error or an alarm from a slave.
    - Five emergencies can be stored in a slave.
    - The emergency can be read through a PLC ladder diagram.
  - It supports the SYNC object (synchronous object) service.
    Several devices can operate synchronously through the synchronous object service
  - The CANopen communication rates which are supported are 20K, 50K, 125K, 250K, 500K, 1Mbps.
  - The mapping data types which are supported:

Storage	Data type
8-bit	SINT USINT BYTE
16-bit	INT UINT WORD
32-bit	DINT UDINT REAL DWORD
64-bit	LINT ULINT LREAL LWORD

- **If the CAN port functions as a slave, it has the following functions.**
  - It supports the standard CANopen protocol DS301 V4.02.
  - It supports the NMT (network management object) service.
    - It supports the NMT state control.
      The state of DVP-ES2-C in the CANopen network is controlled by a master.
    - It supports the NMT error control.
      Heartbeat is supported, but Node Guarding is not supported.
  - It supports the PDO (process data object) service.
    - The PDO message is used to transmit the immediate input data and output data.
    - It supports 8 TxPDO at most, and 8 RxPDO at most.
    - The PDO transmission type: The synchronous mode, and the asynchronous mode
  - It supports the emergency service.
    If an error or an alarm occurs in DVP-ES2-C, the master is notified through the emergency.

7.1.2 The Input/Output Mapping Areas

DVP-ES2-C as a master supports 16 slaves at most, and the slave node ID range from 1 to 16. The output mapping areas are D6250-D6476, and the input mapping areas are D6000-D6226.

Device in the PLC	Mapping area	Mapping length
D6250~D6281	SDO request information, NMT service information, and Emergency request information	64 bytes
D6000~D6031	SDO reply information, and Emergency reply information	64 bytes
D6282~D6476	RxPDO mapping area	390 bytes
D6032~D6226	TxPDO mapping area	390 bytes

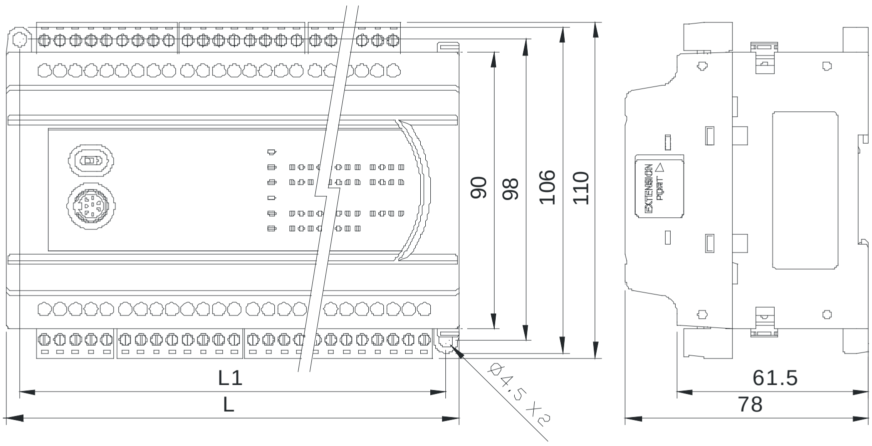
If DVP-ES2-C functions as a slave station, the output mapping areas are D6282-D6313, and the input mapping areas are D6032-D6063.

Device in the PLC	Mapping area	Mapping length
D6032~D6063	RxPDO mapping area	64 bytes
D6282~D6313	TxPDO mapping area	64 bytes

7.2 The Installation and the Network Topology

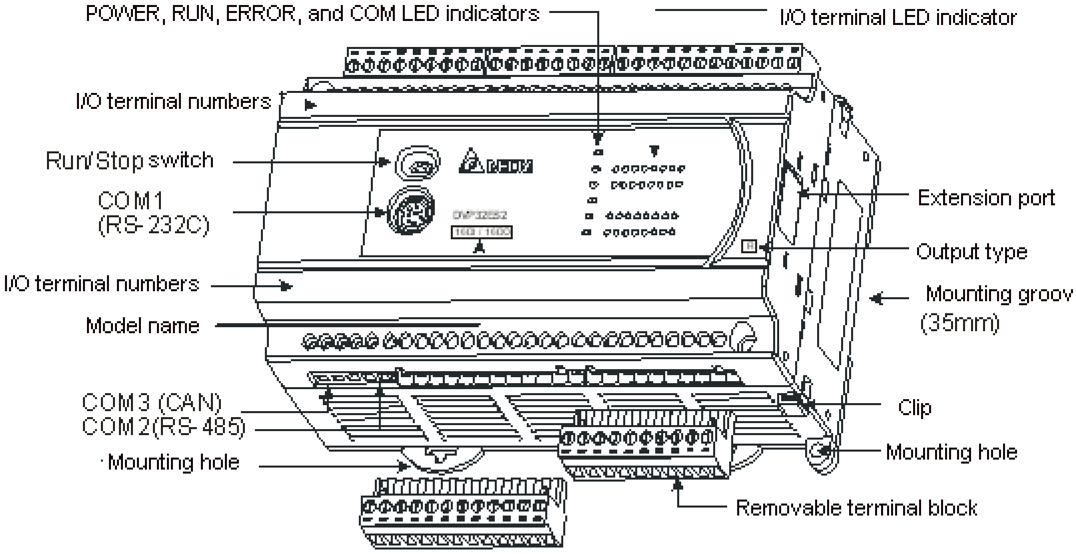
This section introduces the dimensions of DVP-ES2-C, the CAN interface, the CANopen network framework, and the communication distance.

7.2.1 The Dimensions



Unit: millimeter

7.2.2 The Profile

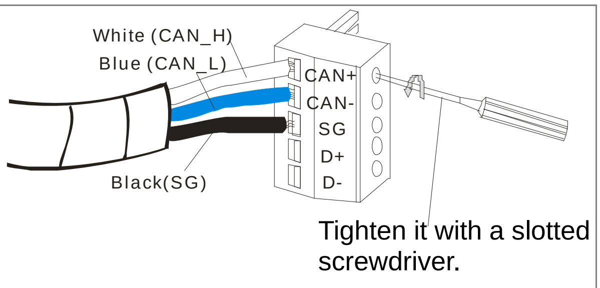


7

7.2.3 The CAN Interface and the Network Topology

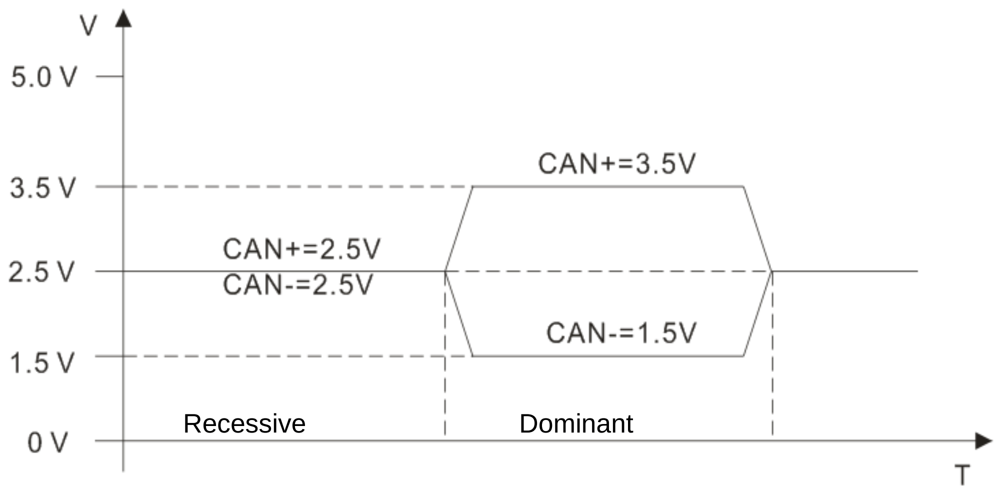
➢ **The pins of COM3 (CAN interface)**

Pin	Description
CAN+	CAN-H
CAN-	CAN-L
SG	Signal ground

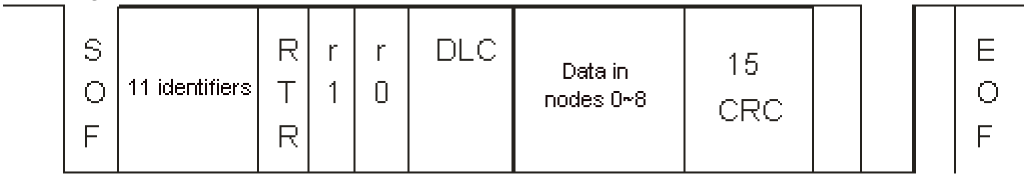


➢ **The CAN signal and the data frame format**

The CAN signal is a differential signal. The voltage of the signal is the voltage difference between CAN+ and CAN-. The voltage of CAN+ and that of CAN- take SG as a reference point. The CAN network can be in two states. One is a dominant level, and is indicated by the logical “0”. The other is a recessive level, and is indicated by the logical “1”. The CAN signal level is shown below.

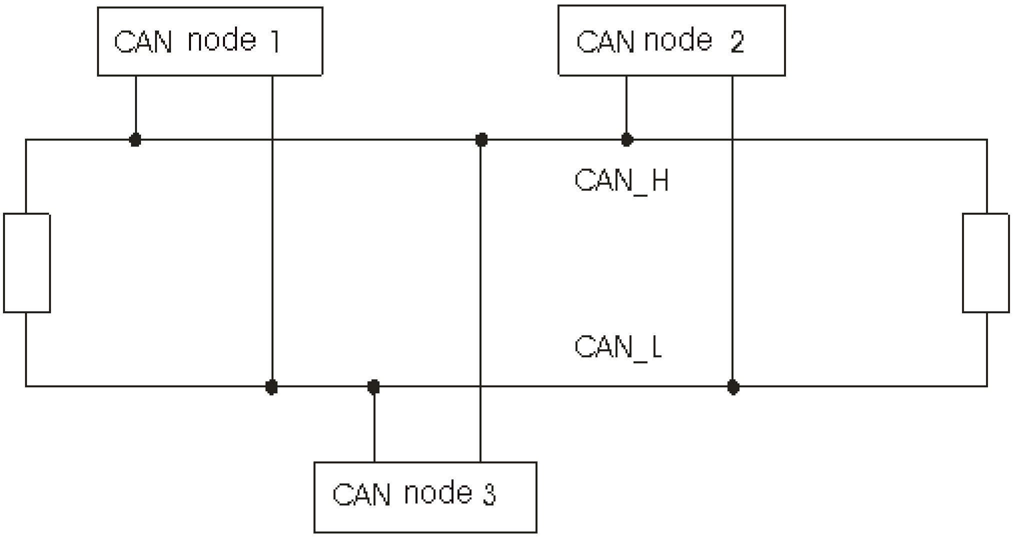


The data frame format is shown below. The CAN nodes transmit the CAN messages to the network from left to right, as the data frame format below shows.

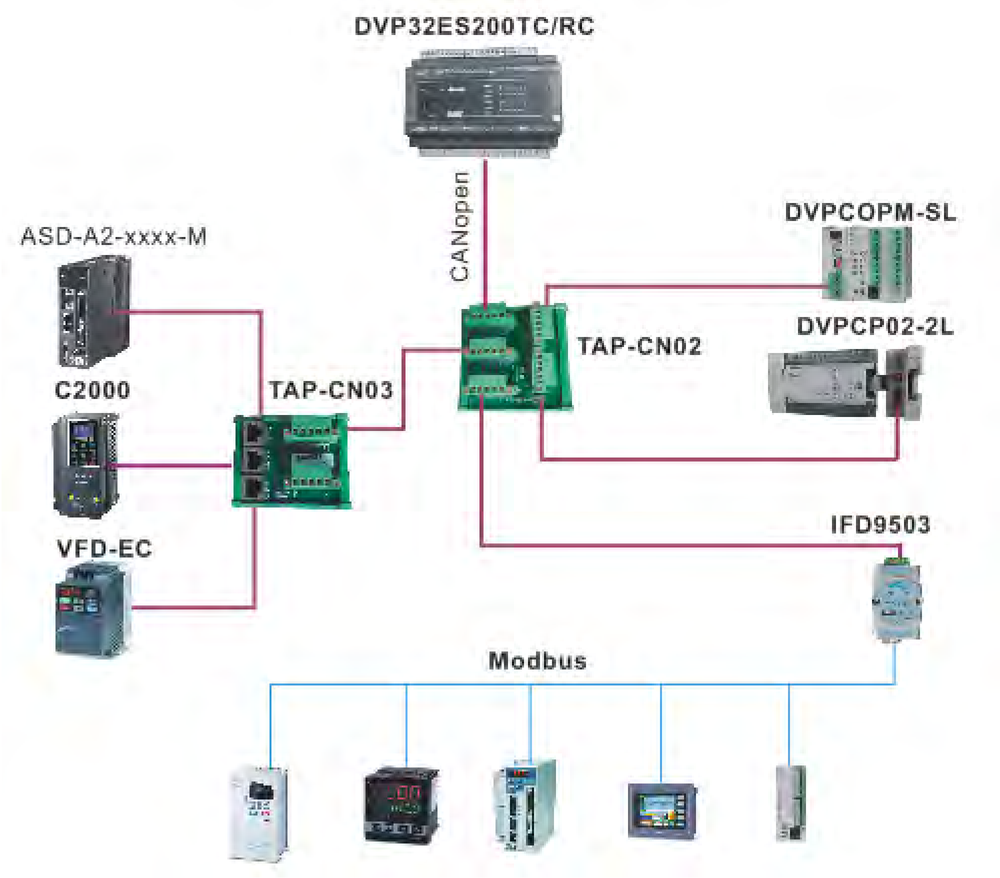


➢ **The CAN network endpoint and the topology structure**

In order to make the CAN communication more stable, the two endpoints of the CAN network are connected to 120 ohm terminal resistors. The topology structure of the CAN network is illustrated below.



➢ **The topology structure of the CANopen network**



1) Users should use standard Delta cables when creating the CANopen network. These cables are the thick cable UC-DN01Z-01A (TAP-CB01), the thin cable UC-DN01Z-02A (TAP-CB02), and the thin cable UC-CMC010-01A (TAP-CB10). The communication cables should be away from the power cables.
2) TAP-TR01. CAN+ and CAN-, which are at the endpoints of the network, should be connected to 120 ohm resistors. Users can purchase the standard Delta terminal resistor TAP-TR01.
3) The limitation on the length of the CANopen network
   The transmission distance of the CANopen network depends on the transmission rate of the CANopen network. The relation between the transmission rate and the maximum communication distance is shown in the following table.

Transmission rate (bit/second)	20K	50K	125K	250K	500K	1M
Maximum communication distance (meter)	2500	1000	500	250	100	25

4) The Delta network products related to the CANopen network are listed below.

Product	Model	Function
	DVP32ES200RC DVP32ES200TC	It is a DVP-ES2-C series PLC with the built-in CAN interface. It can function as the CANopne master or slave.

Product	Model	Function
	DVPCOPM-SL	DVPCOPM-SL is a module connected to the left side of an S series PLC. It can function as the CANopen master or slave. The PLCs which can be connected to DVPCOPM-SL are DVP-28SV, DVP-28SV2, DVP-SX2, DVP-SA2, and DVP-EH2-L.
	IFD9503	It converts CANopen to the Modbus gateway, and connects the device (with the RS-232 or RS-485 interface) which conforms to the standard Modbus protocol to the CANopen network. 15 devices at most can be connected.
	DVPCP02-H2	It is the CANopen slave module, and is connected to the right side of an EH2 series PLC. It can connect the EH2 series PLC to the CANopen network.
	IFD6503	It is a tool used to analyze the CANopen network data. The interfaces at both ends are the CAN interface and the USB interface. It can be used to catch the CAN network data, or allow the CAN nodes to transmit the data. The product is used with the software Netview Builder.
	ASD-A2-xxxx-M servo driver	It is a servo driver with the built-in CANopen interface. It controls the positioning, speed, and torque.
	C2000/CP2000/C200 series AC motor drives	It is an AC motor drive with the built-in CANopen function, and controls the positioning, speed, and torque. Before using the CANopne function of the C2000/CP2000 series AC motor drives, users need to purchase CMC-COP01. This card only provides the CAN interface. The C200 series AC motor drive has the built-in CANopen interface.

Product	Model	Function
	EC series AC motor drive	The EC series AC motor drive has the built-in CANopen interface. It controls the speed and torque.
	TAP-CN01	It is the CANopen network topology distribution box which carries a 120 ohm resistor. Users can enable the resistor through the switch.
	TAP-CN02	It is the CANopen network topology distribution box which carries a 120 ohm resistor. Users can enable the resistor through the switch.
	TAP-CN03	It is the CANopen network topology distribution box which carries a 120 ohm resistor. Users can enable the resistor through the switch.
	UC-CMC003-01A (TAP-CB03), UC-CMC005-01A (TAP-CB05), UC-CMC010-01A (TAP-CB10), UC-CMC020-01A (TAP-CB20)	CANopen sub cable with RJ45 connectors at both ends. UC-CMC003-01A (TAP-CB03): 0.3 meter UC-CMC005-01A (TAP-CB05): 0.5 meter UC-CMC010-01A (TAP-CB10): 1 meter UC-CMC020-01A (TAP-CB20): 2 meter
	UC-DN01Z-01A (TAP-CB01), UC-DN01Z-02A (TAP-CB02)	CANopen network cable UC-DN01Z-01A (TAP-CB01): CANopen main cable UC-DN01Z-02A (TAP-CB02): CANopen sub cable
	TAP-TR01	It is a 120 ohm resistor with a RJ45 connector.

7.3 The CANopen Protocol

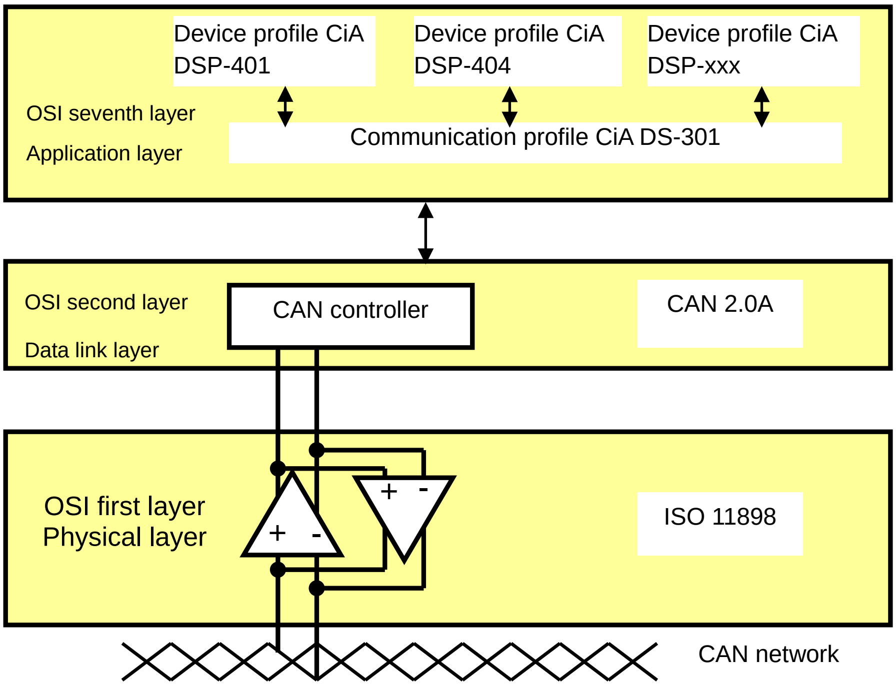
7.3.1 The Introduction of the CANopen Protocol

The CAN (controller area network) fieldbus only defines the physical layer and the data link layer. (See the ISO11898 standard.) It does not define the application layer. In the practical design, the physical layer and the data link layer are realized by the hardware. The CAN fieldbus itself is not complete. It needs the superior protocol to define the use of 11/29-bit identifier and that of 8-byte-data.

The CANopen protocol is the superior protocol base on CAN. It is one of the protocols defined and maintained by CiA (CAN-in-Automation). It is developed on the basis of the CAL (CAN application layer) protocol, using a subset of the CAL communication and service protocols.

The CANopen protocol covers the application layer and the communication profile (CiA DS301). It also covers a framework for programmable devices (CiA 302), the recommendations for cables and connectors (CiA 303-1), and SI units and prefix representations (CiA 303-2).

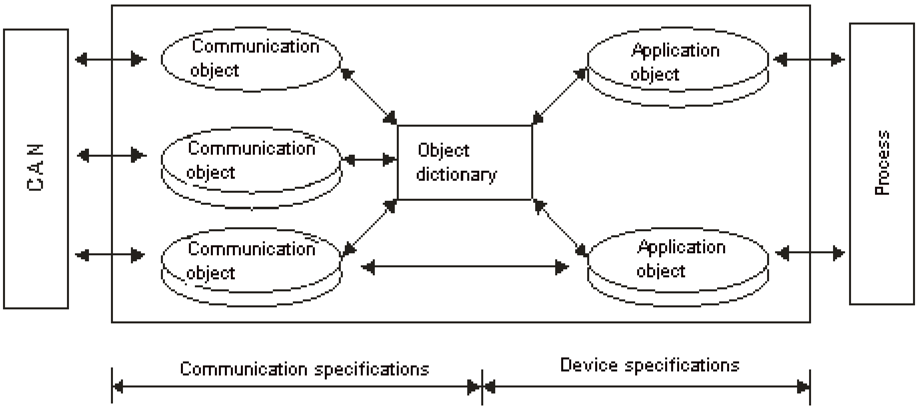
In the OSI model, the relation between the CAN standard and the CANopen protocol is as follow.



- **The object dictionary**

  CANopen uses an object-based way to define a standard device. Every device is represented by a set of objects, and can be visited by the network. The model of the CANopen device is illustrated below. As the figure below shows, the object dictionary is the interface between the communication program and the superior application program.

  The core concept of CANopen is the device object dictionary (OD). It is an orderly object set. Every object adopts a 16-bit index for addressing. In order allow the visit to the single element in the data structure, it also defines, an 8-bit subindex. Every node in the CANopen network has an object dictionary. The object dictionary includes the parameters which describe the device and the network behavior. The object dictionary of a node is described in the electronic data sheet (EDS).



7.3.2 The CANopen Communication Object

The CANopen communication protocol contains the following communication objects.

➢ **PDO (process data object)**

✧ The PDO provides the direct visit channel for the device application object, is used to transmit the real-time data, and has high priority. Every byte in the PDO CAN message data list is used to transmit the data. The rate of making use of the message is high.

✧ There are two kinds of uses for PDOs. The first is data transmission and the second data reception. They are distinguished by Transmit-PDOs (TxPDOs) and Receive-PDOs (RxPDOs). Devices supporting TxPDOs are PDO producers, and devices which are able to receive PDOs are called PDO consumers.

✧ The PDO is described by means of the “producer/consumer mode”. The data is transmitted from one producer to one or many consumers. The data which can be transmitted are limited to 1-byte data to 8-byte data. After the data is transmitted by the producer, the consumer does not need to reply to the data. Every node in the network will detect the data information transmitted by the transmission node, and decides whether to process the data which is received.

✧ Every PDO is described by two objects in the object dictionary: The PDO communication parameters and the PDO mapping parameters

The PDO communication parameters: The COB-ID which will be used by PDO, the transmission type, the prohibition time, and the cycle of the counter

The PDO mapping parameters: They include the object list in an object dictionary. These objects are mapped into the PDO, including the data length (in bits). To explain the contents of the PDO, the producer and the consumer have to understand the mapping.

✧ The PDO transmission mode: synchronous and asynchronous

Synchronous: Synchronous periodic and synchronous non-periodic

Asynchronous: The PDO is transmitted when the data changes, or it is transmitted after a trigger.

The transmission modes supported by are as follows.

Type	**PDO transmission**				
	Periodic	Non-periodic	Synchronous	Asynchronous	RTR
0		X	X		
1 – 240	X		X		
254				X	
255				X	

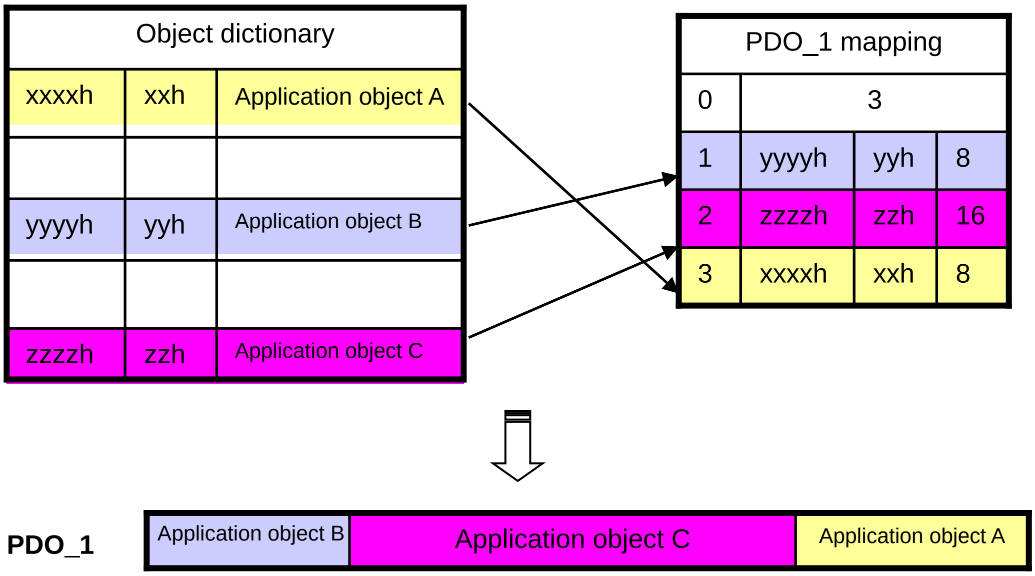
Mode 0: The PDO information is transmitted only when the PDO data changes and the synchronous signal comes.

Modes 1~240: One piece of PDO information is transmitted every 1~240 synchronous signals.

Mode 254: The trigger is defined the manufacturer. The definition of the PLC is the same as mode 255.

Mode 255: PDO is transmitted when the data changes, or it is transmitted after a trigger.

All the data in the PDO has to be mapped from the object dictionary. The following is an example of the PDO mapping.



The data format for RxPDO and TxPDO is as follows.

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Object identifier	Data							

- **SDO (service data object)**
  - The SDO is used to build the client/server relation between two CANopen devices. The client device can read the data from the object dictionary of the server device, and write the data into the object dictionary of the server device. The visit mode of the SDO is “client/server” mode. The mode which is visited is the SDO server. Every CANopen device has at least one service data object which provides the visit channel for the object dictionary of the device. SDO can read all objects in the object dictionary, and write all objects into the object dictionary.
  - The SDO message contains the index information and the subindex information which can be used to position the objects in the object dictionary, and the composite data structure can easily pass the SDO visit. After the SDO client sends the reading/writing request, the SDO server replies. The client and the server can stop the transmission of the SDO .The requested message and the reply message are divided by different COB-IDs.
  - The SDO can transmit the data in any length. If the data length is more than 4 bytes, the data has to be transmitted by segment. The last segment of the data contains an end flag.
  - The structures of the SDO requested message and reply message are as follows.
    The format of the requested message:

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
600（hex）+Node-ID	Requested code	Object index		Object subindex	Requested data			
		LSB	MSB		bit7-0	bit15-8	bit23-16	bit31-24

The definition of the requested code in the requested message:

Request code (hex)	Description
23	Writing the 4-byte data
2B	Writing the 2-byte data
2F	Writing the 1-byte data
40	Reading the data
80	Stopping the current SDO function

The format of the reply message:

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
580（hex）+Node-ID	Reply code	Object index		Object subindex	Reply data			
		LSB	MSB		bit7-0	bit15-8	bit23-16	bit31-24

The definition of the reply code in the reply message:

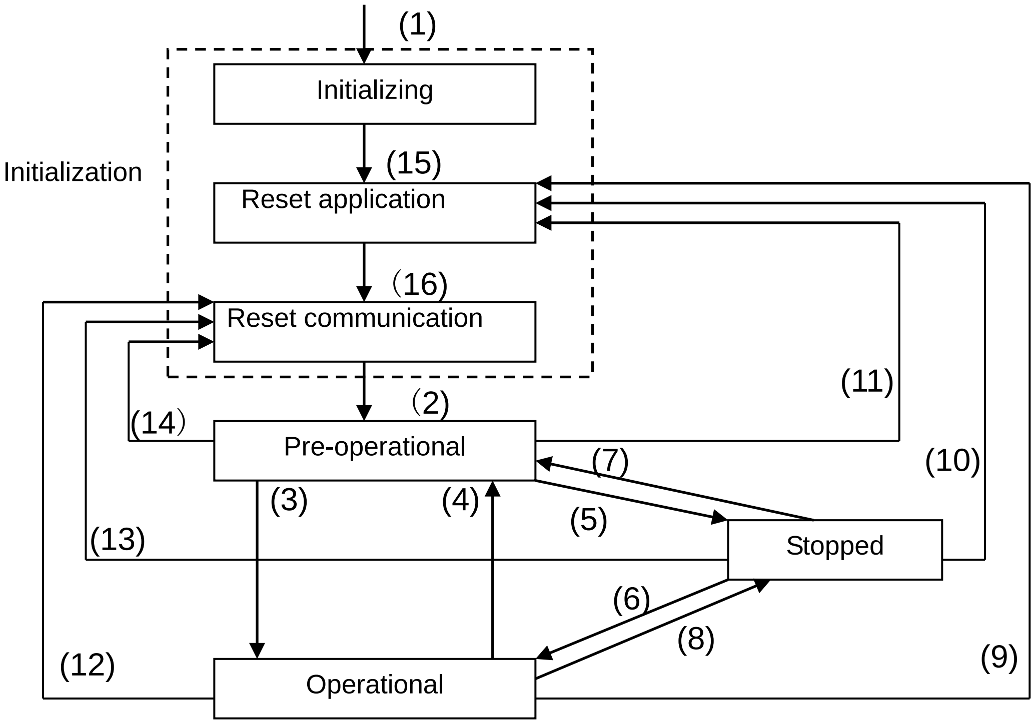
Reply code (hex)	Description
43	Reading the 4-byte data
4B	Reading the 2-byte data
4F	Reading the 1-byte data
60	Writing the 1/2/4-byte data
80	Stopping the SDO function

➢ **NMT (network management object)**

The CANopen network management conforms to the “master/slave” mode. Only one NMT master exists in the CANopen network, and other nodes are considered slaves. NMT realized three services. They are module control services, error control services, and boot-up services.

✧ **Module control services**

The master node in the CANopen network controls the slave by sending the command. The slave executes the command after it received the command. It does not need to reply. All CANopen nodes have internal NMT states. The slave node has four states. They are the initialization state, the pre-operational state, the operational state, and the stop state. The state of the device is illustrated below.



(1) After the power is supplied, the device automatically enters the initialization state.
(2) After the initialization is complete, the device automatically enters the Pre-operational state.
(3)(6) The remote node is started.
(4)(7) The device enters the Pre-operational state.
(5)(8) The remote node is stopped.
(9)(10)(11) The application layer is rest.
(12)(13)(14) The communication is reset.
(15) After the initializing is complete, the device automatically enters the “reset application” state.
(16) After the “reset application” state is complete, the device automatically enters the

"reset communication" state.
The relation between the communication object and the state is shown below. The communication object service can be executed only in a proper state. For example, SDO can be executed only in the operational state and in the pre-operational state.

	Initialization	Pre-operational	Operational	Stopped
PDO			X	
SDO		X	X	
SYNC		X	X	
Time Stamp		X	X	
EMCY		X	X	
Boot-up	X			
NMT		X	X	X

The format of the control message for the node state:

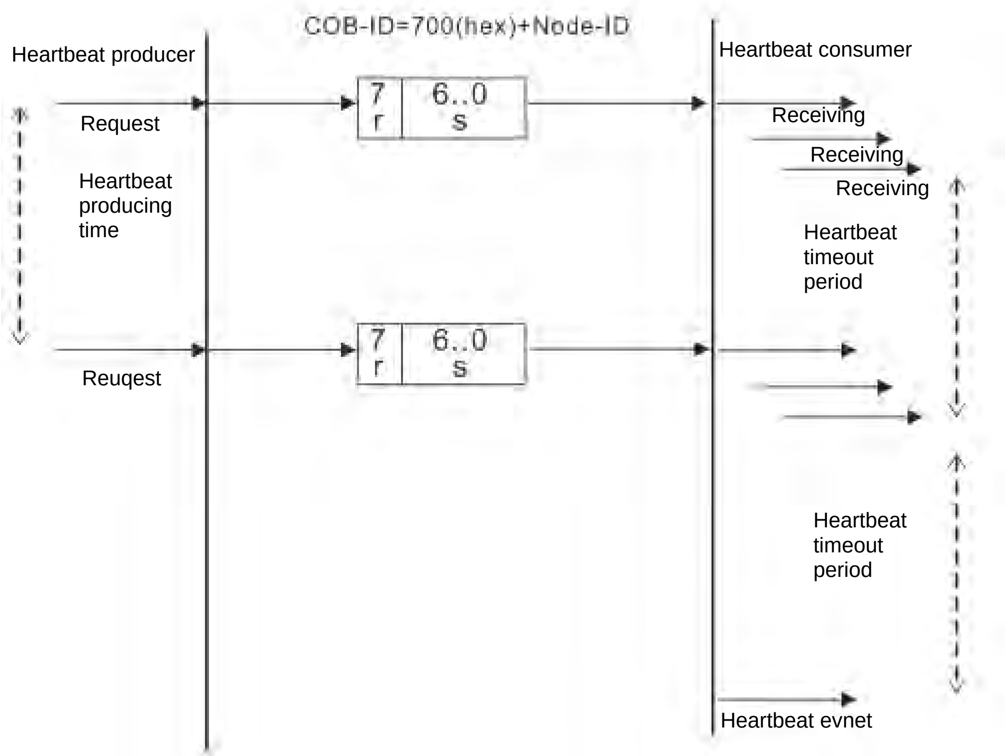
COB-ID	Byte 0	Byte 1
0	Command specifier (CS)	Slave address (0: Broadcast)

The command specifiers are listed below.

Command specifier (hex)	Function
01	Start the remote node
02	Stop the remote node
80	Enter the pre-operational state
81	Reset the application layer
82	Reset the communication

✧ **Error control services**

The error control service is used to detect the disconnection of the node in the network. The error control services can be classified into two types, i.e. Heartbeat and Node Guarding. The PLC only supports Heartbeat. For example, the master can detect the disconnection of the slave only after the slave enables the Heartbeat service.
The Heartbeat principle is illustrated as follows. The Hearbeat producer transmits the Heartbeat message according to the Heartbeat producing time which is set. One or many Heartbeat consumers detect the message transmitted by the Heartbeat producer. If the consumer does not receive the message transmitted by the producer within the timeout period, the CANopen communication is abnormal.



7

✧ **Boot-up services**
After the slave completes the initialization and enters the pre-operational state, it transmits the Boot-up message.

➢ **Other predefined CANopen communication objects (SYNC and EMCY)**

✧ **SYNC Object (Synchronous object)**
The synchronous object is the message broadcasted periodically by the master node in the CANopen network. This object is used to realize the network clock signal. Every device decides whether to use the event and undertake the synchronous communication with other network devices according to its configuration. For example, when controlling the driving device, the devices do not act immediately after they receive the command sent by the master. They do act until they receive the synchronous message. In this way, many devices can act synchronously.

The format of the SYNC message:

COB-ID
80 (hex)

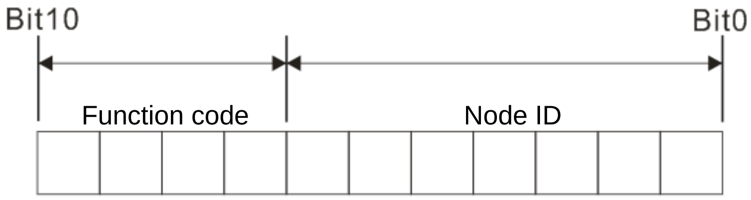
✧ **Emergency object**
The emergency object is used by the CANopen device to indicate an internal error. When an emergency error occurs in the device, the device sent the emergency message (including the emergency error code), and the device enters the error state. After the error is eliminated, the device sends the emergency message, the emergency error code is 0, and the device enters the normal state.
The format of the emergency message:

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
80 (hex) +Node-ID	Emergency error code		Error register	Factory-defined error code				
	LSB	MSB						

Note: The value in the error register is mapped to index 1001 (hex) in the object dictionary. If the value is 0, no error occurs. If the value is 1, a normal error occurs. If the value is H’80, an internal error occurs in the device.

7.3.3 The Predefined Connection Set

In order to decrease the configuration workload of the network, CANopen defines a default identifier. In the predefine connection set, the structure of the 11-bit identifier is as follows.



The objects which are supported and the COB-IDs which are assigned to the objects are listed below.

- The broadcast object in the predefined connection setting

Object	Function code	COB-ID	Index of the communication parameter
NMT	0000	0	-
SYNC	0001	128 (80h)	1005h, 1006h, 1007h
Time stamp	0010	256 (100h)	1012h, 1013h

- The corresponding object in the predefined connection set

Object	Function code	COB-ID	Index of the communication parameter
Emergency	0001	129 (81h)–255 (FFh)	1014h, 1015h
PDO1 (TX)	0011	385 (181h)–511 (1FFh)	1800h
PDO1 (RX)	0100	513 (201h)–639 (27Fh)	1400h
PDO2 (TX)	0101	641 (281h)–767 (2FFh)	1801h
PDO2 (RX)	0110	769 (301h)–895 (37Fh)	1401h
PDO3 (TX)	0111	879 (381h)–1023 (3FFh)	1802h
PDO3 (RX)	1000	1025 (401h)–1151 (47Fh)	1402h
PDO4 (TX)	1001	1153 (481h)–1279 (4FFh)	1803h
PDO4 (RX)	1010	1281 (501h)–1407 (57Fh)	1403h
SDO (TX)	1011	1409 (581h)–1535 (5FFh)	1200h
SDO (RX)	1100	1537 (601h)–1663 (67Fh)	1200h
NMT Error Control	1110	1793 (701h)–1919 (77Fh)	1016h, 1017h

7.4 Sending SDO, NMT and Reading Emergency Message through the Ladder Diagram

Editing the request message mapping area can realize the transmission of SDO, NMT and Emergency message.The corresponding relations between the request message mapping area, response message mapping area and PLC device are shown below.

PLC device	Mapping area	Mapping length
D6250~D6281	SDO request message, NMT service message and Emergency request message	64 bytes
D6000~D6031	SDO response message and Emergency response message	64 bytes

1> CANopen master can only send one SDO, NMT or Emergency request message to the same equipment at a time.
2> We suggest the request message mapping area should be cleared to zero when sending SDO, NMT or Emergency request message through WPL program.

7.4.1 Data Structure of SDO Request Message

Sending SDO through the ladder diagram can read or write the slave parameter.

➢ The data format of the SDO request message:

PLC device	Request message		
		High byte	Low byte
D6250	Message Header	ReqID	Command (Fixed to 01)
D6251		Reserved	Size
D6252		Type	Node ID
D6253	Message Data	High byte of main index	Low byte of main index
D6254		Reserved	Sub-index
D6255		Datum 1	Datum 0
D6256		Datum 3	Datum 2
D6257 ~ D6281		Reserved	

- ✧ Command: Fixed to “01”.
- ✧ ReqID: The request ID. Whenever an SDO request message is sent out, the message will be given a ReqID for CANopen master to identify. When reading/writing another SDO message, the original ID number must be changed. In other words, to read/write SDO is triggered by changing of the value of “ReqID”. ReqID range: 00 (Hex) ~ FF (Hex).
- ✧ Size: The length of the message data. The counting starts from D6253 with byte as the unit. When reading, it is fixed to 4 and when writing, it is 4 plus the byte number of data types of index and subindex and the maximum value is 8. But when writing, if the data type of index and subindex is word, the data length is 6 or it is 5 if byte.
- ✧ Node ID: The node address of the target equipment on CANopen network.
- ✧ Type: 01 indicates the read access; 02 indicates the write access.

➢ The data format of the SDO response message:

PLC device	Response message		
		High byte	Low byte
D6000	Message Header	ResID	Status code
D6001		Reserved	Size
D6002		Type	Node ID
D6003	Message Data	High byte of main index	Low byte of main index
D6004		Reserved	Sub-index
D6005		Datum 1	Datum 0
D6006		Datum 3	Datum 2
D6007 ~ D6031		Reserved	

- ✧ Status code:
  The indication of the status code values in the response message:

Status code	Explanation
0	No data transmission request
1	SDO message transmission succeeds.
2	SDO message is being transmitted.
3	Error: SDO transmission time-out
4	Error: Illegal command code
5	Error: the length of the transmitted data is illegal.
6	Error: the length of the response data is illegal.

Status code	Explanation
7	Error: Equipment to be sent messages is busy.
8	Error: Illegal type
9	Error: Incorrect node address
0A	Error message (See the error code for SDO response message)
0B~FF	Reserved

- ResID: Same as the request ID in the request message.
- Size: The length of the message data. Max. 20 bytes. Unit: byte. When writing, it is 4; the data length is decided by the data type of index and subindex when reading.
- Node ID: The node address of the target equipment on CANopen network.
- Type: In SDO response message, 43 (Hex) refers to reading 4 bytes of data; 4B (Hex) refers to reading 2 bytes of data; 4F (Hex) refers to reading 1 byte of data; 60 (Hex) refers to writing 1/2/4 byte(s) of data; 80 (Hex) refers to stopping SDO command.

Example 1: Write 010203E8 (hex) to (Index_subindex) 2109_0 of slave of No. 3 through SDO and the data type of (Index_subindex) 2109_0 is double words (32 bits).

- Request data:

PLC device	Request message		
		High byte(Hex)	Low byte(Hex)
D6250	Message Header	ReqID=01	Command =01
D6251		Reserved =0	Size =8
D6252		Type =02	Node ID =03
D6253	Message data	Main index high byte =21	Main index low byte =09
D6254		Reserved =0	Subindex =0
D6255		Datum 1=03	Datum 0=E8
D6256		Datum 3=01	Datum 2=02

- Response data:

PLC device	Response message		
		High byte(Hex)	Low byte(Hex)
D6000	Message Header	ResID =01	Command =01
D6001		Reserved =0	Size =4
D6002		Type =60	Node ID =03
D6003	Message data	Main index high byte =21	Main index low byte =09
D6004		Reserved =0	Subindex =0
D6005		Datum 1=00	Datum 0=00
D6006		Datum 3=00	Datum 2=00

Example 2: Read the value of (Index_subindex) 2109_0 of slave of No. 3 through SDO and the data type of (Index_subindex) 2109_0 is double words (32 bits).

- Request data:

PLC device	Request message		
		High byte(Hex)	Low byte(Hex)
D6250	Message Header	ReqID =01	Command =01
D6251		Reserved =0	Size =4
D6252		Type =01	Node ID =03
D6253	Message data	Main index high byte =21	Main index low byte =09
D6254		Reserved =0	Subindex =0
D6255		Datum 1=0	Datum 0=0
D6256		Datum 3=0	Datum 2=0

◆ Response data:

PLC device	Response message		
		High byte(Hex)	Low byte(Hex)
D6000	Message Header	ResID =01	Command =01
D6001		Reserved =0	Size =8
D6002		Type =43	Node ID =03
D6003	Message data	Main index high byte =21	Main index low byte =09
D6004		Reserved =0	Subindex =0
D6005		Datum 1=03	Datum 0=E8
D6006		Datum 3=01	Datum 2=02

7.4.2 Data Structure of NMT Message

NMT service can be used managing the CANopen network such as start, operation, reset of nodes and etc.

➢ The data format of the NMT request message:

PLC device	Request message		
		High byte	Low byte
D6250	Message Header	ReqID	Command (Fixed to 01)
D6251		Reserved	Size (Fixed to 04)
D6252		Type (Fixed to 03)	Node ID
D6253	Message data	Reserved	NMT service code
D6254		Reserved	Node ID

✧ Command: Fixed to 01.

✧ ReqID: The request ID. Whenever an NMT request message is sent out, the message will be given a ReqID for the CANopen master to identify. Before another NMT request message is sent out, the original ID number must be changed. In other words, to send out the NMT request message is triggered by changing of the value of “ReqID”. ReqID range: 00 (Hex) ~ FF (Hex).

✧ Node ID: The node address of the target equipment on CANopen network. (0: Broadcast)

✧ NMT service code:

NMT service code (Hex)	Function
01	Start remote node
02	Stop remote node
80	Enter the pre-operational state
81	Reset application
82	Reset communication

➢ The data format of the NMT Response message:

PLC device	Response message		
		High byte	Low byte
D6000	Message header	ResID	Status code
D6001		Reserved	Reserved
D6002		Reserved	Node ID

✧ When status code is 1, it indicates that NMT operation succeeds. When status code is not equal to1, it indicates that NMT operation fails and in the meantime, you should check if the data in NMT request message are correct.

✧ Node ID: The node address of the target equipment on CANopen network.

Example 1: Stop slave of No. 3 through NMT

◆ Request data:

PLC device	Request message		
		High byte(Hex)	**Low byte(Hex)**
D6250	Message header	ReqID =01	Command =01
D6251		Reserved =0	Size =04
D6252		Type =03	Node ID =03
D6253	Message data	Reserved	NMT service code =02
D6254		Reserved	Node ID =03

◆ Response data:

PLC device	Response message		
		High byte(Hex)	**Low byte(Hex)**
D6000	Message header	ResID=01	Status code =01
D6001		Reserved =0	Reserved =0
D6002		Reserved =0	Node ID =03

7.4.3 Data Structure of EMERGENCY Request Message

Through reading Emergency, the slave error and alarm information can be read.

➢ The data format of the Emergency request message:

PLC device	Request message		
		High byte	**Low byte**
D6250	Message header	ReqID	Command (Fixed to 1)
D6251		Reserved	Size (Fixed to 0)
D6252		Type (Fixed to 04)	Node ID
D6253~D6281	Message data	Reserved	

✧ Command: Fixed to 01.

✧ ReqID: The request ID. Whenever an Emergency message is sent out, the message will be given a ReqID for the CANopen master to identify. Before another Emergency request message is sent out, the original ID number must be changed. In other words, to send out the Emergency request message is triggered by changing of the value of “ReqID”. ReqID range: 00 (Hex) ~ FF (Hex).

✧ Node ID: The node address of the target equipment on CANopen network.

➢ The data format of the Emergency response message:

PLC device	Response message		
		High byte(Hex)	**Low byte(Hex)**
D6000	Message header	ResID	Status code
D6001		Reserved	Size Fixed to 2A
D6002		Type (Fixed to 04)	Node ID
D6003	Message data	Total number of data	Number of data stored
D6004		Datum 1	Datum 0
D6005		Datum 3	Datum 2
D6006		Datum 5	Datum 4
D6007		Datum 7	Datum 6
D6008 ~ D6011		Emergency2	
D6012 ~ D6015		Emergency3	
D6016 ~ D6019		Emergency4	
D6020~ D6023		Emergency5	
D6024~ D6031		Reserved	

✧ Command: Fixed to 01(Hex).

✧ When status code is 1, it indicates that reading Emergency message succeeds. When status code is not equal to1, it indicates that reading Emergency message fails and in the meantime, you should check if the data in Emergency message are correct.

- ✧ Node ID: The node address of the target equipment on CANopen network.
- ✧ Total number of data: The total number of Emergency messages CANopen master receives from the slave.
- ✧ Number of data stored: The latest number of Emergency messages CANopen master receives from the slave. (5 messages at most)
- ✧ The data in D6004-D6007 are the content of Emergency 1 and every Emergency message consists of 8 bytes of data.

  The data format of Emergency message on CAN bus is shown below. Datum 0~ datum 7 in Emergency response message correspond to byte 0~ byte 7 respectively

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
80（hex）+Node-ID	Emergency error code		Error storage register	Vendor custom error code				

Example 1: Read the Emergency message of slave of No.2 and the Emergency messages the slave sends out successively are shown below.

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
82（hex）	43	54	20	14	0	0	0	0

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
82（hex）	42	54	20	15	0	0	0	0

- ◆ Request data:

PLC device	Request message		
		High byte	Low byte
D6250	Message header	ReqID=01	Command =01
D6251		Reserved	Size =0
D6252		Type =04	Node ID =02

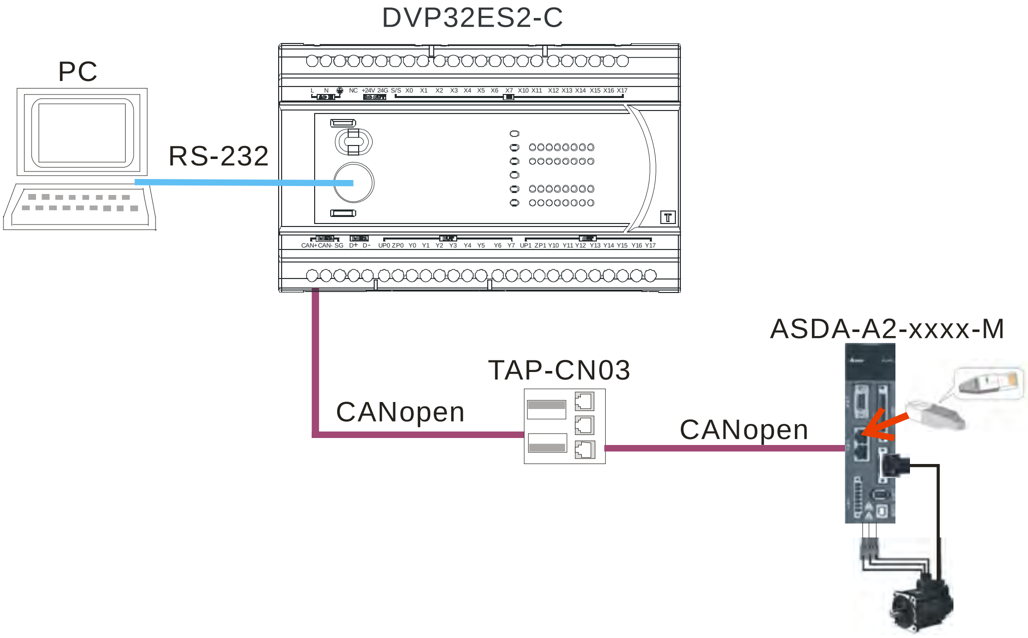
- ◆ Emergency response data

PLC device	Response message		
		High byte	Low byte
D6000	Message header	ResID=01	Status code =01
D6001		Reserved =0	Size =2A (Hex)
D6002		Type =04	Node ID =02
D6003	Message data	Total number of data =1	Number of data stored =1
D6004		Datum 1=54	Datum 0=42
D6005		Datum 3=15	Datum 2=20
D6006		Datum 5=0	Datum 4=0
D6007		Datum 7=0	Datum 6=0
D6004		Datum 1=54	Datum 0=43
D6005		Datum 3=14	Datum 2=20
D6006		Datum 5=0	Datum 4=0
D6007		Datum 7=0	Datum 6=0

7.4.4 Example on Sending SDO through the Ladder Diagram

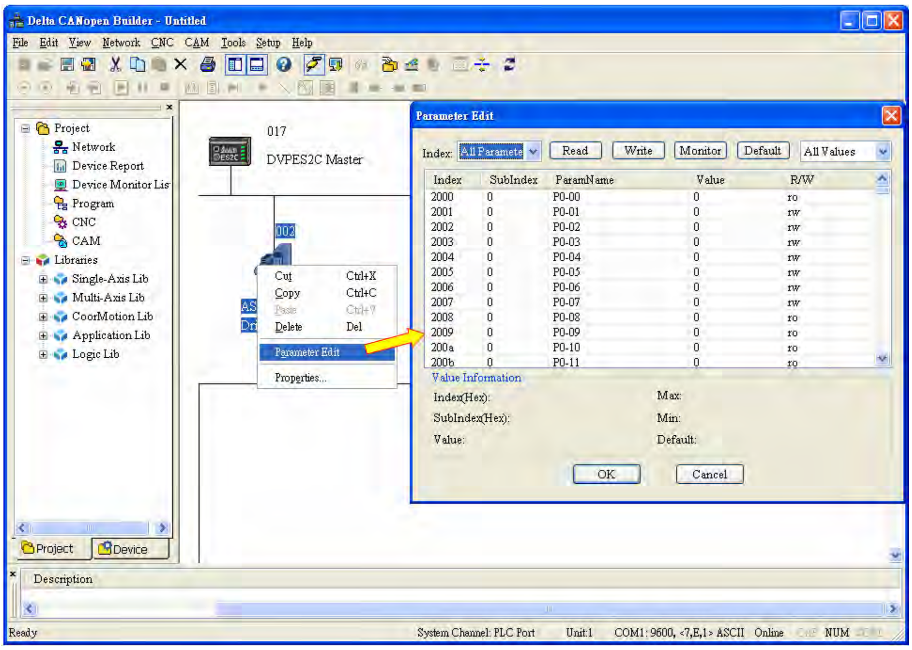
- ➢ **Control Requirement:**

  Read the value of P0-09 of servo in cycle through SDO.
- ➢ **Hardware Connection:**



- **The Corresponding Relation between Slave Parameter and Index/Subindex**

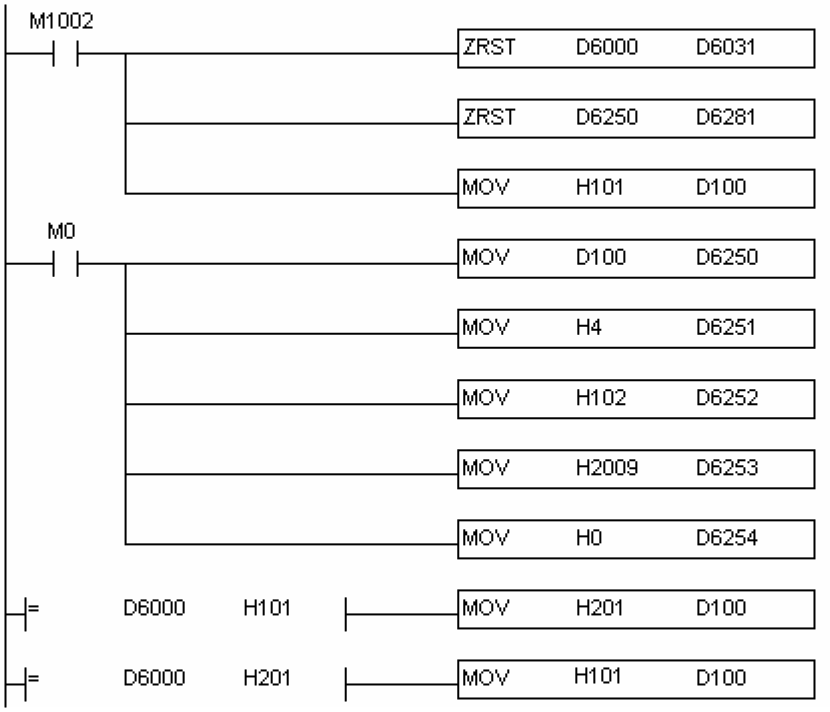
  The index_subindex corresponding to P0-09 of servo is 2009_0. On the interface of the network configuration software, right click the servo icon; select “Parameter Edit” and then the following dialog box will appear. You can see the index_subindex corresponding to the servo parameter in the dialog box.
  For more details on how to operate the network configuration interface, please refer to section 11.1.1 of the help file of CANopen Builder software.



- **Explanation of Request Message Devices:**

PLC device		Content (Hex)	Explanation	
			High byte(Hex)	Low byte(Hex)
SDO request message mapping area	D6250	0101	ReqID = 01	Command = 01
	D6251	0004	Reserved	Size = 04
	D6252	0102	Type = 01	Node ID = 02
	D6253	2009	Index high byte = 20	Index low byte = 09
	D6254	0000	Reserved	Subindex = 00

- **Editing the Ladder Diagram through WPLsoft**



When M0=ON, DVP-ES2-C sends out the first request message and D6000 should be 101(hex) after the response message is transmitted back successfully. In program, if the value of D6000 is judged as 101(hex), the ReqID is changed into 2 and D6250 is given a new value 201(hex) and DVP-ES2-C sends out the request message again. By dong so, the real-time reading can be realized. After reading succeeds, the data returning from the target device are stored in D6000~D6005. The value of D6005: 100(hex)is the read value of P0-09.

- **Explanation of Response Message Devices:**

PLC device		Content (Hex)	Explanation	
			High byte(Hex)	Low byte(Hex)
SDO response message mapping area	D6000	0101	ResID = 01	Status code = 01
	D6001	0008	Reserved	Size = 08
	D6002	4302	Type = 43	Node ID = 02
	D6003	2009	Main index high byte = 20	Index low byte = 09
	D6004	0004	Reserved	Subindex = 00
	D6005	0100	Datum 1= 01	Datum 0= 00
	D6006	0100	Datum 3= 00	Datum 2= 00

7.5 Indicators and Troubleshooting

There are 6 LED indicators on DVP-ES2-C. Power indicator shows whether the power is normal, RUN and ERROR indicator display the state of running of internal program and COM3 displays the communication state of CANopen.

7.5.1 Description of Indicators

- **POWER indicator**

LED indicator	Description	How to deal with
Light is off or the green light flashes	The supply power is abnormal	Check if the supply power is in the valid range
The green light keeps on	The supply power is normal	No resolution is required

➢ **RUN indicator**

LED indicator	Description	How to deal with
The green light is on.	PLC is running	No resolution is required
Light is off	PLC is in stop status	Make PLC run through RUN/STOP switch or WPLSoft

➢ **ERROR indicator**

LED indicator	Description	How to deal with
Light is off	PLC is normal	No resolution is required
The red light flashes.	There are syntax error existing in the program written to PLC or the PLC device or instruction is out of the allowed range.	Judge the error cause according to the content value of the special register D1004 in PLC; find the program error position according to the content value of D1137. For more details on the error codes in D1004, please refer to “ES2 series PLC operation manual (Programming)”.
The red light keeps on.	PLC scan is timed-out	Reduce the time for executing PLC program or use WDT instruction

➢ **COM3（CANopen）Indicator**

LED indicator	Description	How to deal with
The green light keeps on.	DVP-ES2-C is normal.	No resolution is required
The green light is in single flash.	DVP-ES2-C stops.	The superior equipment is downloading the network configuration and waiting to complete downloading.
The green light flashes.	As DVP-ES2-C is in slave mode, it is preoperational; As DVP-ES2-C is in master mode, some slave is offline.	1. Check whether the wiring of the CANopen bus cable is correct. 2. Check whether the baud rate of the master is the same as that of the slave. 3. Check if the configured slaves have connected to the network. 4. Check if any slave is offline.
The red light is in double flashes.	The slave is off-line.	1. Check whether the CANopen bus cable is a standard one. 2. Check whether both ends of the CANopen bus are connected to the terminal resistors.
The red light in single flash.	At least one error counter in the CAN controller reaches or exceeds the warning level.	1. Check whether the CANopen bus cable is a standard one. 2. Check whether both ends of the CANopen bus are connected to the terminal resistors. 3. Check whether there is much interference around the CANopen bus cable.
The red light keeps on.	Bus-off	1. Check whether the wiring of the bus cable in the CANopen network is correct. 2. Check whether the baud rate of the master is the same as that of the slave.

7.5.2 CANopen Network Node State Display

➢ While the special auxiliary relay M1349 of DVP-ES2-C is ON, the CANopen function is enabled and D9980~D9998 will be used as the special registers as the table shows below.

Special register	Function
D9980	Used for displaying the state of DVP-ES2-C.
D9981~D9996	Used for displaying the state of 16 nodes in the network
D9998	Used for monitoring the state of the entire CANopen network

D9999	Used for displaying a CAB baud rate K1: 20K; K2: 50K; K3: 125K; K4: 250K; K5: 500K; K6: 1M (Only applicable to DVP-ES2-C V3.26 (and above))

- As a master, DVP-ES2-C supports maximum 16 slaves ranging from node 1 to node 16. D9998 can be used for monitoring the state of nodes from 1 to 16 in the network. And the 16 bits of D9998 corresponds to 16 slaves and the corresponding relations of them are shown below.

Bit	**b7**	**b6**	**b5**	**b4**	**b3**	**b2**	**b1**	**b0**
Node	Node 8	Node 7	Node 6	Node 5	Node 4	Node 3	Node 2	Node 1
Bit	**b15**	**b14**	**b13**	**b12**	**b11**	**b10**	**b9**	**b8**
Node	Node16	Node15	Node14	Node13	Node12	Node11	Node10	Node 9

When the node in the master node list is normal, the corresponding bit is OFF; when the node in the master node list is abnormal (E.g. Initializing fails or slave is offline due to other abnormality), the corresponding bit is ON.

- The error code of every node is displayed through the corresponding special register and the relations between special register and corresponding node are shown below.

Special register	**D9981**	**D9982**	**D9983**	**D9984**	**D9985**	**D9986**	**D9987**	**D9988**
Node	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8
Special register	**D9989**	**D9990**	**D9991**	**D9992**	**D9993**	**D9994**	**D9995**	**D9996**
Node	Node 9	Node10	Node11	Node12	Node13	Node14	Node15	Node16

- Code display in D9981~D9996 as DVP32ES2-C is in master mode:

Code	Indication	How to correct
E0	DVP-ES2-C master module receives the emergency message sent from slave.	Read the relevant message via PLC program
E1	PDO data length returned from the slave is not consistent with the length set in the node list.	Set the PDO data length of slave and re-download them.
E2	PDO of slave is not received.	Check and ensure the setting is correct.
E3	Downloading auto SDO fails.	Check and ensure auto SDO is correct.
E4	Configuration of PDO parameter fails.	Ensure that the PDO parameter setting is legal.
E5	Error in key parameter setting.	Ensure that the actually connected slave is consistent with the configured slave.
E6	The slave does not exist in the network	Ensure that the supply power of slave is normal and the connection in the network is proper.
E7	Slave error control is timed-out.	
E8	The node IDs of master and slave repeat.	Set the node ID of master and slave again and ensure their node IDs are sole.

- Code display in D9980 as DVP-ES2-C is in master mode:

Code	Indication	How to correct
F1	Slave has not been added to node list of CANopen Builder software	Add slave into the node list and then re-download the configured data.
F2	The data are being downloaded to DVP-ES2-C	Wait to finish downloading the configured data.
F3	DVP-ES2-C is in error status	Re-download parameter configuration
F4	Bus-off is detected.	Check if CANopen bus cables are properly connected and ensure that all the node devices run at the same baud rate before re-powering.

Code	Indication	How to correct
F5	DVP-ES2-C setting error such as incorrect node address	The node address of DVP-ES2-C should be set in the range: 1~127.
F8	Internal error; the error is detected in the internal memory	After re-powering, change into a new one if the error still exists.
FB	The sending buffer in DVP-ES2-C is full.	Check if the CANopen bus cable is properly connected and then re-power.
FC	The receiving buffer in DVP-ES2-C is full.	Check if the CANopen bus cable is properly connected and then re-power.

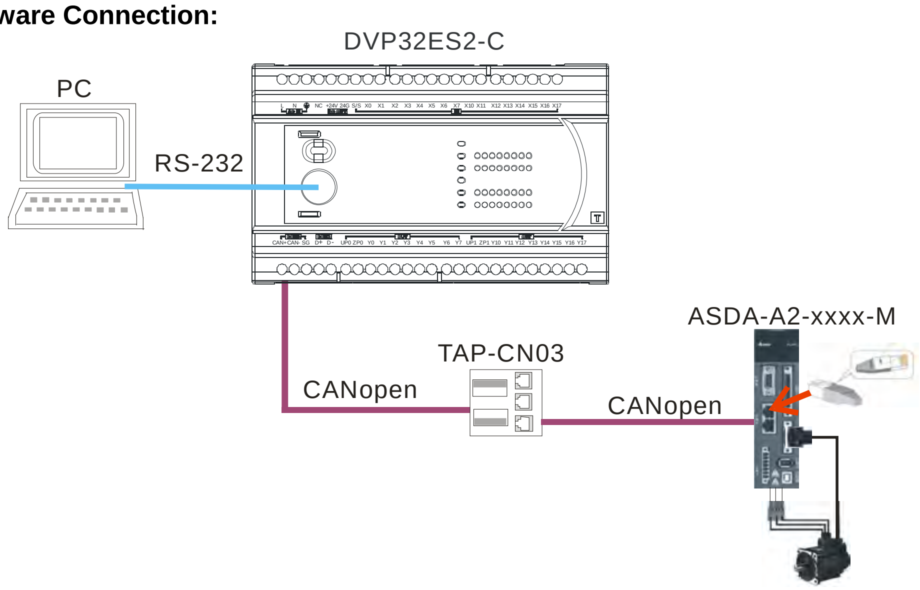
➢ Code display in D9980 as DVP32ES2-C is in slave mode:

Code	Indication	How to correct
A0	DVP-ES2-C is being initialized.	--
A1	DVP-ES2-C is pre-operational.	Check if the CANopen bus cable is properly connected
A3	The data are being downloaded to DVP-ES2-C	Wait to finish downloading the configured data.
B0	Heartbeat message is timed-out	Check if the CANopen bus cable is properly connected.
B1	PDO data length returned from the slave is not consistent with the length set in the node list.	Reset the PDO data length in the slave and download the new setting to DVPCOPM-SL.
F4	BUS-OFF state is detected.	Check if CANopen bus cables are properly connected and ensure that all the node devices run at the same baud rate before re-powering.
FB	The sending buffer in DVP-ES2-C is full.	Check if the CANopen bus cable is properly connected and then re-power.
FC	The receiving buffer in DVP-ES2-C is full.	Check if the CANopen bus cable is properly connected and then re-power.

7.6 Application Example

DVP-ES2-C is used to control Delta A2 servo rotation and monitor the actual rotation speed of motor in real time. The principle of operation is to map the relevant parameters of servo drive to the corresponding PDO and read or write the relevant parameters of servo drive through the CAN bus to accomplish the control requirement.

➢ **Hareware Connection:**



Note:

1. We recommend user use the standard communication cable UC-DN01Z-01A (TAP-CB01) / UC-DN01Z-02A (TAP-CB02) / UC-CMC010-01A (TAP-CB10) and connect the terminal resistors such as Delta standard terminal resistor TAP-TR01 to either terminal of the network when constructing the network.
2. TAP-CN03 is a distribution box and the resistance it has can be effective after its SW1 is switched to ON. According to actual demand, user could select TAP-CN01/CN02/CN03 for wiring.
3. M of ASD-A2-xxxx-M refers to the model code and currently only the M-model servo supports CANopen communication.

➢ **Setting Servo Parameters:**

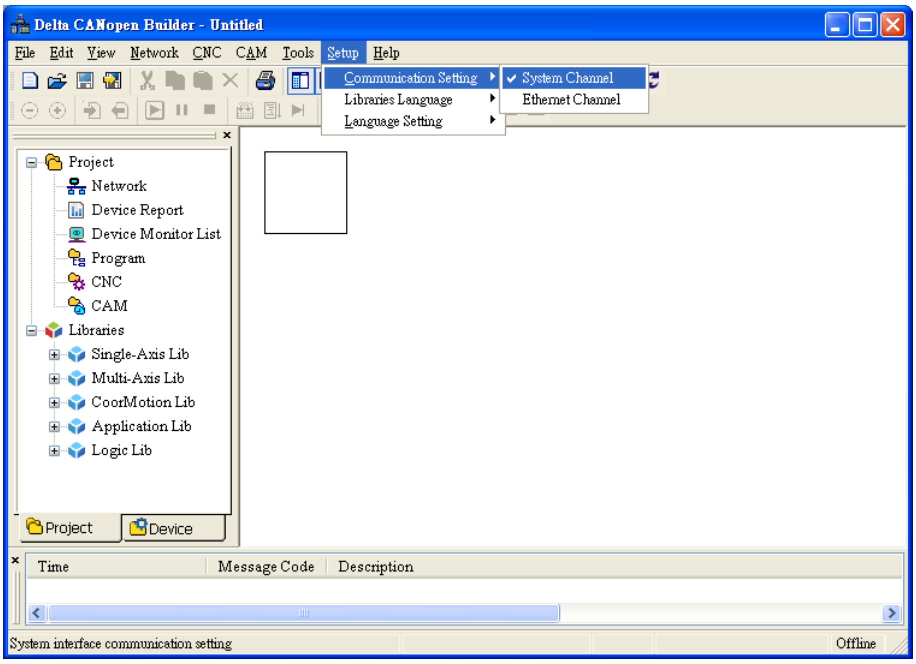
✧ Set servo parameters as follows:

Parameter	Setting	Explanation
3-00	02	The Node ID of A2 servo is 2
3-01	400	CAN communication rate is 1Mbps.
1-01	04	Speed mode
0-17	07	Drive displays the motor rotation speed (r/min)
2-10	101	Set DI1 as the signal for Servo On
2-12	114	Set DI3 and DI4 as the signal for speed selection

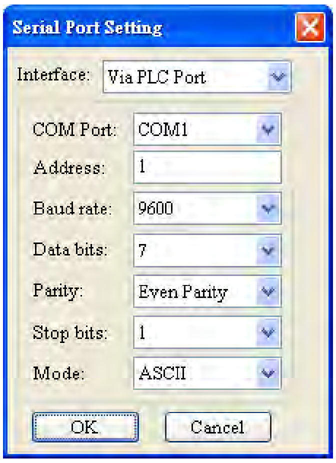
➢ **Setting CANopen Baud Rate and Node ID of DVP-ES2-C**

DVP-ES2-C uses the default setting values: Node ID: 17 and baud rate: 1Mbps. CANopen Node ID and baud rate of DVP-ES2-C are set up through CANopen Builder software. See the detailed operation steps below:

1) Open CANopen Builder software and then click menu “Setup” > “Communication setting” > “System Channel”.

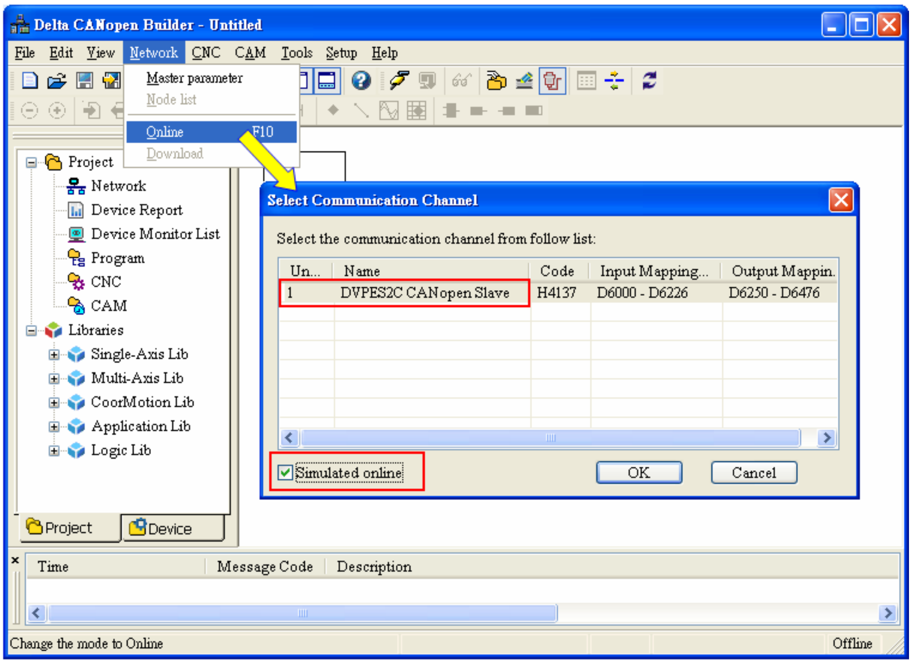


2) The following window will appear where to set up the serial port communication parameters.



Item	Explanation	Default
Interface	If the equipment connected to computer is DVP10MC11T, select Via Local Port; otherwise, select Via PLC Port.	--
COM port	The serial port of computer used for communication with DVP-ES2-C.	COM1
Address	The communication address of DVP-ES2-C	01
Baud rate	The communication rate between computer and DVP-ES2-C	9600 bps
Data bits	The communication protocol between computer and DVP-ES2-C	7
Parity		Even parity
Stop bit		1
Mode	The communication mode between computer and DVP-ES2-C	ASCII Mode

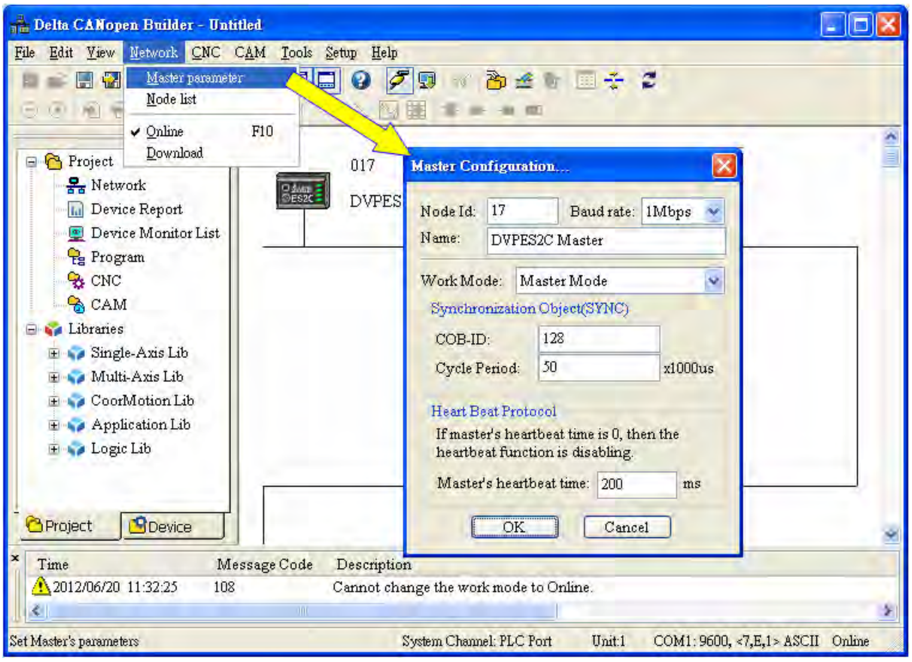
3) After setting is finished, click “Network”> “Online” and the “Select communication channel” page appears.



1> When “CANopen Slave” displays in the Name column, it indicates that PLC is in the mode of CANopen slave. At that time, select “Simulated online” on the bottom left side on the page and finally click “OK” to start the online scanning.

2> When “CANopen Master” displays in the Name column, it indicates that PLC is in the mode of CANopen master. At that time, directly click “OK” to start the online scanning.

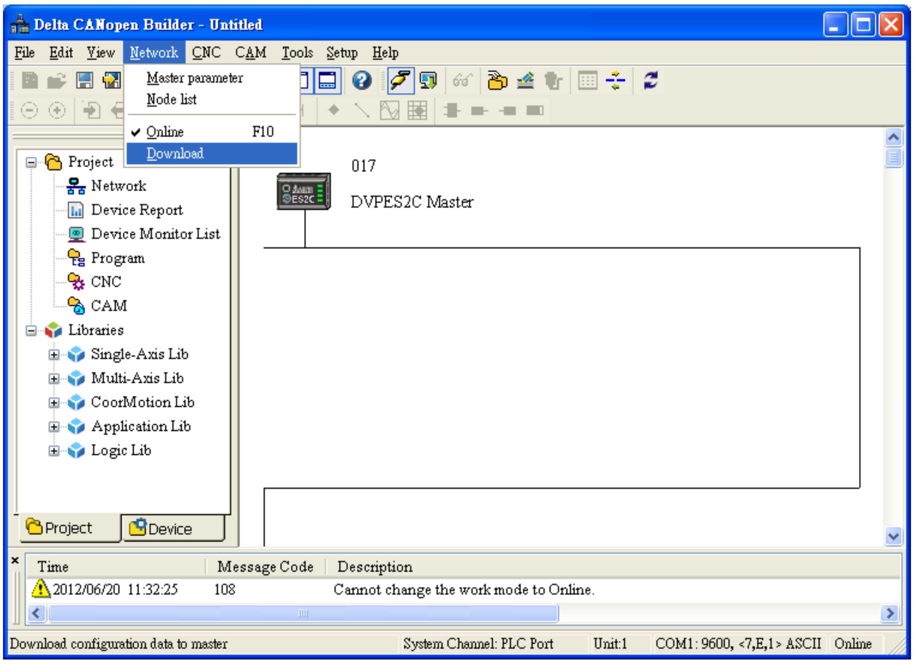
4) Click “Network”> “Master Parameter” and the following “Master configure…” dialog box appears.



Item	Explanation	Default
Node ID	The node ID of DVP-ES2-C on the CANopen network	17
Baud rate	CANopen communication rate	1M bit/second
Work mode	CANopen master/slave mode	Master
Cycle period	The cycle time for sending one SYNC message	50ms
Master’s heartbeat time	The interval time for sending the master heartbeat message	200ms

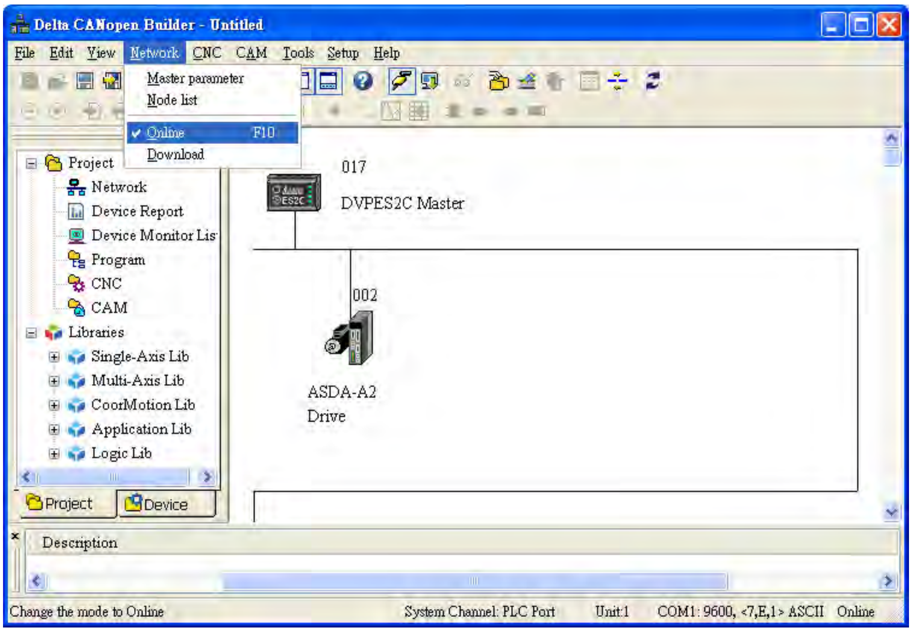
According to actual requirement, user can set the CANpen Node ID, baud rate and master/slave mode of DVP-ES2-C.

5) After the steps above are finished, the download will be performed as the figure shows below.



Note:
The new parameters after being downloaded will be effective unless DVP-ES2-C is re-powered.

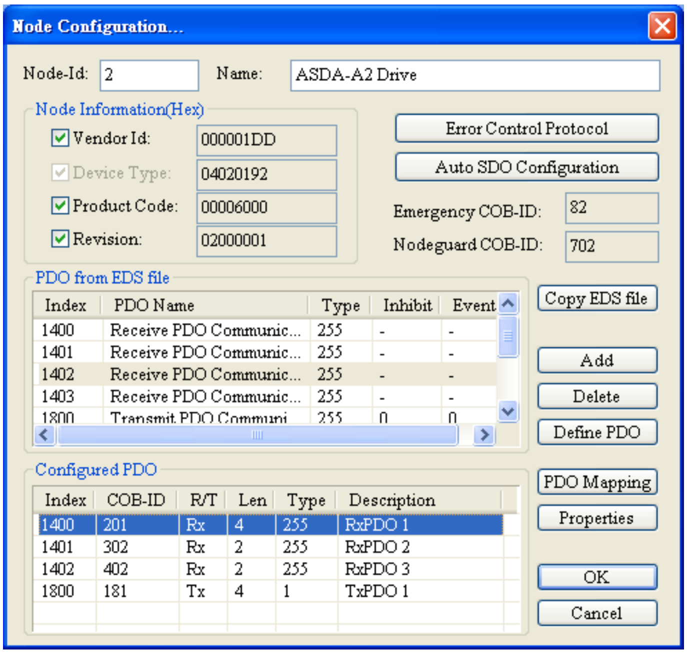
➢ **Network Scanning:**
Scan the master and slave on the CANopen network by clicking menu “Network”>>”Online”. The scanned master and slave are displayed on the page below. For detailed operation steps, please refer to Section 11.1.1 in the help file of CANopen Builder software.

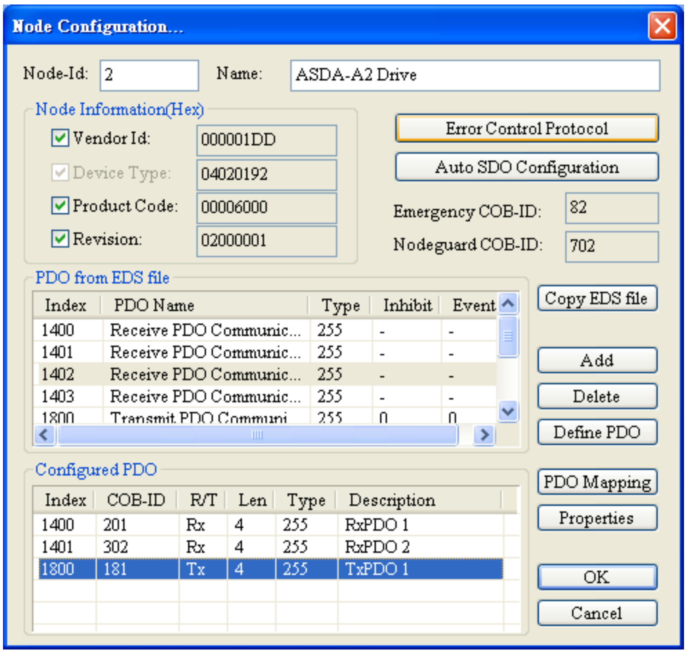


➢ **Node Configuration:**
Double click the slave icon on the above page and then the following “Node configuration” dialog box pops up.

✧ “Error Control Protocol”
Used for setting the error control protocol for master to monitor if the slave is offline.

- ✧ "Auto SDO Configuration"
  Used for doing one write action to the slave parameter via SDO and the write action is finished when the slave enters the operational state from pre-operational state. Up to 20 SDOs can be configured by "Auto SDO configuration".
- ✧ "PDO Mapping" and "Properties"
  Used for setting the mapping parameter and transmission type of PDO.
  For the details on the function buttons mentioned above, please refer to Section 11.1.1 in the help file of CANopen Builder software.





- ✧ PDO Mapping:
  RxPDO1: the mapping parameter P1-09; transmission type 255.
  RxPDO2: the mapping parameter P3-06, P4-07; transmission type 255.
  TxPDO1: the mapping parameter P0-09; transmission type 1.
- ✧ PDO transmission type :
  PDO can be classified into RxPDO and TxPDO. RxPDO data are sent from master to slave and TxPDO data are sent from slave to master.

PDO transmission type can be synchronous transmission and asynchronous transmission. In synchronous transmission, master will send out the SYNC message in the fixed cycle. The length of the cycle is set in master properties dialog box with the default value: 50ms. In asynchronous transmission, the message is sent out once the PDO mapping parameter is changed.

PDO Transmission types in details are introduced in the following table.

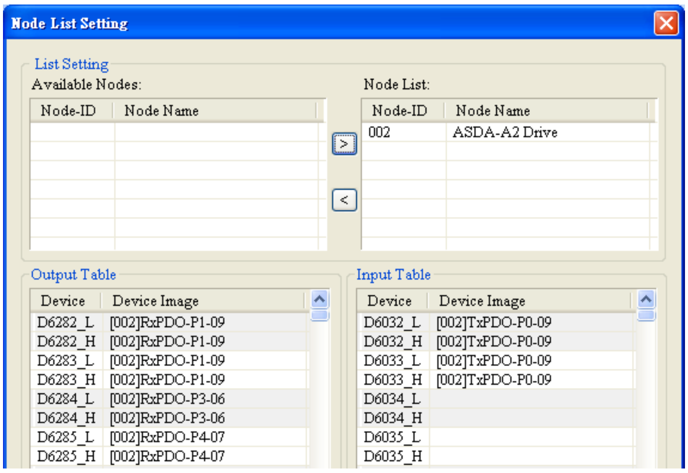
<table>
<tr><th colspan="2">Transmission Type</th><th>Description</th><th>Remark</th></tr>
<tr><td rowspan="2">0</td><td>RxPDO</td><td>Once any change for the mapped data happens, RxPDO data are sent out immediately. The data that slave receives are valid only when receiving the next SYNCH message. If no change for RxPDO data, they are not sent out.</td><td rowspan="2">SYNCH
SYNCH
non-cycle</td></tr>
<tr><td>TxPDO</td><td>Once any change for the mapped data happens and slave receives the SYNC message, the data are sent out immediately. The TxPDO data are valid immediately after master receives them. If no change for TxPDO data, the data are not sent out.</td></tr>
<tr><td rowspan="2">N
（N:1~240）</td><td>RxPDO</td><td>After N messages are sent out and no matter whether the mapped data are changed, the data that slave receives will be valid only when receiving the next SYNCH message.</td><td rowspan="2">SYNCH
cycle</td></tr>
<tr><td>TxPDO</td><td>After N messages are sent out and no matter whether the mapped data are changed, the data that master receives will be valid at once.</td></tr>
<tr><td rowspan="2">254</td><td>RxPDO</td><td>The mapped data are sent out immediately once changed and they are valid once they are received by slave. RxPDO data will not be sent out if no change for the data.</td><td rowspan="2">ASYNCH</td></tr>
<tr><td>TxPDO</td><td>Slave sends out the data once every one Event timer time and after that, the TxPDO data are not allowed to be sent out within an inhibit timer time.
When Event timer and Inhibit timer are both equal to 0, TxPDO data are sent to master immediately once changed and the data that master receives will be valid at once.</td></tr>
<tr><td>255</td><td colspan="3">Same as Type254</td></tr>
</table>

Note**:**

1> Synchronous transmission type can fulfill multi-axis motion at the same time.

2> If user is going to monitor the real-time changing parameter such as the actual rotation speed of the motor, we suggest TxPDO should be set as the synchronous transmission type in case the frequent changing of slave data causes to block the CANopen network.

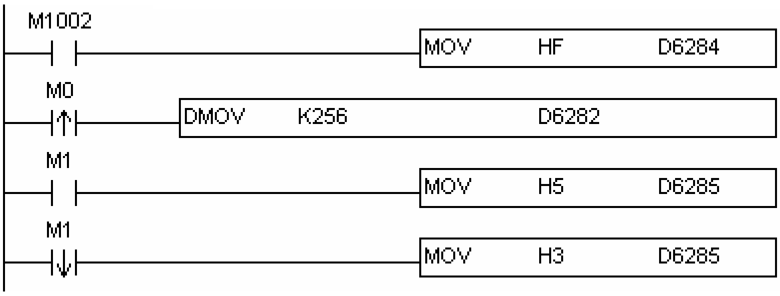
3> After the above setting is finished, double click the master, select ASDA-A2 Drive, and click “>” to move A2 to the right list and download the configured data.



The mapping relation between master and slave:

DVP-ES2-C master register	Data transmission on CANopen bus	A2 device
D6282	→	Low word of P1-09 of servo
D6283		High word of P1-09 of servo
D6284		P3-06 of servo
D6285		P4-07 of servo
D6032	←	Low word of P0-09 of servo
D6033		High word of P0-09 of servo

➢ Program control: D6282 is given the value K256 through WPL software. That is, the speed command is set as 256r/min. See details in the following figure.



Program explanation:

While DVP-ES2-C is running for the first time, set the parameter P3-06 of servo drive to F.

- When M0 turns from OFF to ON, write K256 to D6282 and then the value is written to P1-09 of servo parameter through RxPDO1.
- When M1 turns from OFF to ON, turn P2-12 on and call the speed specified by parameter P1-09 of servo for rotation.
- When M1 turns from ON to OFF, the speed command becomes 0 and the motor stops running.

7.7 Object Dictionary

The communication objects in the object dictionary are shown as below:

Index	Subindex	Object name	Data type	Attribute	Default value
H'1000	H'00	Device type	Unsigned 32 bits	R	0x00000000
H'1001	H'00	Error register	Unsigned 8 bits	R	0
H'1005	H'00	COB-ID SYNC	Unsigned 32 bits	RW	0x00000080
H'1008	H'00	manufacturer device name	Vis-String	R	DVPES2C
H'1014	H'00	COB-ID EMCY	Unsigned 32 bits	R	0x80 + Node-ID
H'1016	--	Consumer heartbeat time			
	H'00	Number of valid subindex	Unsigned 8 bits	R	1
	H'01	Consumer heartbeat time	Unsigned 32 bits	RW	0
H'1017	H'00	Producer heartbeat time	Unsigned 16 bits	RW	0
H'1018	--	Identity Object			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	Vendor-ID	Unsigned 32 bits	R	0x000001DD
	H'02	Product code	Unsigned 32 bits	R	0x00000055
	H'03	Revision number	Unsigned 32 bits	R	0x00010002
H'1400	--	RxPDO1 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO1	Unsigned 32 bits	RW	0x00000200+ Node-ID
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1401	--	RxPDO2 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO2	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1402	--	RxPDO3 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO3	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF

Index	Subindex	Object name	Data type	Attribute	Default value
H'1402	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1403	--	RxPDO4 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO4	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1404	--	RxPDO5 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO5	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1405	--	RxPDO6 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO6	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1406	--	RxPDO7 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO7	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1407	--	RxPDO8 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO8	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1600	--	RxPDO1 mapping parameter			

Index	Subindex	Object name	Data type	Attribute	Default value
H'1600	H'00	Number of valid subindex	Unsigned 8 bits	RW	4
	H'01	The first mapped object	Unsigned 32 bits	RW	0x20000110
	H'01	The second mapped object	Unsigned 32 bits	RW	0x20000210
	H'02	The third mapped object	Unsigned 32 bits	RW	0x20000310
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0x20000410
H'1601	--	RxPDO2 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0

Index	Subindex	Object name	Data type	Attribute	Default value
H'1602	--	RxPDO3 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0
H'1603	--	RxPDO4 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0
H'1604	--	RxPDO5 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0

Index	Subindex	Object name	Data type	Attribute	Default value
H’1604	H’01	The first mapped object	Unsigned 32 bits	RW	0
	H’01	The second mapped object	Unsigned 32 bits	RW	0
	H’02	The third mapped object	Unsigned 32 bits	RW	0
	H’03	The fourth mapped object	Unsigned 32 bits	RW	0
H’1605	--	RxPDO6 mapping parameter			
	H’00	Number of valid subindex	Unsigned 8 bits	RW	0
	H’01	The first mapped object	Unsigned 32 bits	RW	0
	H’01	The second mapped object	Unsigned 32 bits	RW	0
	H’02	The third mapped object	Unsigned 32 bits	RW	0
	H’03	The fourth mapped object	Unsigned 32 bits	RW	0

Index	Subindex	Object name	Data type	Attribute	Default value
H’1606	--	RxPDO7 mapping parameter			
	H’00	Number of valid subindex	Unsigned 8 bits	RW	0
	H’01	The first mapped object	Unsigned 32 bits	RW	0
	H’01	The second mapped object	Unsigned 32 bits	RW	0
	H’02	The third mapped object	Unsigned 32 bits	RW	0
	H’03	The fourth mapped object	Unsigned 32 bits	RW	0
H’1607	--	RxPDO8 mapping parameter			
	H’00	Number of valid subindex	Unsigned 8 bits	RW	0
	H’01	The first mapped object	Unsigned 32 bits	RW	0
	H’01	The second mapped object	Unsigned 32 bits	RW	0
	H’02	The third mapped object	Unsigned 32 bits	RW	0
	H’03	The fourth mapped object	Unsigned 32 bits	RW	0
H’1800	--	TxPDO1 communication parameter			
	H’00	Number of valid subindex	Unsigned 8 bits	R	5
	H’01	COB-ID of TxPDO1	Unsigned 32 bits	RW	0x00000180+ Node-ID
	H’02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H’03	Inhibit time	Unsigned 16 bits	RW	50

Index	Subindex	Object name	Data type	Attribute	Default value
H'1800	H'05	Timer	Unsigned 16 bits	RW	100
H'1801	--	TxPDO2 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO2	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100

Index	Subindex	Object name	Data type	Attribute	Default value
H'1802	--	TxPDO3 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO3	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1803	--	TxPDO4 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO4	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1804	--	TxPDO5 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO5	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1805	--	TxPDO6 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO6	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF

Index	Subindex	Object name	Data type	Attribute	Default value
H'1805	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1806	--	TxPDO7 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO7	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1807	--	TxPDO8 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO8	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1A00	--	TxPDO1 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	4
	H'01	The first mapped object	Unsigned 32 bits	RW	0x20010110
	H'02	The second mapped object	Unsigned 32 bits	RW	0x20010210
	H'03	The third mapped object	Unsigned 32 bits	RW	0x20010310
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0x20010410
H'1A01	--	TxPDO2 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0
H'1A02	--	TxPDO3 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0

Index	Subindex	Object name	Data type	Attribute	Default value
H'1A02	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0
H'1A03	--	TxPDO4 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
H'1A04	--	TxPDO5 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0

Index	Subindex	Object name	Data type	Attribute	Default value
H'1A05	--	TxPDO6 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0
H'1A06	--	TxPDO7 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0
H'1A07	--	TxPDO8 mapping parameter			

Index	Subindex	Object name	Data type	Attribute	Default value
H’1A07	H’00	Number of valid subindex	Unsigned 8 bits	RW	0
	H’01	The first mapped object	Unsigned 32 bits	RW	0
	H’02	The second mapped object	Unsigned 32 bits	RW	0
	H’03	The third mapped object	Unsigned 32 bits	RW	0
	H’04	The fourth mapped object	Unsigned 32 bits	RW	0

Appendix

Installing a USB Driver in the PLC

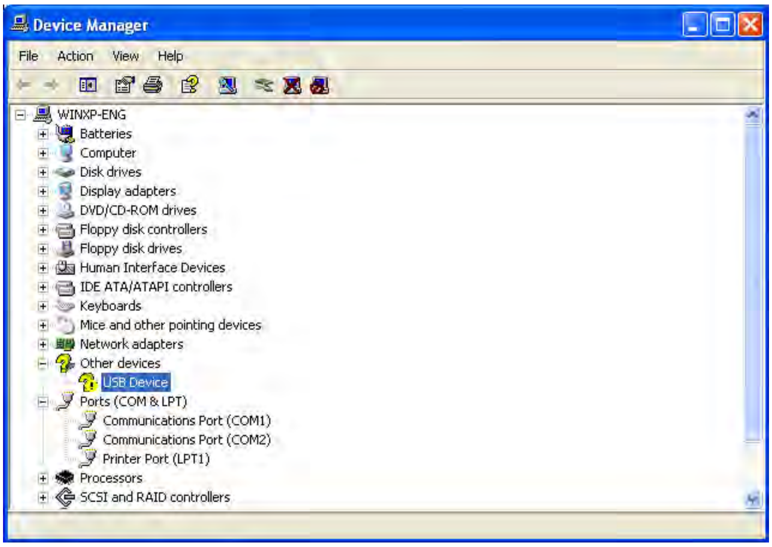
Contents

A.1 Installing the USB Driver in Windows 7

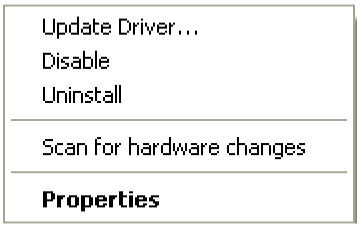
This section introduces the installation of the DELTA PLC USB driver in the computer. After the driver is installed, the USB interface can be used as the serial port (RS-232). Please use the standard USB cable. The length of the cable should be within fiver meters.

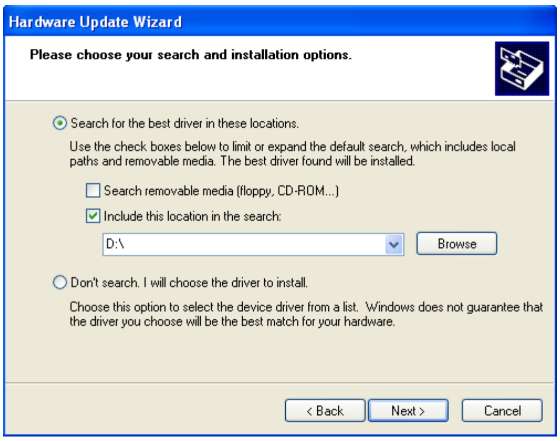
Installing the driver

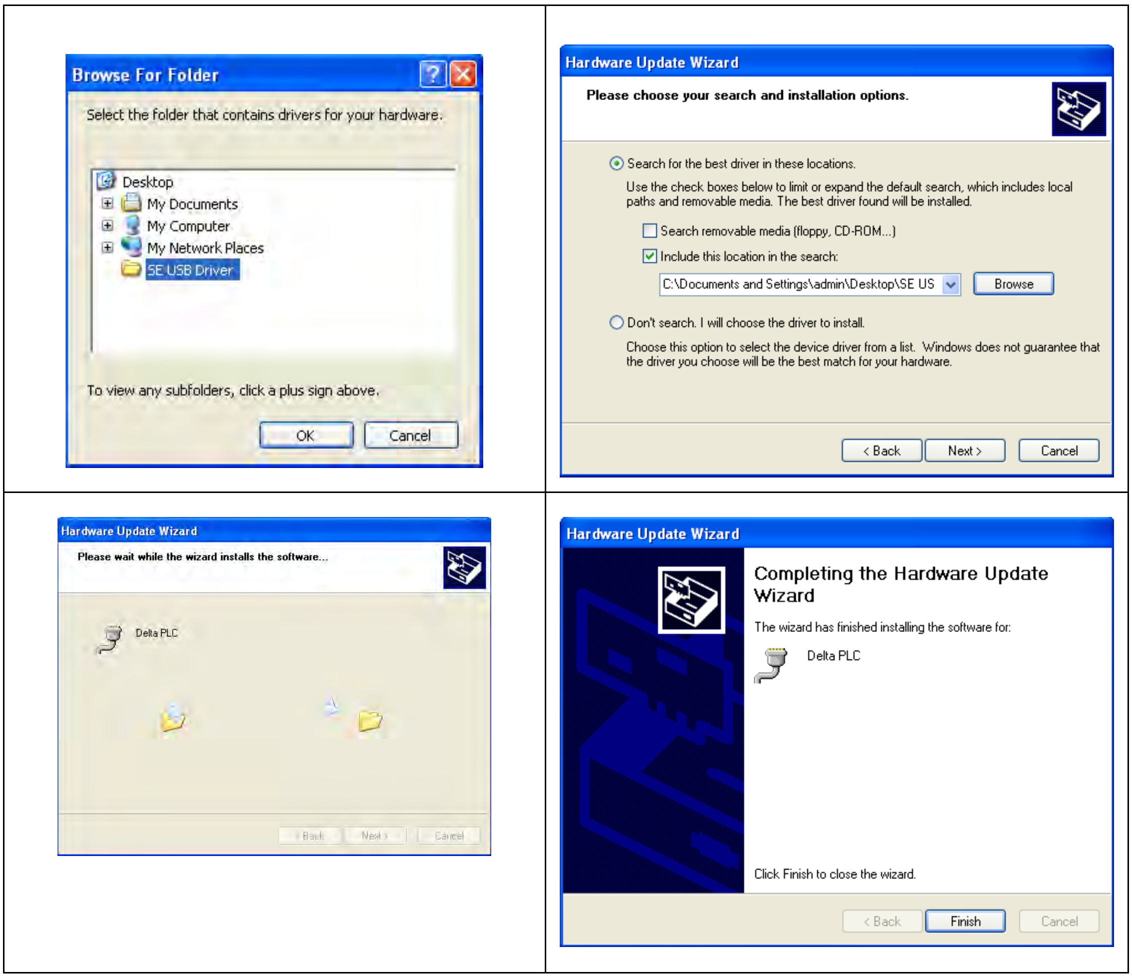
The personal computer and the PLC are connected through the USB and the mini USB cable. After they are connected, users can find **USB Device** in the **Device Manager** window.



Click the right mouse button, and select **Update Driver…** to open the **Hardware Update Wizard** window. Click **Browse** to specify the folder, and then click **Next** to start the installation of the driver.

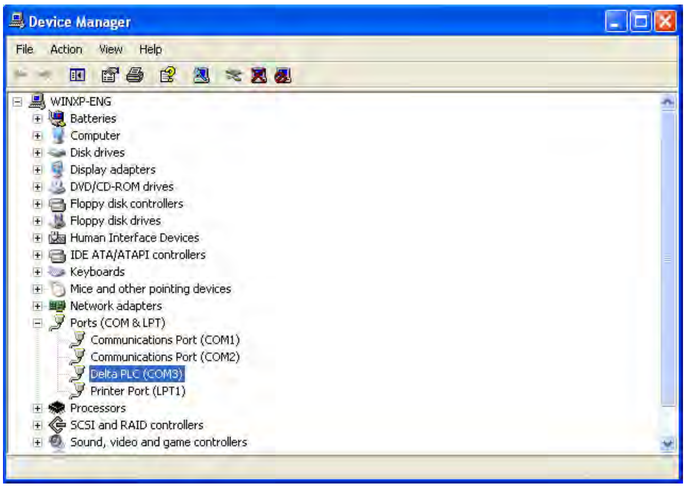




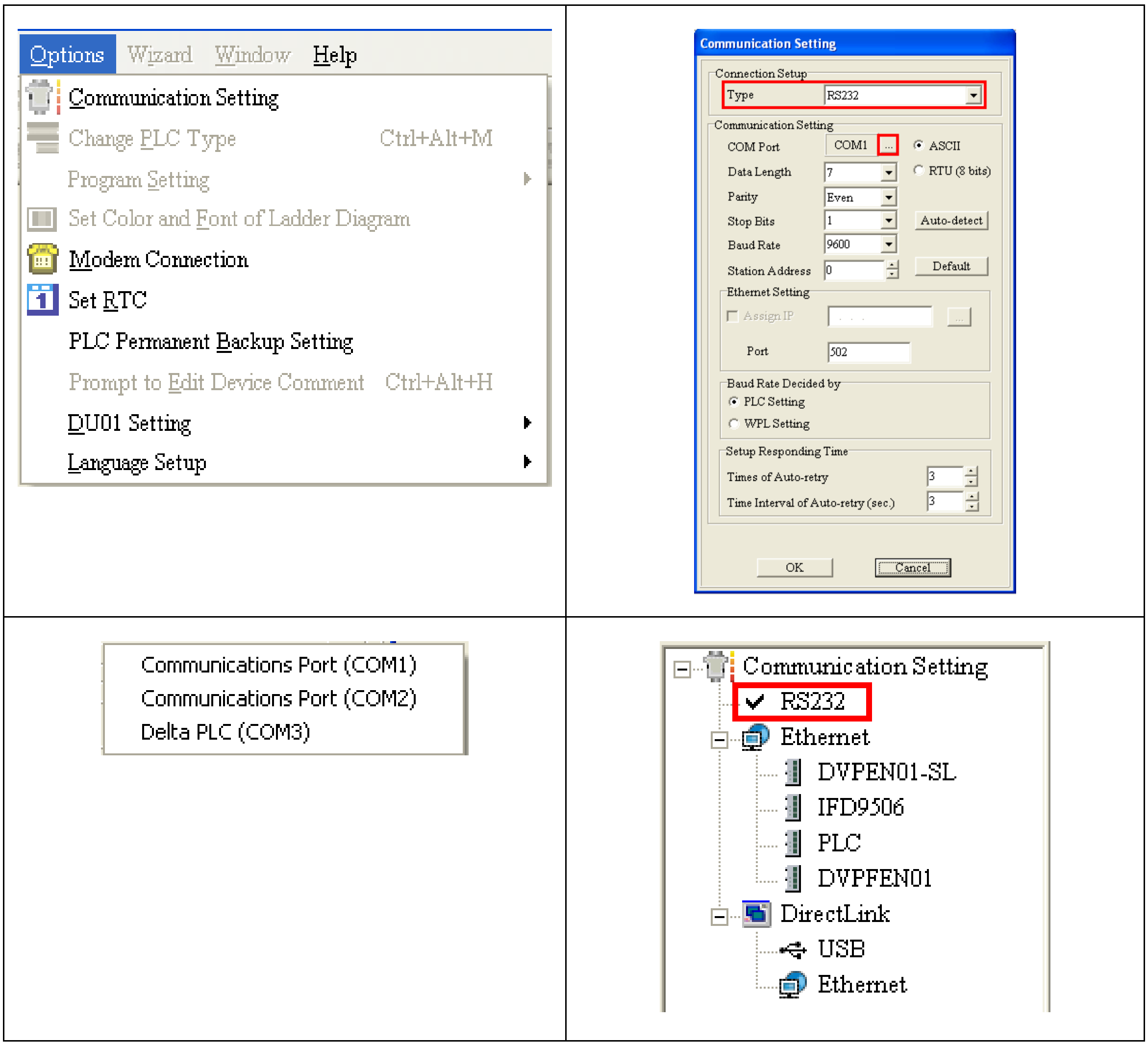


After the driver is installed, users can find the **Delta PLC** device and the communication port assigned to it in the **Device Manger** window. The usage of this device is the same as that of RS-232.

Note: If more than two USB COM ports are being used at the same time, there should be 2 different COM port numbers. If 2 COM port shares the same number, you need to edit the COM port number manually.



Select **Communication Setting** in **Options** to open the **Communication Setting** window. Select **RS232** in the **Connection Setup** box, select the communication port assigned by the USB in the **Communication Setting** box, and click **OK**. After the communication setting is complete, users can find that **RS232** in the communication work area is checked. They can download the program to the PLC and upload the program from the PLC through the USB, and can use the online mode.



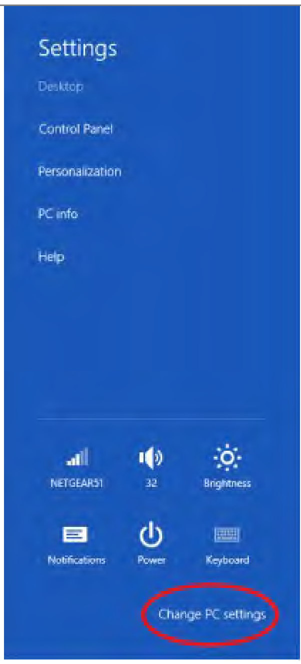
A.2 Installing the USB in Windows 8

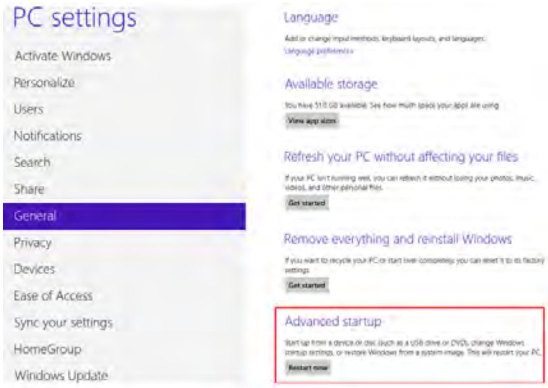
Windows 8 driver signature enforcement provides a way to improve the security of the operating system by validating the integrity of a driver or system file each time it is loaded into memory. However since Delta PLC USB driver does not include the driver signature, this section will help users to disable driver signature enforcement functionality in Windows 8 to ensure a success Delta PLC USB installation. This act is only valid for a single time. The setting will return to its original state after restarting.

Steps to disable driver signature enforcement in Windows 8:

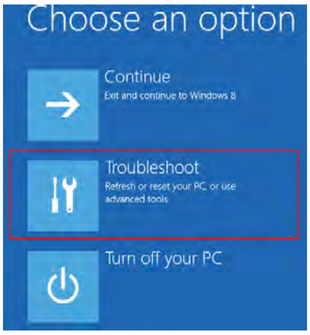
1. Press the button  【WIN】+【I】 on the keyboard to see the Setting interface. Click “Change PC settings”.

2. The PC settings window will appear. Select “General” and then “Restart now” under “Advanced startup”.

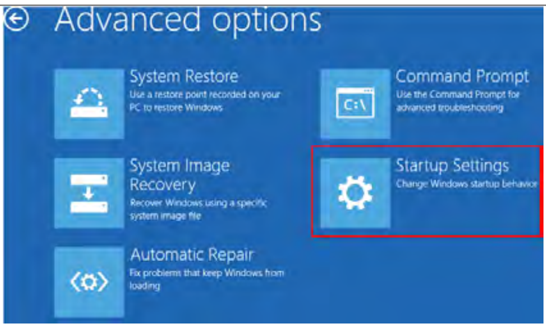


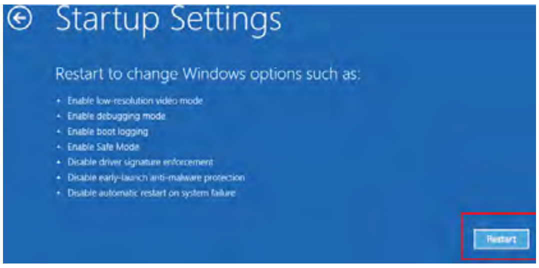


3. After the computer is restarted, select “Troubleshoot” under “Choose an option”. And then select “Advanced options”.

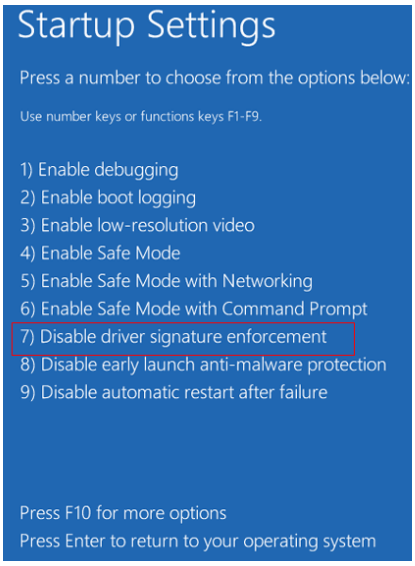


4. From the Advanced options page, select “Startup Settings” to see the Startup Settings. From this page select “Restart” to restart the computer.





5. Press “7” or “F7” to choose “Disable driver signature enforcement” and the system will direct you to the Windows 8 operating page. Users can then install the Delta PLC USB driver now.

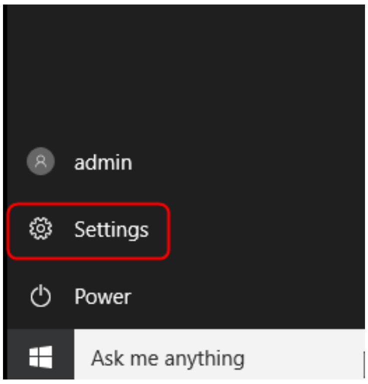
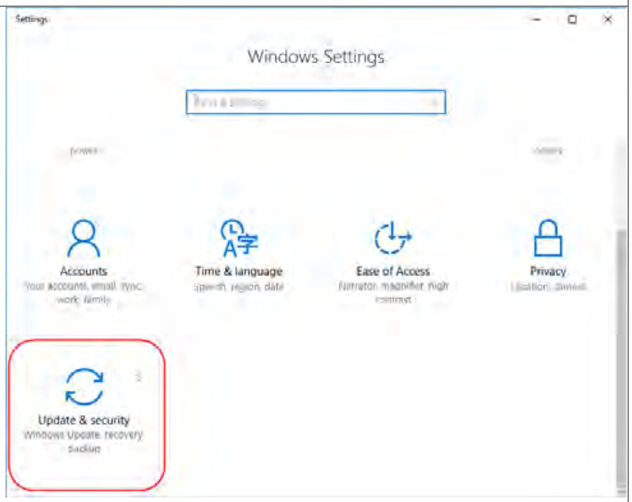


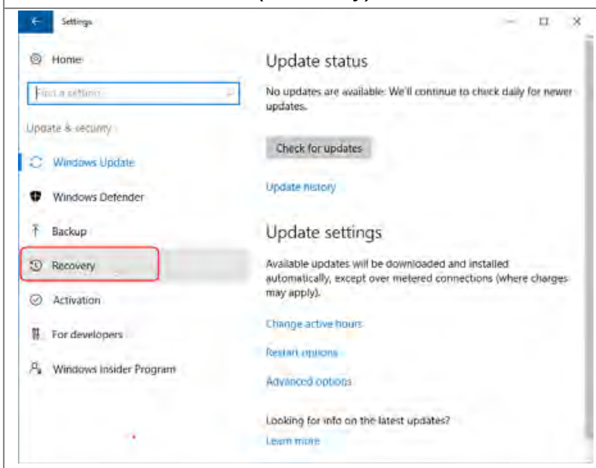
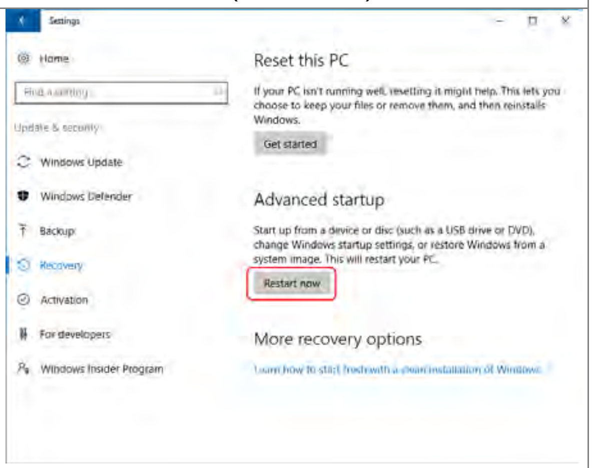
A.3 Installing the USB Driver in Windows 10

Windows 10 driver signature enforcement provides a way to improve the security of the operating system by validating the integrity of a driver or system file each time it is loaded into memory. However since Delta PLC USB driver does not include the driver signature, this section will help users to disable driver signature enforcement functionality in Windows 10 to ensure a success Delta PLC USB installation. This act is only valid for a single time. The setting will return to its original state after restarting.

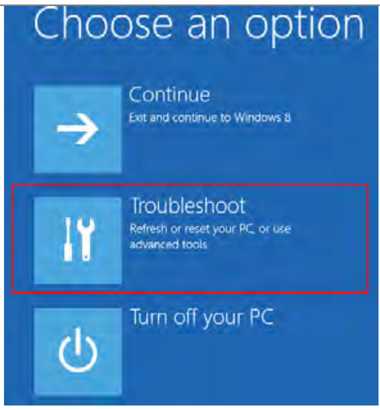
Steps to disable driver signature enforcement in Windows 10:

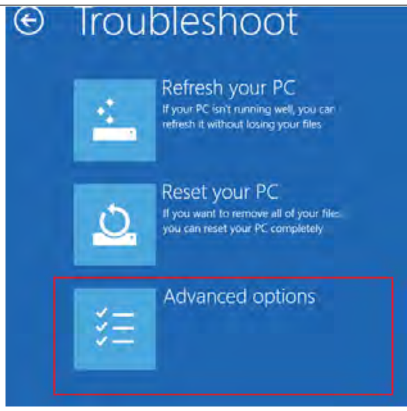
1. Please follow the instructions A (Setting) => B (Update & Security) => C (Recovery) => D (Restart now)

<table>
<tr><th>A (Setting)</th><th>B (Update & Security)</th></tr>
<tr><td>
</td><td>
</td></tr>
</table>

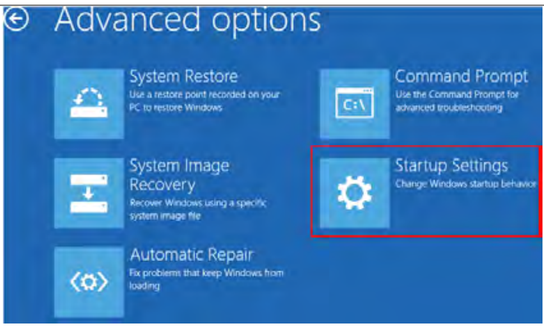
<table>
<tr><th>C (Recovery)</th><th>D (Restart now)</th></tr>
<tr><td>
</td><td>
</td></tr>
</table>

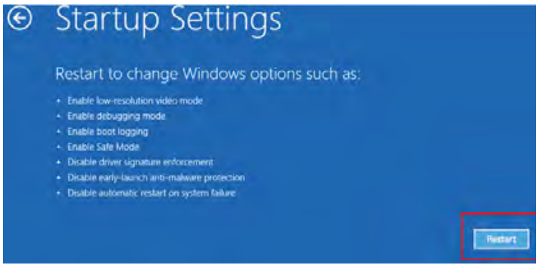
2. After the computer is restarted, select “Troubleshoot” under “Choose an option”. And then select “Advanced options”.



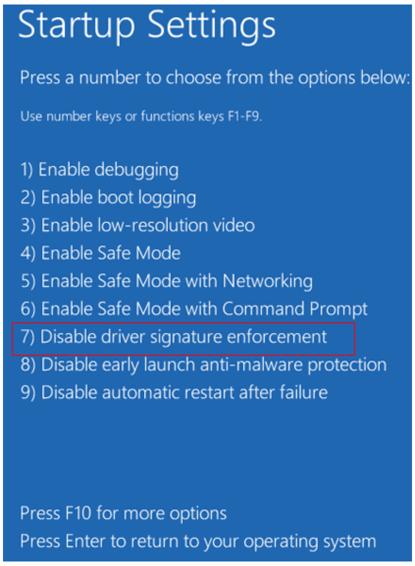


3. From the Advanced options page, select “Startup Settings” to see the Startup Settings. From this page select “Restart” to restart the computer.





4. Press “7” or “F7” to choose “Disable driver signature enforcement” and the system will direct you to the Windows 10 operating page. Users can then install the Delta PLC USB driver now.



5. For the installation of the USB driver, please refer to section A1 for more information.

A.4 Notes on Utilizing USB Communication

- Suggested to use USB communication in the following occasions: PLC program upload / download, PLC parameters monitoring, and firmware upgrade.
- Suggested NOT to use USB communication for applications that require long time communication and without any connection drop.
- When experiencing connection lost, you can unplug the USB cable and then plug it back in and try the communication again.
- For the first time USB communication user, you need to install the USB driver for the DVP Series PLC CPU.
- If the communication is still not working after unplugging and plugging, you need to open the Devices (Windows settings -> Devices) to see if the USB driver is still valid. The USB driver may be lost due to Windows updates. If your USB driver is invalid, install the USB driver again.

A

Memo

Appendix

B

Setting and Using an Ethernet PLC/Module

Contents

B.1 Specifications for an Ethernet PLC/Module

The specifications for a DVP series Ethernet port and the functions of a DVP series Ethernet port are listed below.

Specifications for an Ethernet interface:

Item	Specifications
Interface	RJ-45 with Auto MDI/MDIX
Number of ports	1 Port
Transmission method	IEEE802.3, IEEE802.3u
Transmission cable	Category 5e
Transmission rate	10/100 Mbps Auto-Defect
Protocol	ICMP, IP, TCP, UDP, DHCP, SMTP, NTP, MODBUS TCP

Ethernet functions:

Function		DVP12SE	ES2-E & DVP26SE	DVPEN01-SL	DVP-FEN01 (Function card for a DVP-EH3 series PLC)
MODBUS / TCP	Supporting mode	Server & Client	Server & Client	Server & Client	Server & Client
	Number of servers connected	16	16	16	4
	Number of clients connected	8	8	16	4
EtherNet/IP	Supporting mode	Adapter	Adapter	-	-
	Number of CIP connections	4	8	-	-
	Number of TCP connections	4	4	-	-
	Number of I/O connections	-	8	-	-
Number of connections for data mapping		8	8	24	8
RTU mapping		4	4	4	-
E-mail		-	-	4	-
SNMP		-	-	2	-
IP filter		4	4	8	4

B.2 Ethernet Control Registers

B.2.1 Station Addresses of Ethernet Modules

Model name	Ethernet port in DVP-SE / ES2-E Series	DVPEN01-SL	FEN01 communication card (Applicable to a DVP-EH3 series MPU)
FROM/TO station address	K108	Please refer to Example 1.	K108

Example 1: A DVP-SV series MPU is connected to three left-side communication modules.

MPU/Module name	DVPEN01-SL	DVPCOPM-SL	DVPEN01-SL	DVP28SV11R
FROM/TO station address	K102	K101	K100	--

B.2.2 DVP-SE / ES2-E Series PLC (Ethernet PLC)

In order to control and monitor Ethernet communication, users can read the data in the control registers listed below by means of the instruction FROM, and write data into the control registers listed below by means of the instruction TO. (Please refer to the explanation of API 78 and that of API 79 in chapter 3 for more information about FROM/TO.)

[Note] Please refer to DVPEN01-SL Manual for more information about control registers.

CR number		Attribute	Register name	Description
HW	LW			
#12~#0		-	Reserved	
	#13	R/W	Enabling the data exchange	Users can set CR#13 to “sending the data” or “not sending the data”.
	#14	R/W	Writing function of the RTU mapping	0: The PLC writes data continually. 1: The PLC writes data when the input changes.
	#15	R/W	Enabling flag for RTU mapping	1: Enable; 0: Disable. Default = 1
	#16	R/W	Connection status of RTU mapping slave	b0: Status of RTU slave 1 b1: Status of RTU slave 2 b2: Status of RTU slave 3 b3: Status of RTU slave 4
	#17	R/W	Execution cycle of the data exchange	Time unit: ms
	#18	-	Reserved	
	#19	R	States of the slaves involved in the data exchange	If the value of a bit is 1, an error occurs in the slave corresponding to the bit. b[0:7] indicate the states of the slaves 1~8 involved in the data exchange.
#26~#20		-	Reserved	
#27		R/W	Function code for a data exchange mode	0: The function code for the reading of data and the writing of data is “17”. 1: The function codes for the reading of data is “03, the function code for the writing of a single piece of data is “06”, and the function code for the writing of multiple pieces of data is “10”.
#86~#28		-	Reserved	
	#87	R/W	IP address setting mode	0: Static IP 1: DHCP

CR number		Attribute	Register name	Description
HW	LW			
#89	#88	R/W	IP address	When the IP address is 192.168.1.5, the data in CR#89 is 192.168, and the data in CR#88 is 1.5. One CR takes two bytes, and the decimal number should convert to hexadecimal one. 192 is H'C0, 168 is H'A8; type H'C0A8 in #89 for 192168.
#91	#90	R/W	Mask address	When the mask address is 255.255.255.0 the data in CR#91 is 255.255, and the data in CR#90 is 255.0. One CR takes two bytes, and the decimal number should convert to hexadecimal one.
#93	#92	R/W	Gateway IP address	When the GIP address is 192.168.1.1, the data in CR#89 is 192.168, and the data in CR#88 is 1.1. One CR takes two bytes, and the decimal number should convert to hexadecimal one.
	#94	R/W	Enabling the IP address setting	0: The setting of the IP address is not executed. 1: The setting of the IP address is executed.
	#95	R	IP address setting status	0: The setting is unfinished. 1: The setting is being executed. 2: The setting is complete.
#113~#96		-	Reserved	
	#114	R/W	MPDBUS TCP time-out	Setting up MODBUS TCP time-out (in ms) Default: 3000
	#115	R/W	MODBUS TCP trigger	Setting up whether to send out data in MODBUS TCP mode
	#116	R/W	MODBUS TCP status	Displaying current status of MODBUS TCP mode
#118	#117	R/W	MODBUS TCP destination IP	Setting up destination IP address for MODBUS TCP transaction
	#119	R/W	MODBUS TCP data length	Setting up the data length for MODBUS TCP transaction
#219~#120		R/W	MODBUS TCP data buffer	Data buffer of MODBUS TCP for storing sending/receiving data
#248~#220		-	Reserved	
	#249	R	Sub-version	
	#250	R	Update date	0xC820 (April 8, 2012)
	#251	R	Error code	Displaying the errors. See the error code table for more information.
#255~#252		-	Reserved	
Symbols "R" refers to "able to read data by FROM instrcution"; "W" refers to "able to write data by TO instrcution".				

B.2.3 DVPEN01-SL (Left-side Ethernet Communication Module)

DVPEN01-SL Ethernet communication module				
CR number		Attribute	Register name	Description
HW	LW			
	#0	R	Model name	Set up by the system; read only. Model code of DVPEN01-SL = H'4050
	#1	R	Firmware version	Displaying the current firmware version in hex.
	#2	R	Communication mode	b0: MODBUS TCP mode b1: data exchange mode
	#3	W	E-Mail Event 1 trigger	Set up whether to send E-Mail 1
	#4	W	E-Mail Event 2 trigger	Set up whether to send E-Mail 2
	#5	W	E-Mail Event 3 trigger	Set up whether to send E-Mail 3
	#6	W	E-Mail Event 4 trigger	Set up whether to send E-Mail 4
	#7	R	Status of E-Mail 1, 2	b0 ~ b7: Current status of E-Mail 2 b8 ~ b15: Current status of E-Mail 1

DVPEN01-SL Ethernet communication module				
CR number		Attribute	Register name	Description
HW	LW			
	#8	R	Status of E-Mail 3, 4	b0 ~ b7: Current status of E-Mail 4 b8 ~ b15: Current status of E-Mail 3
	#9	R/W	E-Mail 1 additional message	Filled in by the user, and it will be sent by E-mail.
	#10	R/W	E-Mail 2 additional message	Filled in by the user, and it will be sent by E-mail.
	#11	R/W	E-Mail 3 additional message	Filled in by the user, and it will be sent by E-mail.
	#12	R/W	E-Mail 4 additional message	Filled in by the user, and it will be sent by E-mail.
	#13	R/W	Data exchange trigger	Set up whether to send out data in data exchange mode
	#14	R	Status of data exchange	Displaying current status of data exchange.
	#15	R/W	Enabling flag for RTU mapping	1: Enable; 0: Disable. Default = 0
	#16	R/W	Connection status of RTU mapping slave	b0: Status of RTU slave 1 b1: Status of RTU slave 2 b2: Status of RTU slave 3 b3: Status of RTU slave 4
	#17	R/W	Data exchange cycle time	The control register is used to set data exchange cycle time. The unit used is a millisecond.
#19	#18	R	Error status of slaves in data exchange	0: No error occurs. 1: An error occurs in data exchange. b0~b15 in CR#19: States of slave 1~slave 16. b0~b8 in CR#18: States of slave 17~slave 24.
#24~#20		-	Reserved	
#26	#25	R/W	Destination IP	Destination IP address for data exchange One CR takes two bytes, and the decimal number should convert to hexadecimal one.
	#27	R/W	Function code for a data exchange mode	0: The function code for the reading of data and the writing of data is “17”. 1: The function codes for the reading of data is “03, the function code for the writing of a single piece of data is “06”, and the function code for the writing of multiple pieces of data is “10”.
	#28	R/W	Destination Slave ID	Destination Slave ID for data exchange
#48~#29		R/W	Data transmission buffer	Buffer for transmitted data in data exchange
#68~#49		R	Data receiving buffer	Buffer for received data in data exchange
#80~#69		-	Reserved	
	#81	R/W	Read address for data exchange	Slave transmission buffer address for data exchange
	#82	R/W	Read length for data exchange	Number of registers for read data
	#83	R/W	Received address for data exchange	Buffer address for the receiving Master in data exchange
	#84	R/W	Written-in address for data exchange	Buffer address for the receiving Slave in data exchange
	#85	R/W	Written-in length for data exchange	Number of registers for data transmission
	#86	R/W	Transmission address for data exchange	Master transmission buffer address for data exchange
	#87	R/W	Mode of setting an IP address	0: Static IP address 1: DHCP

DVPEN01-SL Ethernet communication module				
CR number		Attribute	Register name	Description
HW	LW			
#89	#88	R/W	IP address	Setting an IP address One CR takes two bytes, and the decimal number should convert to hexadecimal one.
#91	#90	R/W	Netmask	Setting a netmask One CR takes two bytes, and the decimal number should convert to hexadecimal one.
#93	#92	R/W	Gateway IP address	Setting a gateway IP address One CR takes two bytes, and the decimal number should convert to hexadecimal one.
	#94	R/W	Enabling the setting of an IP address	Executing the setting of an IP address
	#95	R	Status of setting an IP address	Showing the status of setting an IP address 0: The setting of an IP address is successful. 1: The setting of an IP address fails.
#101~#96		-	Reserved	
	#102	R/W	MC Protocol UDP port	Setting the UDP port of an MC protocol data exchange slave (Default value: 1025)
#110~#103		-	Reserved	
	#111	R/W	8-bit processing mode	Setting up MODBUS TCP Master control as 8-bit mode
	#112	R/W	MODBUS TCP Keepalive time	MODBUS TCP Keepalive time (Unit: Second)
	#113	-	Reserved	
	#114	R/W	MODBUS TCP timeout	Setting up MODBUS TCP timeout (Unit: Millisecond)
	#115	R/W	MODBUS TCP trigger	Setting up whether to send out data in MODBUS TCP mode
	#116	R/W	MODBUS TCP status	Displaying current status of MODBUS TCP mode
#118	#117	R/W	MODBUS TCP destination IP	Setting up destination IP address for MODBUS TCP transaction
	#119	R/W	MODBUS TCP data length	Setting up the data length for MODBUS TCP transaction
#219~#120		R/W	MODBUS TCP data buffer	Data buffer of MODBUS TCP for storing sending/receiving data
#248~#220		-	Reserved	
	#251	R	Error code	Displaying the errors. See the error code table for more information.
#255~#252		-	Reserved	
Symbols “R” refers to “able to read data by FROM instrcution”; “W” refers to “able to write data by TO instrcution”.				

B.2.4 DVP-FEN01 (DVP-EH3 Series Ethernet Communication Card)

DVP-FEN01 Ethernet communication card				
CR number		Attribute	Register name	Description
HW	LW			
#0		R	Model code	The model code of DVP-FEN01 is set by its system, and can only be read. It is H’6151.
#1		R	Firmware version	It adopts the hexadecimal system, and the present firmware version is stored in it.
#2~#12		-	Reserved	
#13		R/W	Enabling the data exchange	Users can set CR#13 to “sending the data” or “not sending the data”.
#16~#14		-	Reserved	
#17		R/W	Execution cycle of the data exchange (ms)	

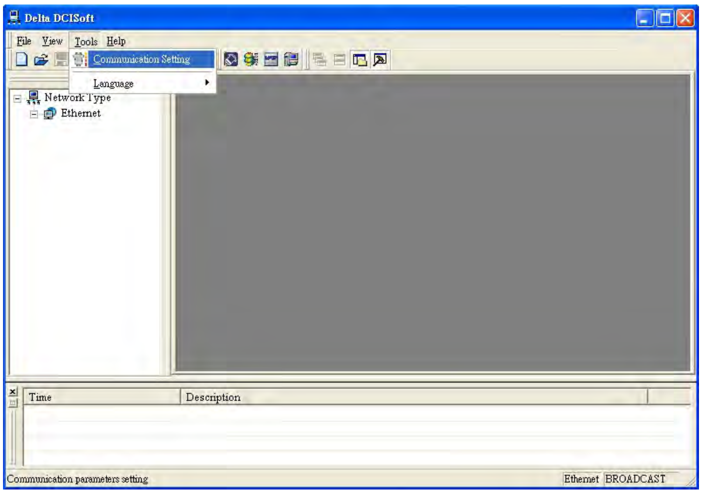
DVP-FEN01 Ethernet communication card				
CR number		Attribute	Register name	Description
HW	LW			
#18		-	Reserved	
#19		R	States of the slaves involved in the data exchange	b[0:7] indicate the states of the slaves 1~8 involved in the data exchange.
#26~#20		-	Reserved	
#27		R/W	Function code for a data exchange mode	0: The function code for the reading of data and the writing of data is “17”. 1: The function code for the reading of data is “03, the function code for the writing of a single piece of data is “06”, and the function code for the writing of multiple pieces of data is “10”.
#86~#28		-	Reserved	
#87		R/W	IP address setting mode	0: Static IP 1: DHCP
#89	#88	R/W	IP address	When the IP address is 192.168.1.5, the data in CR#89 is 192.168, and the data in CR#88 is 1.5. One CR takes two bytes, and the decimal number should convert to hexadecimal one. 192 is H’C0, 168 is H’A8; type H’C0A8 in #89 for 192168.
#91	#90	R/W	Mask address	When the mask address is 255.255.255.0 the data in CR#91 is 255.255, and the data in CR#90 is 255.0. One CR takes two bytes, and the decimal number should convert to hexadecimal one.
#93	#92	R/W	Gateway IP address	When the GIP address is 192.168.1.1, the data in CR#89 is 192.168, and the data in CR#88 is 1.1. One CR takes two bytes, and the decimal number should convert to hexadecimal one.
#94		R/W	Enabling the IP address setting	0: The setting of the IP address is not executed. 1: The setting of the IP address is executed.
#95		R	IP address setting status	0: The setting is unfinished. 1: The setting is being executed. 2: The setting is complete.
#250~#96		-	Reserved	
#251		R	Error status	bit 0: The network is unconnected. bit 3: CR#13 is set to “sending the data”, but the data exchange is not enabled. bit 8: DHCP does not acquire the correct network parameter.
#255~#252		-	Reserved	
Symbols “R” refers to “able to read data by FROM instrcution”; “W” refers to “able to write data by TO instrcution”.				

B.3 Searching for an Ethernet PLC

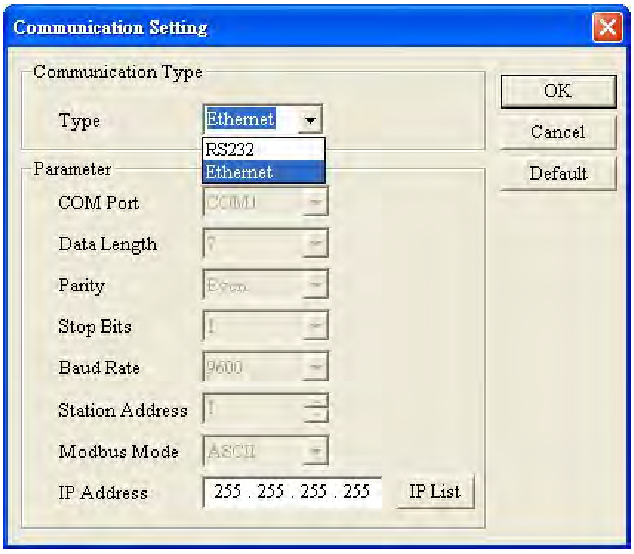
This section introduces how to search for and set an Ethernet PLC by DCISoft. Before you start a setup page, you have to select **Ethernet** in the **Communication Setting** window. Next, you can search by a broadcast, or an IP address. An Ethernet PLC is set up by UDP port 20006; therefore, you have to be aware of the relevant settings of the firewall.

B.3.1 Communication setting

(1) Start DCISoft in your PC, and click **Communication Setting** on the **Tools** menu.

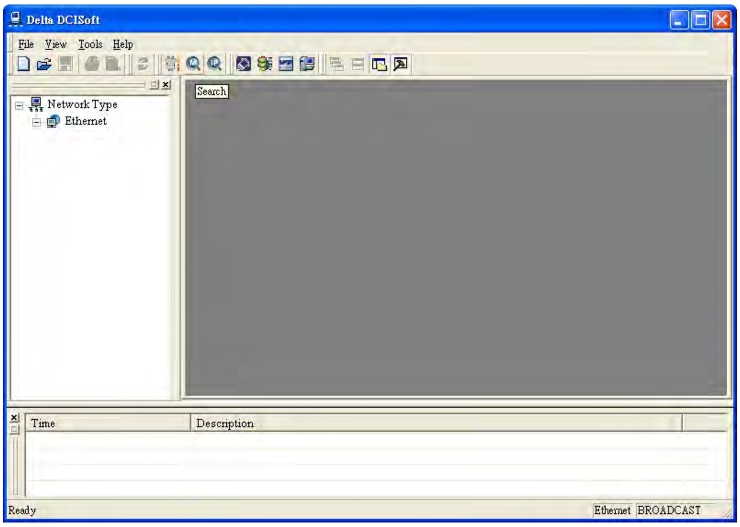


(2) Select **Ethernet** in the **Type** drop-down list box.

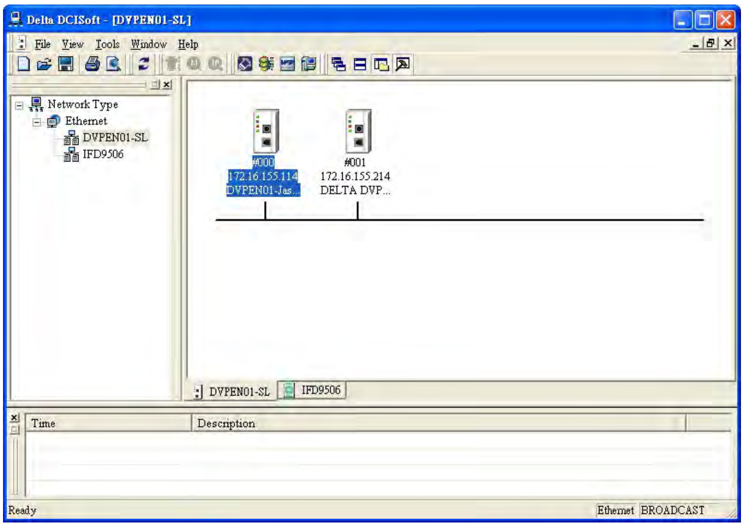


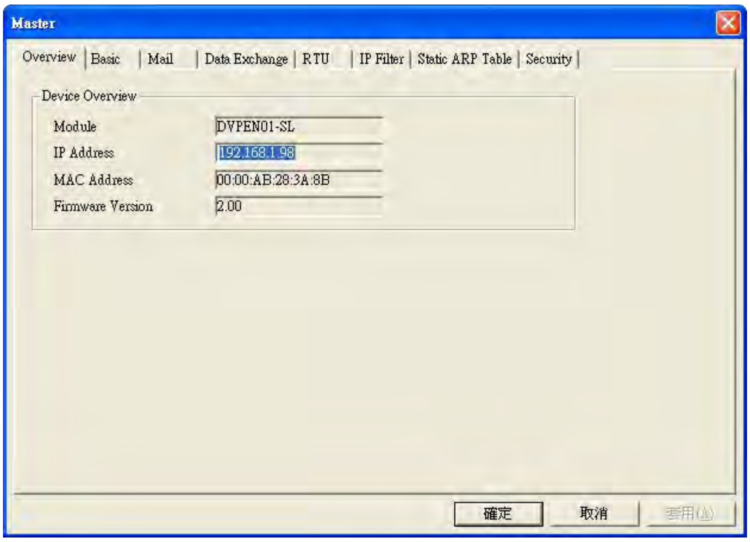
B.3.2 Broadcast Search

(1) Click **Search** on the toolbar in DCISoft to search for all Delta Ethernet products on the network. The window on the left hand side shows the models found, and the window on the right hand side displays the device list of all models.



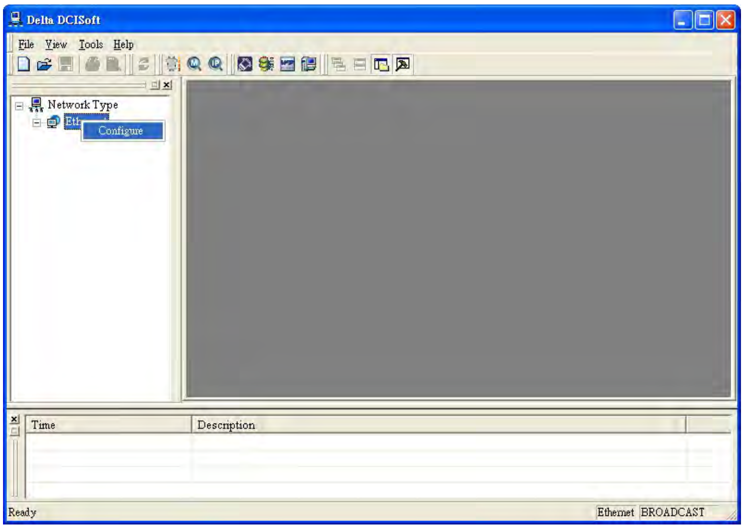
(2) Click a model on the left hand side, and you will see the device list of the model selected on the right hand side. Click the device to be set up to enter the setup page.



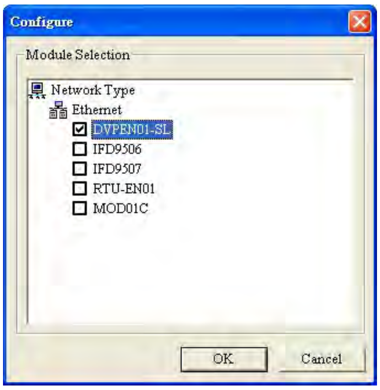


B.3.3 Searching for a Model Specified

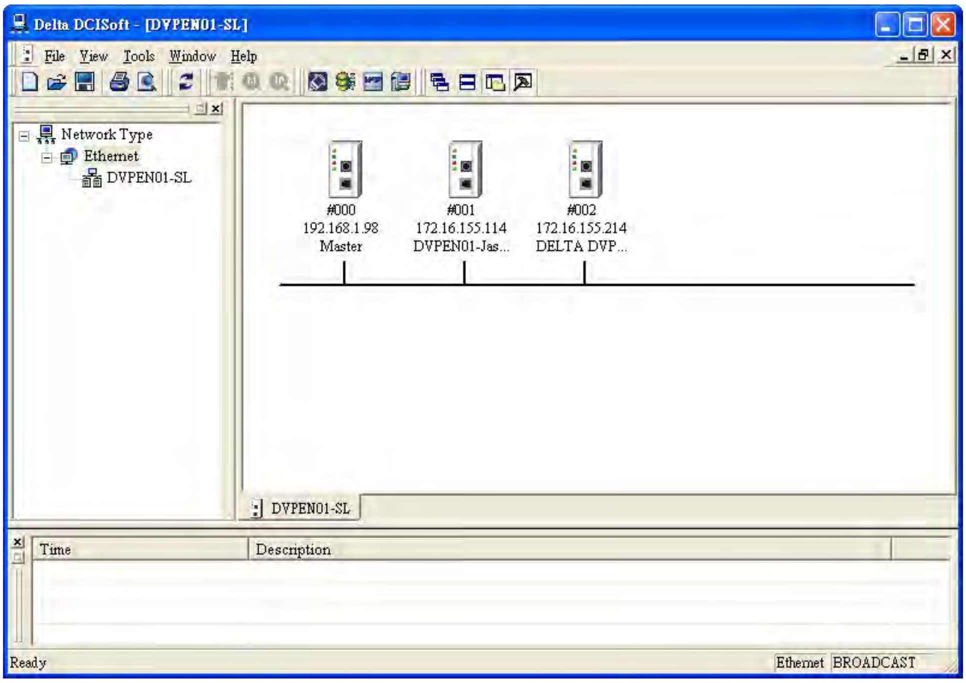
(1) Right-click **Ethernet** in the left hand side window, and click **Configure** to designate a model to be searched for.



(2) After users select a model which will be searched for, they can click **OK** to auto-search for the model on the network. In the window shown below, the **DVPEN01-SL** checkbox is selected.

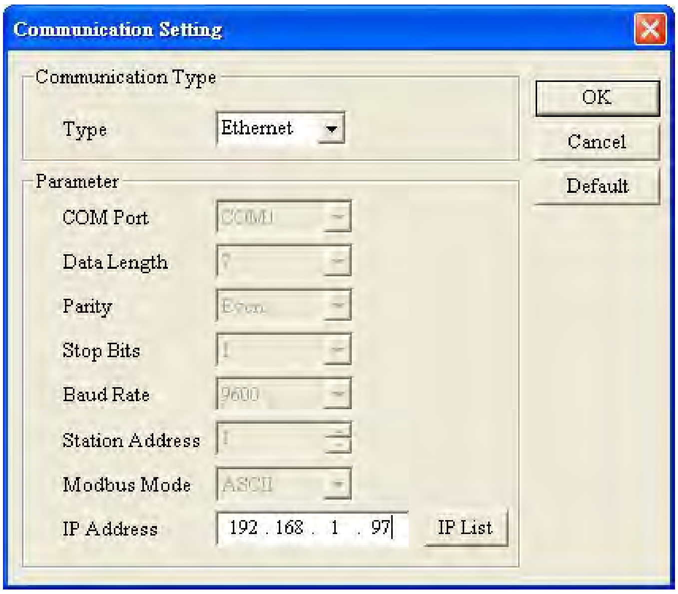


(3) A list of specified devices is in the window. If the users have selected several models, they can view these models.

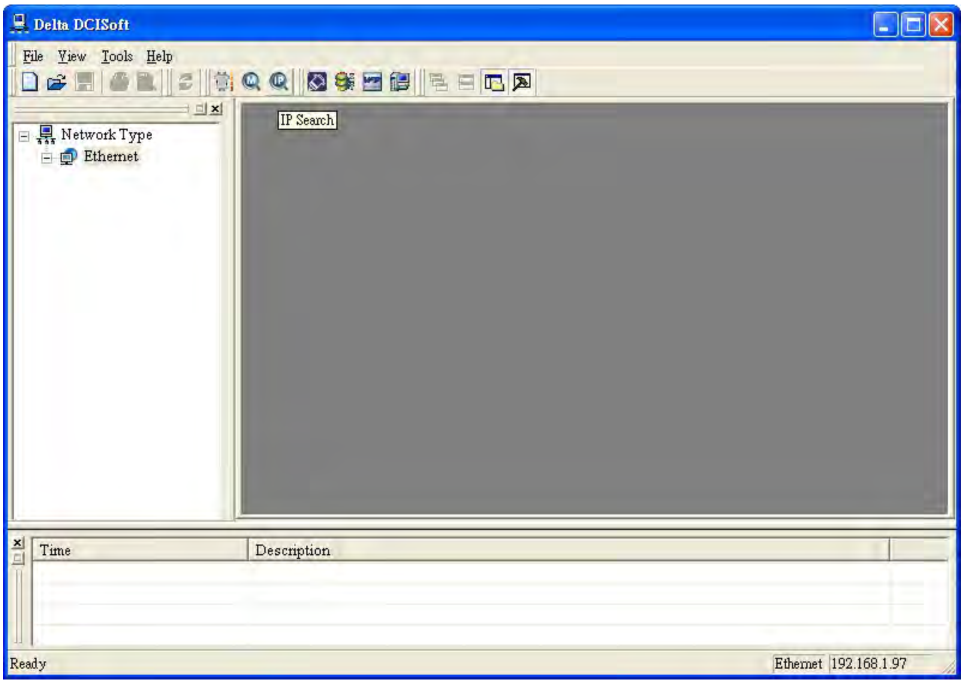


B.3.4 Searching by an IP Address

(1) Select **Ethernet** in the **Type** drop-down list box, type an IP address in the **IP Address** box, and click **OK**.



(2) Click **Search** on the toolbar to start searching for the designated IP address.



(3) The model found will be displayed in the right hand side window. Double-click the device to enter the setup page.

B.4 Data Exchange

A Delta Ethernet master can read/write data from/into a slave by means of instructions. It can also read/write data from/into a slave by means of tables. The number of data exchanges that models provide is different. Please refer to section B.1 for more information about the number of devices exchanging data.

(1) Enable:

Users can enable or disable a data exchange. After a data exchange is enabled, the data will be exchanged.

(2) Enable Condition:

You can select **Always Enable** or **Program Control**. If **Always Enable** is selected, DVPEN01-SL will execute data exchange continuously until the setting in DCISoft is changed. If **Program Control** is selected, DVPEN01-SL will execute data exchange according to the program setting. The internal registers in different models used to enable data exchanges are different. Please refer to section B.2 for more information.

(In DVPEN01-SL, the data exchange is executed if CR#13=2, and the data exchanged is not executed if CR#13=0.)

(3) Station Address-IP Address:

You have to type the IP address of a slave. If the IP address of a slave is 192.168.0.1, and the station number of the slave is 1, you can type 1 in the first **Station Address** cell, select the box in the first **Enable** cell, and type 192.168.0.1 in the first **IP Address** cell.

(4) Master Device, Slave Device, and Quantity:

Reading (←): Initial reception register in a master←Initial transmission register in a slave

Writing (→): Initial transmission register in a master→ Initial reception register in a slave

If a data exchange is enabled, the Ethernet PLC will write data, and then read data.

Quantity: A slave station can send 100 pieces of data at most and receive 100 pieces of data at most simultaneously.

※ If a device which is not a Delta PLC is connected, users can type a hexadecimal four-digit MODBUS absolute position in the **Slave Device** cell.

B.5 EtherNet/IP List

EtherNet/IP is a communication protocol defined by ODVA, and is different from the Ethernet mentioned in the previous sections. DVP-12SE (whose version are 1.20 or above), DVP26SE and ES2-E Series PLCs support the EtherNet/IP slave communication protocol. The other DVP series PLCs can communicate with products related to EtherNet/IP through IFD9507 (an EtherNet/IP-MODBUS converter). The EtherNet/IP objects which are supported are described below.

B.5.1 EtherNet/IP Information Supported by DVP-SE / ES2-E Series PLCs

(1) Object list

	DVP12SE		ES2-E & DVP26SE		
Object Name	Class Code	#of Instance	Class Code	#of ES2-E Instance	#of 26SE Instance
Identity	0x01	7	0x01	8	8
Message Router	0x02	NA	0x02	2	2
Assembly	0x04	7	0x04	8	8
Connection Manager	0x06	NA	0x06	NA	NA
X input	0x64	256	0x350	256	256
Y output	0x65	256	0x351	256	256
T Timer	0x66	256	0x355	256	256
M Relay	0x67	4096	0x353	4096	4096
C Counter	0x68	256	0x356	256	256
D Register	0x69	12000	0x352	10000	12000
S Relay	-	-	0x354	1024	1024
TCP/IP Interface	0xF5	6	0xF5	7	7
Ethernet Link	0xF6	3	0xF6	5	5

(2) Data types

8-bit	16-bit	32-bit	64-bit
USINT	WORD	UDINT	ULINT
SINT	UINT	DWORD	LINT
BYTE	INT	DINT	

(3) Error codes

Value	Name	Description
0	Success	Success
0x01	Connection Failure	The forwarding function can not be enabled.
0x04	Path Segment Error	The segment type is not supported. (ref. V1 C-1.4)
0x05	Path Destination Unknown	The instance is not supported.
0x08	Service Not Supported	The service (Get or Set) is not supported.
0x09	Invalid Attribute Value	The value written is incorrect.
0x0E	Attribute Not Settable	The setting of the attribute is not allowed.
0x13	Not Enough Data	The length of the data written is too short.
0x14	Attribute Not Supported	The attribute is not supported.
0x15	Too Much Data	The length of the data written is too long.
0x16	Object Not Exist	The object is not supported.
0x20	Invalid Parameter	The service parameter is not supported. (ref. V1 5-2.3.1)
0x26	Path Size Invalid	Incorrect item length

B.5.2 EtherNet/IP Objects Supported by DVP-SE / ES2-E Series PLCs

(1) Identity Object (0x01)

Instance: 0x01

Attribute	Name	Access	Data Type	Value
0x01	Vendor ID	Get	UINT	799 (Delta Electronics, inc.)
0x02	Device Type	Get	UINT	14 (Programmable Logic Controller)
0x03	Product Code	Get	UINT	Product code
0x04	Revision	Get	STRUCT of:	Device version; Display as Major.Minor
	Major		USINT	Major: 0x01~0x7F
	Minor		USINT	Minor: 0x01~0xFF
0x05	Status	Get	WORD	0 (Owned)
0x06	Serial Number	Get	UDINT	Serial number: the last 4 digits of the MAC address, ab：cd
0x07	Product Name	Get	SHORT_STRING	Module Name
0x08	State	Get	UINT	0x03 (ES2-E & DVP26SE Only)

(2) Message Router (0x02)

Instance: 0x01

Attribute	Name	Access	Data Type	Value
0x02	Number Available	Get	UINT	0
0x03	Number Active	Get	UINT	0

(3) Assembly (0x04)

Explicit message

DVP12SE: Conformance Test is not supported.

Instance	Attribute	Name	Access	Data Type	Data
0x65	0x03	D Block 1	Set	10 words	D500~D509
0x66		D Block 2	Set	30 words	D510~D539
0x67		D Block 3	Set	60 words	D540~D599
0x68		D Block 4	Set	100 words	D600~D699
0x69		D Block 5	Set	100 words	D700~D799
0x6A		D Block 6	Set	100 words	D800~D899
0x6B		D Block 7	Set	100 words	D900~D999

ES2-E & DVP26SE

Connection No.	Name	Instance Attribute	Data Length	Default
Connection 1	Input	0x65	100 words	D0 ~ D99
	Output	0x64	100 words	D3000 ~ D3099
	Configuration	0x80	8 words	Refer to Config Data below
Connection 2	Input	0x67	100 words	D100 ~ D199
	Output	0x66	100 words	D3100 ~ D3199
	Configuration	0x81	8 words	Refer to Config Data below
Connection 3	Input	0x69	100 words	D200 ~ D299
	Output	0x68	100 words	D3200 ~ D3299
	Configuration	0x82	8 words	Refer to Config Data below
Connection 4	Input	0x6B	100 words	D300 ~ D399

	Output	0x6A	100 words	D3300 ~ D3399
	Configuration	0x83	8 words	Refer to Config Data below
Connection 5	Input	0x6D	100 words	D400 ~ D499
	Output	0x6C	100 words	D3400 ~ D3499
	Configuration	0x84	8 words	Refer to Config Data below
Connection 6	Input	0x6F	100 words	D500 ~ D599
	Output	0x6E	100 words	D3500 ~ D3599
	Configuration	0x85	8 words	Refer to Config Data below
Connection 7	Input	0x71	100 words	D600 ~ D699
	Output	0x70	100 words	D3600 ~ D3699
	Configuration	0x86	8 words	Refer to Config Data below
Connection 8	Input	0x73	100 words	D700 ~ D799
	Output	0x72	100 words	D3700 ~ D3799
	Configuration	0x87	8 words	Refer to Config Data below

Config Data

Parameter	Data Type	Data	Default
Input Device Type	INT	Input (T to O) device type 0: D register	0
Input Device Quantity	INT	Input (T to O) device quantity	100
Input Device Address	DINT	Input (T to O) device address 0: D0, 1: D1…	Refer to Connection 1 ~ 8
Output Device Type	INT	Output (O to T) device type 0：D register	0
Output Device Quantity	INT	Output (O to T) device quantity	100
Output Device Address	DINT	Output (O to T) device address 0: D0, 1: D1…	Refer to Connection 1 ~ 8

(4) X input

DVP12SE: (0x64)

Instance	Attribute	Name	Access	Data Type
1	0x64	X0	Get	BYTE
2	0x64	X1	Get	BYTE
……				
256	0x64	X377	Get	BYTE

ES2-E & DVP26SE: (0x350)

Instance	Attribute	Name	Access	Data Type
1	0	X0	Get	BOOL
1	1	X1	Get	BOOL
……				
1	255	X377	Get	BOOL

(5) Y output

DVP12SE: (0x65)

Instance	Attribute	Name	Access	Data Type
1	0x64	Y0	Set	BYTE (0x00 or 0x01)
2	0x64	Y1	Set	BYTE (0x00 or 0x01)
……				
256	0x64	Y377	Set	BYTE (0x00 or 0x01)

ES2-E & DVP26SE: (0x351)

Instance	Attribute	Name	Access	Data Type
1	0	Y0	Set	BOOL
1	1	Y1	Set	BOOL
……				
1	255	Y377	Set	BOOL

(6) T timer

DVP12SE: (0x66)

Instance	Attribute	Name	Access	Data Type
1	0x64	T0	Set	INT
2	0x64	T1	Set	INT
……				
256	0x64	T255	Set	INT

Instance	Attribute	Name	Access	Data Type
1	0x65	T0	Set	BYTE (0x00 or 0x01)
2	0x65	T1	Set	BYTE (0x00 or 0x01)
……				
256	0x65	T255	Set	BYTE (0x00 or 0x01)

ES2-E & DVP26SE: (0x355)

Instance	Attribute	Name	Access	Data Type
1	0	T0 Bit	Set	BOOL
1	1	T1 Bit	Set	BOOL
……				
1	255	T255 Bit	Set	BOOL

Instance	Attribute	Name	Access	Data Type
2	0	T0 Register	Set	INT
2	1	T1 Register	Set	INT
……				
2	255	T255 Register	Set	INT

(7) M Relay

DVP12SE: (0x67)

Instance	Attribute	Name	Access	Data Type
1	0x64	M0	Set	BYTE
2	0x64	M1	Set	BYTE
……				
4096	0x64	M4095	Set	BYTE

ES2-E & DVP26SE: (0x353)

Instance	Attribute	Name	Access	Data Type
1	0	M0	Set	BOOL
1	1	M1	Set	BOOL
……				
1	4095	M4095	Set	BOOL

(8) C counter

DVP12SE: (0x68)

Instance	Attribute	Name	Access	Data Type
1	0x64	C0	Set	INT
2	0x64	C1	Set	INT
……				
200	0x64	C199	Set	INT

Instance	Attribute	Name	Access	Data Type
201	0x64	C200	Set	DINT
202	0x64	C201	Set	DINT
……				
256	0x64	C255	Set	DINT

Instance	Attribute	Name	Access	Data Type
1	0x65	C0	Set	BYTE (0x00 or 0x01)
2	0x65	C1	Set	BYTE (0x00 or 0x01)
……				
256	0x65	C255	Set	BYTE (0x00 or 0x01)

ES2-E (0x356)

Instance	Attribute	Name	Access	Data Type
1	0	C0 Bit	Set	BOOL
1	1	C1 Bit	Set	BOOL
……				
1	255	C255 Bit	Set	BOOL

Instance	Attribute	Name	Access	Data Type
2	0	C0 Register	Set	INT
2	1	C1 Register	Set	INT
……				
2	199	C199 Register	Set	INT
2	200	C200 Register	Set	DINT
……				
2	255	C255 Register	Set	DINT

(9) D Register

DVP12SE: (0x69)

Instance	Attribute	Name	Access	Data Type
1	0x64	D0	Set	INT
2	0x64	D1	Set	INT
……				
12000	0x64	D11999	Set	INT

ES2-E: (0x352)

Instance	Attribute	Name	Access	Data Type
1	0	D0	Set	INT
1	1	D1	Set	INT
……				
1	9999	D9999	Set	INT

DVP26SE: (0x352)

Instance	Attribute	Name	Access	Data Type
1	0	D0	Set	INT
1	1	D1	Set	INT
……				
1	11999	D11999	Set	INT

(10) TCP/IP Interface Object (0xF5)

Instance: 0x01

Attribute	Name	Access	Data Type	Value
0x01	Status	Get	DWORD	0x00000001UL
0x02	Configuration Capability	Get	DWORD	DVP12SE = 0x00000014UL (DHCP client, Configuration Settable) ES2-E & DVP26SE = 0x00000015UL (DHCP client, BOOTP Client, Configuration Settable)
0x03	Configuration Control	Get	DWORD	Static IP: 0U BOOTP: 0x01U(ES2-E Only) DHCP: 0x02U
0x04	Physical Link Object:	Get	STRUCT of:	
	Path Size		UINT	
	Path		Padded EPATH	
0x05	Interface Configuration:	Set	STRUCT of:	
	IP Address		UDINT	
	Network Mask		UDINT	
	Gateway Address		UDINT	
	Name Server		UDINT	
	Name Server 2		UDINT	
	Domain Name		STRING	
0x06	Host Name	Get	STRING	DVP12SE or ES2-E
0x0D	Encapsulation Inactivity Timeout	Set	UINT	Keep Alive Timeout 120 s

(11) Ethernet Link Object (0xF6)

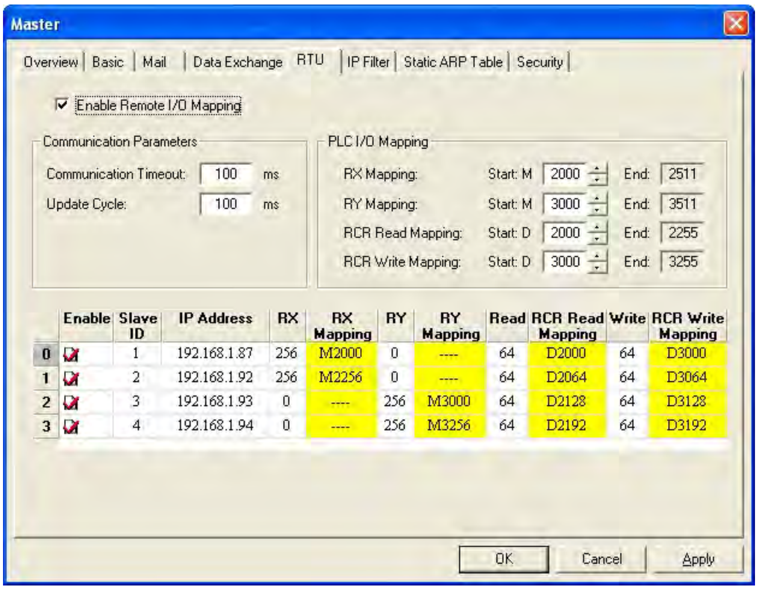
Instance: 0x01

Attribute	Name	Access	Data Type	Value
0x01	Interface Speed	Get	UDINT	10 or 100 Mbps
0x02	Interface Flag	Get	UDINT	Bit 0: Link Status Bit 1: Half/Full Duplex
0x03	MAC Address	Get	USINT[6]	
0x0A	Interface Label	Get	SHORT_ STRING	Define Ethernet port name
0x0B	Interface Capability	Get	STRUCT of:	01 31
	Capability Bits		DWORD	00 00 00 07
	Speed/Duplex Array Count		USINT	04
	Interface Speed 1		UINT	00 0A
	Interface Duplex Mode 1		USINT	00
	Interface Speed 2		UINT	00 0A
	Interface Duplex Mode 2		USINT	01
	Interface Speed 3		UINT	00 64
	Interface Duplex Mode 3		USINT	00
	Interface Speed 4		UINT	00 64
	Interface Duplex Mode 4		USINT	01

B.6 RTU Mapping

Users can connect the Delta network product DVPEN01-SL/DVP-SE/ES2-E to RTU-EN01 by means of RTU mapping. After the users finish setting mapping information, they can operate RTU-EN01 by means of corresponding bits (M devices) and registers (D devices) in DVPEN01-SL/DVP-SE/ES2-E instead of communication programs.

B.6.1 Setting the RTU Mapping



(1) **Enable Remote I/O Mapping**

Users can select the **Enable Remote I/O Mapping** checkbox. After the checkbox is selected, the network module used will be mapped onto RTU-EN01 according to the data set.

(2) **Communication Parameters**

Users can enter a time interval in the **Communication Timeout** box, and a cycle in the **Update Cycle** box.

(3) **PLC I/O Mapping**

Users can set the bit devices and the registers which correspond to digital inputs, digital outputs, and analog registers on RTU-EN01. The bit devices set start from M2000. The registers used for the reading of data start from D2000, and the registers used for the writing of data start from D3000. The software automatically calculates end addresses according to the numbers set.

(4) Setting the remote device mapping

After users check an **Enable** cell, they have to enter the station address of RTU-EN01, an IP address, the number of digital inputs, the number of digital outputs, the number of registers used for the reading of data, and the number of registers used for the writing of data. DVPEN01-SL can be mapped onto four slaves. The maximum number of digital inputs used for mapping, the maximum number of digital outputs used for mapping, the maximum number of registers used for mapping are described below.

Digital I/O (RX+RY): 256

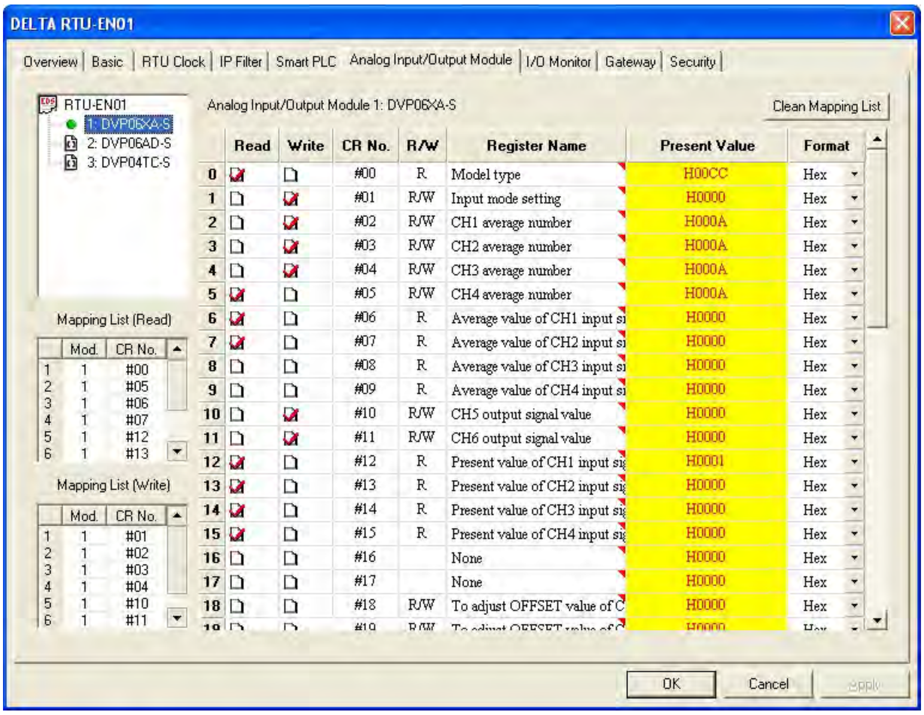
Analog (Reading) register: 64

Analog (Writing) register: 64

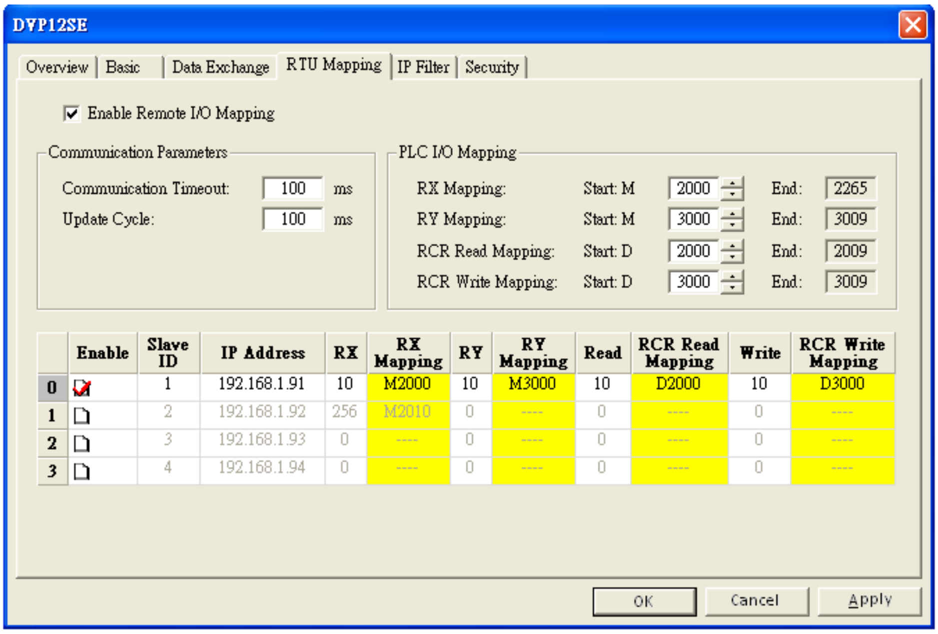
B.6.2 Application of the RTU Mapping

Application	Using RTU mapping to read data from/write data into remote digital inputs/outputs and analog input/output registers DVP-SE/ES2-E → RTU-EN01+DVP06XA+DVP16SP
Network environment	(1) Use a static IP address. (2) IP address of DVP-SE: 192.168.1.90 (3) IP address of RTU-EN01: 92.168.1.91 (4) Use DCISoft for RTU-EN01, and check 10 pieces of mapping data for reading and 10 pieces of mapping data for writing. (5) Set a start RX address, a start RY address, a start RCR (reading) address, and a start RCR (writing address) in DVP-SE, and set corresponding numbers. (6) Enable the mapping function in DVP-SE. Use M2000 and D2000 in DVP-SE to read values from RTU-EN01, and use M3000 and D3000 to write values into RTU-EN01.

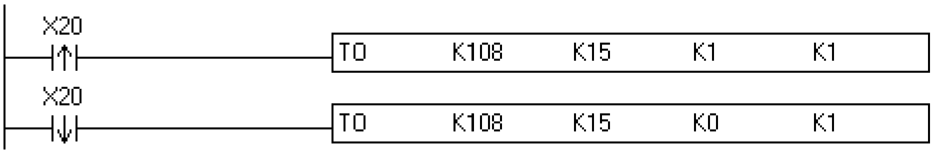
1. Please refer to section B.6.1 for more information about setting communication.
2. Use DCISoft for RTU-EN01 to set mapping control registers used for reading/writing.



3. Use DCISoft for DVP-SE to set start addresses and numbers. (RX: M2000~M2009; RY: M3000~M3009; RCR (Reading): D2000~D2009; RCR (Writing): D3000~D3009)



4. Edit a ladder diagram, and download it to DVP-SE. The program edited is like the one shown below.



Description:

(1) Enabling mapping: CR15=1

(2) Disabling mapping: CR15=0

(3) After CR#15 is enabled, M2000~M2009 and D2000~D2009 will be used to read data, and present values will be read before M3000~M3009 and D3000~D3009 are used to write data.

※ During the execution of mapping, other devices can not be used to modify the values in mapping registers.

※ If DVPEN01-SL is used, K108 will be changed to the number assigned to DVPEN01-SL. If DVPEN01-SL is the first module connected to the left side of a PLC, K108 will be changed to K100.

MEMO

Appendix

Inforamation about TP Series Text Panels

Contents

C.1 TP Memory Map

Specifications					
Control Method				Stored program, cyclic scan system	
I/O Processing Method				Batch processing method (when END instruction is executed)	
Execution Speed				LD instructions – 0.54μs, MOV instructions – 3.4μs	
Program language				Instruction List + Ladder + SFC	
Program Capacity				TP70P-RM0: 2k, TP70P: 4k, TP04P: 8ksteps	
Relay (Bit)	X	External inputs		X0~X7;X10~X17	(*4)
	Y	External outputs		Y0~Y7;Y10~Y17	
	M	Auxiliary relay	General	M0~M511, 512 points max, (*1) M768~M999, 232 points max, (*1) M2000~M2047, 48 points max, (*1)	Total 4096 points
			Latched	M512~M767, 256 points max, (*2) M2048~M4095, 2048 points max, (*2)	
			Special	M1000~M1999, 1000 points, some are latched	
	T	Timer	100ms (M1028=ON, T64~T126: 10ms)	T0~T126, 127 points, (*1) T128~T183, 56 points, (*1)	Total 251 points
				T184~T199 for Subroutines, 16 points, (*1)	
				T250~T255 (accumulative), 6 points, (*1)	
			10ms (M1038=ON, T200~T245: 1ms)	T200~T239, 40 points, (*1)	
				T240~T245 (accumulative), 6 points, (*1)	
Relay (Bit)	C	Counter	16-bit count up	C0~C111, 112 points, (*1) C128~C199, 72 points, (*1)	Total 140 points
				C112~C127, 16 points, (*2)	
			32-bit count up/down	C200~C223, 24 points, (*1)	
				C224~C232, 9 points, (*2) C233~C234, 2 points, (*2) C237~C250, 14 points, (*2) C252~C255, 3 points, (*2)	
			32bit high-speed count up/down	C235, C236 , 1 phase 1 input, 2 points (*2)	Total 3 points
				C251, 2 phase 2 input, 1 point (*2) Only available for twofold and fourfold frequencies	
	S	Step point	Initial step point	S0~S9, 10 points, (*2)	Total 1024 points
			Zero point return	S10~S19, 10 points (use with IST instruction), (*2)	
			Latched	S20~S127, 108 points, (*2)	
			General	S128~S911, 784 points, (*1)	
			Alarm	S912~S1023, 112 points, (*2)	
Word Register	T	Current value		T0~T255, 256 words	
	C	Current value		C0~C199, 16-bit counter, 200 words	
				C200~C254, 32-bit counter, 55 words	
	D	Data register	General	D0~D407, 408 words, (*1) D600~D999, 400 words, (*1) D3920~D3999, 80 words, (*1)	Total 5000
			Latched	D408~D599, 192 words, (*2) D2000~D3919, 1920 words, (*2)	

Specifications				
			Special	D1000~D1999, 1000 words, some are latched D4000~D4999, 1000 words, (*3)
			Index	E0~E7, F0~F7, 16 words, (*1)
Pointer	N	Master control loop		N0~N7, 8 points
	P	Pointer		P0~P255, 256 points
	I	Interrupt Service	External interrupt	I000/I001(X0), I100/I101(X1) (01: rising-edge trigger , 00: falling-edge trigger)
			Timer interrupt	NA
			High-speed counter interrupt	I010,1 point
			Communication interrupt	I150(COM2), 1point, (*3)
Constant	K	Decimal		K-32,768 ~ K32,767 (16-bit operation) K-2,147,483,648 ~ K2,147,483,647 (32-bit operation)
	H	Hexadecimal		H0000 ~ HFFFF (16-bit operation) H00000000 ~HFFFFFFFF (32-bit operation)
Serial ports				COM1: USB (Slave); used for programming communication COM2: RS-485 for PLC (Master/Slave) COM3: RS-485 for TP (Master/Slave)
Real Time Clock				Year, Month, Day, Week, Hours, Minutes, Seconds

Notes:

1. Non-latched area cannot be modified.
2. Latched area cannot be modified.
3. COM2: RS485 for PLC communication.
4. There are 16-point models, and 32-point models. Extension modules are not supported.

C.2 Special Data Register

The types and functions of special registers (special D) are listed in the table below. Care should be taken that some registers of the same No. may bear different meanings in different series MPUs. Special M and special D marked with "*" will be further illustrated in 2.13. Columns marked with "R" refers to "read only", "R/W" refers to "read and write", "-" refers to the status remains unchanged and "#" refers to that system will set it up according to the status of the PLC. For detailed explanation please also refer to the table below.

Special D	Content	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1000*	Setting value of the watchdog timer (WDT) (Unit: 1ms)	300	-	-	R/W	NO	300
D1001	Displaying the firmware version of TP (For example, the firmware version is 1.0 if the value in D1001 is HXX10.)	-	-	-	R	NO	#
D1002*	Program capacity	7920	-	-	R	NO	7920
D1003	Sum of the PLC internal program memory	-	-	-	R	YES	7920
D1004*	Syntax check error code	0	0	-	R	NO	0
D1008*	Step address when WDT is ON	0	-	-	R	NO	0
D1009	Number of LV (Low voltage) signal occurrence	-	-	-	R	YES	0
D1010*	Current scan time (Unit: 0.1ms)	#	#	#	R	NO	0
D1011*	Minimum scan time (Unit: 0.1ms)	#	#	#	R	NO	0
D1012*	Maximum scan time (Unit: 0.1ms)	#	#	#	R	NO	0
D1015*	Value of accumulative high-speed timer (0~32,767, unit: 0.1ms)	0	-	-	R/W	NO	0
D1018*	π PI (Low word)	H'0FDB	H'0FDB	H'0FDB	R/W	NO	H'0FDB
D1019*	π PI(High word)	H'4049	H'4049	H'4049	R/W	NO	H'4049
D1022	Counting mode selection (twofold frequency / fourfold frequency) for AB phase counter (From X0, X1 input)	4	-	-	R/W	NO	4
D1025*	Code for communication request error	0	-	-	R	NO	0
D1028	Index register E0	0	-	-	R/W	NO	0
D1029	Index register F0	0	-	-	R/W	NO	0
D1036*	COM1 (RS-232) communication protocol	H'86	-	-	R/W	NO	H'86
D1038*	1. Delay time setting for data response when PLC is SLAVE in COM2 / COM3 RS-485 communication. Range: 0 ~ 10,000 (unit: 0.1ms). 2. By using PLC LINK in COM2 (RS-485), D1038 can be set to send next communication data with delay. Range: 0 ~ 10,000 (Unit: one scan cycle)	-	-	-	R/W	NO	0
D1039*	Fixed scan time (ms)	0	-	-	R/W	NO	0
D1040	No. of the 1st step point which is ON.	0	-	-	R	NO	0
D1041	No. of the 2nd step point which is ON	0	-	-	R	NO	0
D1042	No. of the 3rd step point which is ON.	0	-	-	R	NO	0

<table>
<tr><th>Special D</th><th>Content</th><th>OFF ⇩ ON</th><th>STOP ⇩ RUN</th><th>RUN ⇩ STOP</th><th>Attrib.</th><th>Latch -ed</th><th>Default</th></tr>
<tr><td>D1043</td><td>No. of the 4th step point which is ON</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1044</td><td>No. of the 5th step point which is ON.</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1045</td><td>No. of the 6th step point which is ON</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1046</td><td>No. of the 7th step point which is ON.</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1047</td><td>No. of the 8th step point which is ON</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1049</td><td>No. of alarm which is ON</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1050
↓
D1055</td><td>Converted data for Modbus communication data processing. PLC automatically converts the ASCII data in D1070~D1085 into Hex data and stores the 16-bit Hex data into D1050~D1055</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1062*</td><td>Average number of times an analog signal is input</td><td>-</td><td>-</td><td>-</td><td>R/W</td><td>YES</td><td>2</td></tr>
<tr><td>D1067*</td><td>Error code for program execution error</td><td>0</td><td>0</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1068*</td><td>Address of program execution error</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1070
↓
D1085</td><td>Feedback data (ASCII) of Modbus communication. When PLC's RS-485 communication instruction receives feedback signals, the data will be saved in the registers D1070~D1085. Usres can check the received data in these registers.</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1086</td><td>High word of the password in DVP-PCC01

(displayed in hex according to its ASCII codes)</td><td>0</td><td>-</td><td>-</td><td>R/W</td><td>NO</td><td>0</td></tr>
<tr><td>D1087</td><td>Low word of the password in DVP-PCC01 (displayed in hex according to its ASCII codes)</td><td>0</td><td>-</td><td>-</td><td>R/W</td><td>NO</td><td>0</td></tr>
<tr><td>D1089
↓
D1099</td><td>Sent data of Modbus communication.

When PLC's RS-485 communication instruction sends out data, the data will be stored in D1089~D1099. Users can check the sent data in these registers.</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1109*</td><td>COM3 (RS-485) Communication protoco</td><td>H'86</td><td>-</td><td>-</td><td>R/W</td><td>NO</td><td>H'86</td></tr>
<tr><td>D1110*</td><td>Average value of analog input channel 0 (AD 0) When average times in D1062 is set to 1, D1110 indicates present value.</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1111*</td><td>Average value of analog input channel 1 (AD 1) When average times in D1062 is set to 1, D1111 indicates present value</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1112*</td><td>Average value of analog input channel 2 (AD 2) Whenaverage times in D1062 is set to 1, D1112 indicates present value</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1113*</td><td>Average value of analog input channel 3 (AD 3) Whenaverage times in D1062 is set to 1, D1113 indicates present value</td><td>0</td><td>-</td><td>-</td><td>R</td><td>NO</td><td>0</td></tr>
<tr><td>D1114*</td><td>Setting the mode of analog input/output
(available for TP04P)<table><tr><td>Bit</td><td>11-10</td><td>9-8</td><td>7-6</td><td>5-4</td><td>3-2</td><td>1-0</td></tr><tr><td>Channel</td><td>CH5</td><td>CH4</td><td>CH3</td><td>CH2</td><td>CH1</td><td>CH0</td></tr></table>Setting the mode of input:</td><td>-</td><td>-</td><td>-</td><td>R/W</td><td>YES</td><td>0</td></tr>
</table>

Special D	Content	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
	00: Voltage mode 01: Current mode (0~20mA) 11: Current mode (4~20mA) Setting the mode of output: 00: Voltage mode 01: Current mode						
D1115*	Analog input/output mode setting (available for TP70P)	-	-	-	R/W	YES	0
D1116*	Output value of analog output channel 0 (DA 0)	0	0	0	R/W	NO	0
D1117*	Output value of analog output channel 1 (DA 1)	0	0	0	R/W	NO	0
D1118*	Sampling time of analog/digital converstion. Default: 2. Unit: 1ms. Sampling time will be regarded as 2ms if D1118≦2	2	-	-	R/W	YES	2
D1120*	COM2 (RS-485) communication protocol	H'86	-	-	R/W	NO	H'86
D1121*	COM1(RS-232) and COM2(RS-485) PLC communication address	-	-	-	R/W	YES	1
D1122	COM2(RS-485) Residual number of words of transmitting data	0	0	-	R	NO	0
D1123	COM2(RS-485) Residual number of words of the receiving data	0	0	-	R	NO	0
D1124	COM2(RS-485) Definition of start character (STX)	H'3A	-	-	R/W	NO	H'3A
D1125	COM2(RS-485) Definition of first ending character (ETX1)	H'0D	-	-	R/W	NO	H'0D
D1126	COM2(RS-485) Definition of second ending character (ETX2)	H'0A	-	-	R/W	NO	H'0A
D1129	COM2 (RS-485) Communication time-out setting (ms)	0	-	-	R/W	NO	0
D1130	COM2 (RS-485) Error code returning from Modbus	0	-	-	R	NO	0
D1137*	Address where incorrect use of operand occurs	0	0	-	R	NO	0
D1140	Number of Analog I/O modules (max. 1) (# => 1: TP04P-22XA11R / TP70P-22XA11R / TP04P-21EX11R/TP70P-21EX11R; 0: Other models)	-	-	-	R	NO	#
D1167	The specific end word to be detected for RS instruction to execute an interruption request (I140) on COM1 (RS-232).	0	-	-	R/W	NO	0
D1168	The specific end word to be detected for RS instruction to execute an interruption request (I150) on COM2 (RS-485)	0	-	-	R/W	NO	0
D1169	The specific end word to be detected for RS instruction to execute an interruption request (I160) on COM3 (RS-485)	0	-	-	R/W	NO	0
D1182	Index register E1	0	-	-	R/W	NO	0
D1183	Index register F1	0	-	-	R/W	NO	0
D1184	Index register E2	0	-	-	R/W	NO	0
D1185	Index register F2	0	-	-	R/W	NO	0
D1186	Index register E3	0	-	-	R/W	NO	0
D1187	Index register F3	0	-	-	R/W	NO	0

Special D	Content	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1188	Index register E4	0	-	-	R/W	NO	0
D1189	Index register F4	0	-	-	R/W	NO	0
D1190	Index register E5	0	-	-	R/W	NO	0
D1191	Index register F5	0	-	-	R/W	NO	0
D1192	Index register E6	0	-	-	R/W	NO	0
D1193	Index register F6	0	-	-	R/W	NO	0
D1194	Index register E7	0	-	-	R/W	NO	0
D1195	Index register F7	0	-	-	R/W	NO	0
D1240*	When interupt I400/I401/I100/I101 occurs, D1240 stores the low word of high-speed counter.	0	0	-	R	NO	0
D1241*	When interupt I400/I401/I100/I101 occurs, D1241 stores the high Word of the high-speed counter.	0	0	-	R	NO	0
D1249	Set value for COM1 (RS-232) data receiving time-out (Unit: 1ms, min. 50ms, value smaller than 50ms will be regarded as 50ms) (only applicable for MODRW/RS instruction) In RS instruction, no time-out setting if “0” is specified.	0	-	-	R/W	NO	0
D1250	COM1 (RS-232) communication error code (only applicable for MODRW/RS instruction)	0	-	-	R/W	NO	0
D1252	Set value for COM3 (RS-485) data receiving time-out (Unit: 1ms, min. 50ms, value smaller than 50ms will be regarded as 50ms) (only applicable for MODRW/RS instruction) In RS instruction, no time-out setting if “0” is specified	0	-	-	R/W	NO	0
D1253	COM3 (RS-485) communication error code (only applicable for MODRW/RS instruction)	0	-	-	R/W	NO	0
D1255*	COM3 (RS-485) PLC communication address	-	-	-	R/W	YES	1
D1256 ↓ D1295	For COM2 RS-485 MODRW instruction. D1256~D1295 store the sent data of MODRW instruction. When MODRW instruction sends out data, the data will be stored in D1256~D1295. Users can check the sent data in these registers.	0	-	-	R	NO	0
D1296 ↓ D1311	For COM2 RS-485 MODRW instruction. D1296~D1311 store the converted hex data from D1070 ~ D1085 (ASCII). PLC automatically converts the received ASCII data in D1070 ~ D1085 into hex data.	0	-	-	R	NO	0
D1313*	Second of RTC: 00 ~ 59	-	-	-	R/W	YES	0
D1314*	Minute of RTC: 00 ~ 59	-	-	-	R/W	YES	0
D1315*	Hour of RTC: 00 ~ 23	-	-	-	R/W	YES	0
D1316*	Day of RTC: 01 ~ 31	-	-	-	R/W	YES	1
D1317*	Month of RTC: 01 ~ 12	-	-	-	R/W	YES	1
D1318*	Week of RTC: 1 ~ 7	-	-	-	R/W	YES	2/5
D1319*	Year of RTC: 00 ~ 99 (A.D.)	-	-	-	R/W	YES	8/10
D1320	Analog I/O module code	-	-	-	R	NO	#

Special D	Content	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
	0X22: TP04P-22XA11R/TP70P-22XA11R 0X41: TP04P-21EX11R/TP70P-21EX11R						
D1354	PLC Link scan cycle (Unit: 1ms) ■ Max: K32000 ■ D1354 = K0 when PLC Link stops or when the first scan is completed	0	0	0	R	NO	0
D1355*	Starting reference for Master to read from Slave ID#1	-	-	-	R/W	YES	H'1064
D1356*	Starting reference for Master to read from Slave ID#2	-	-	-	R/W	YES	H'1064
D1357*	Starting reference for Master to read from Slave ID#3	-	-	-	R/W	YES	H'1064
D1358*	Starting reference for Master to read from Slave ID#4	-	-	-	R/W	YES	H'1064
D1359*	Starting reference for Master to read from Slave ID#5	-	-	-	R/W	YES	H'1064
D1360*	Starting reference for Master to read from Slave ID#6	-	-	-	R/W	YES	H'1064
D1361*	Starting reference for Master to read from Slave ID#7	-	-	-	R/W	YES	H'1064
D1362*	Starting reference for Master to read from Slave ID#8	-	-	-	R/W	YES	H'1064
D1363*	Starting reference for Master to read from Slave ID#9	-	-	-	R/W	YES	H'1064
D1364*	Starting reference for Master to read from Slave ID#10	-	-	-	R/W	YES	H'1064
D1365*	Starting reference for Master to read from Slave ID#11	-	-	-	R/W	YES	H'1064
D1366*	Starting reference for Master to read from Slave ID#12	-	-	-	R/W	YES	H'1064
D1367*	Starting reference for Master to read from Slave ID#13	-	-	-	R/W	YES	H'1064
D1368*	Starting reference for Master to read from Slave ID#14	-	-	-	R/W	YES	H'1064
D1369*	Starting reference for Master to read from Slave ID#15	-	-	-	R/W	YES	H'1064
D1370*	Starting reference for Master to read from Slave ID#16	-	-	-	R/W	YES	H'1064
D1399*	Starting ID of Slave designated by PLC LINK	-	-	-	R/W	YES	1
D1415*	Starting reference for Master to write in Slave ID#1	-	-	-	R/W	YES	H'10C8
D1416*	Starting reference for Master to write in Slave ID#2	-	-	-	R/W	YES	H'10C8
D1417*	Starting reference for Master to write in Slave ID#3	-	-	-	R/W	YES	H'10C8
D1418*	Starting reference for Master to write in Slave ID#4	-	-	-	R/W	YES	H'10C8
D1419*	Starting reference for Master to write in Slave ID#5	-	-	-	R/W	YES	H'10C8
D1420*	Starting reference for Master to write in Slave ID#6	-	-	-	R/W	YES	H'10C8
D1421*	Starting reference for Master to write in Slave ID#7	-	-	-	R/W	YES	H'10C8
D1422*	Starting reference for Master to write in Slave ID#8	-	-	-	R/W	YES	H'10C8
D1423*	Starting reference for Master to write in Slave ID#9	-	-	-	R/W	YES	H'10C8
D1424*	Starting reference for Master to write in Slave ID#10	-	-	-	R/W	YES	H'10C8
D1425*	Starting reference for Master to write in Slave ID#11	-	-	-	R/W	YES	H'10C8
D1426*	Starting reference for Master to write in Slave ID#12	-	-	-	R/W	YES	H'10C8
D1427*	Starting reference for Master to write in Slave ID#13	-	-	-	R/W	YES	H'10C8

Special D	Content	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1428*	Starting reference for Master to write in Slave ID#14	-	-	-	R/W	YES	H'10C8
D1429*	Starting reference for Master to write in Slave ID#15	-	-	-	R/W	YES	H'10C8
D1430*	Starting reference for Master to write in Slave ID#16	-	-	-	R/W	YES	H'10C8
D1431*	Times of PLC LINK polling cycle	0	-	-	R/W	NO	0
D1432*	Current times of PLC LINK polling cycle	0	-	-	R/W	NO	0
D1433*	Number of slave units linked to EASY PLC LINK	0	-	-	R/W	NO	0
D1434*	Data length to be read on Slave ID#1	-	-	-	R/W	YES	16
D1435*	Data length to be read on Slave ID#2	-	-	-	R/W	YES	16
D1436*	Data length to be read on Slave ID#3	-	-	-	R/W	YES	16
D1437*	Data length to be read on Slave ID#4	-	-	-	R/W	YES	16
D1438*	Data length to be read on Slave ID#5	-	-	-	R/W	YES	16
D1439*	Data length to be read on Slave ID#6	-	-	-	R/W	YES	16
D1440*	Data length to be read on Slave ID#7	-	-	-	R/W	YES	16
D1441*	Data length to be read on Slave ID#8	-	-	-	R/W	YES	16
D1442*	Data length to be read on Slave ID#9	-	-	-	R/W	YES	16
D1443*	Data length to be read on Slave ID#10	-	-	-	R/W	YES	16
D1444*	Data length to be read on Slave ID#11	-	-	-	R/W	YES	16
D1445*	Data length to be read on Slave ID#12	-	-	-	R/W	YES	16
D1446*	Data length to be read on Slave ID#13	-	-	-	R/W	YES	16
D1447*	Data length to be read on Slave ID#14	-	-	-	R/W	YES	16
D1448*	Data length to be read on Slave ID#15	-	-	-	R/W	YES	16
D1449*	Data length to be read on Slave ID#16	-	-	-	R/W	YES	16
D1450*	Data length to be written on Slave ID#1	-	-	-	R/W	YES	16
D1451*	Data length to be written on Slave ID#2	-	-	-	R/W	YES	16
D1452*	Data length to be written on Slave ID#3	-	-	-	R/W	YES	16
D1453*	Data length to be written on Slave ID#4	-	-	-	R/W	YES	16
D1454*	Data length to be written on Slave ID#5	-	-	-	R/W	YES	16
D1455*	Data length to be written on Slave ID#6	-	-	-	R/W	YES	16
D1456*	Data length to be written on Slave ID#7	-	-	-	R/W	YES	16
D1457*	Data length to be written on Slave ID#8	-	-	-	R/W	YES	16
D1458*	Data length to be written on Slave ID#9	-	-	-	R/W	YES	16
D1459*	Data length to be written on Slave ID#10	-	-	-	R/W	YES	16
D1460*	Data length to be written on Slave ID#11	-	-	-	R/W	YES	16
D1461*	Data length to be written on Slave ID#12	-	-	-	R/W	YES	16

Special D	Content	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1462*	Data length to be written on Slave ID#13	-	-	-	R/W	YES	16
D1463*	Data length to be written on Slave ID#14	-	-	-	R/W	YES	16
D1464*	Data length to be written on Slave ID#15	-	-	-	R/W	YES	16
D1465*	Data length to be written on Slave ID#16	-	-	-	R/W	YES	16
D1480* ↓ D1495*	The data which is read from slave ID#1 in the PLC LINK at the time when M1353 is OFF	0	-	-	R	NO	0
	The initial data register where the data read from slave ID#1~ID#16 in the PLC LINK is stored at the time when M1353 is ON	-	-	-	R	YES	0
D1496* ↓ D1511*	The data which is written into slave ID#1 in the PLC LINK at the time when M1353 is OFF	0	-	-	R/W	NO	0
	The initial data register where the data written into slave ID#1~ID#16 in the PLC LINK is stored at the time when M1353 is ON	-	-	-	R/W	YES	0
D1512* ↓ D1527*	The data which is read from slave ID#2 in the PLC LINK	0	-	-	R	NO	0
D1528* ↓ D1543*	The data which is written into slave ID#2 in the PLC LINK	0	-	-	R/W	NO	0
D1544* ↓ D1559*	The data which is read from slave ID#3 in the PLC LINK	0	-	-	R	NO	0
D1560* ↓ D1575*	The data which is written into slave ID#3 in the PLC LINK	0	-	-	R/W	NO	0
D1576* ↓ D1591*	The data which is read from slave ID#4 in the PLC LINK	0	-	-	R	NO	0
D1592* ↓ D1607*	The data which is written into slave ID#4 in the PLC LINK	0	-	-	R/W	NO	0
D1608* ↓ D1623*	The data which is read from slave ID#5 in the PLC LINK	0	-	-	R	NO	0
D1624* ↓ D1639*	The data which is written into slave ID#5 in the PLC LINK	0	-	-	R/W	NO	0
D1640* ↓ D1655*	The data which is read from slave ID#6 in the PLC LINK	0	-	-	R	NO	0
D1656* ↓ D1671*	The data which is written into slave ID#6 in the PLC LINK	0	-	-	R/W	NO	0
D1672* ↓ D1687*	The data which is read from slave ID#7 in the PLC LINK	0	-	-	R	NO	0
D1688* ↓ D1703*	The data which is written into slave ID#7 in the PLC LINK	0	-	-	R/W	NO	0

Special D	Content	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
D1704* ↓ D1719*	The data which is read from slave ID#8 in the PLC LINK	0	-	-	R	NO	0
D1720* ↓ D1735*	The data which is written into slave ID#8 in the PLC LINK	0	-	-	R/W	NO	0
D1736* ↓ D1751*	The data which is read from slave ID#9 in the PLC LINK	0	-	-	R	NO	0
D1752* ↓ D1767*	The data which is written into slave ID#9 in the PLC LINK	0	-	-	R/W	NO	0
D1768* ↓ D1783*	The data which is read from slave ID#10 in the PLC LINK	0	-	-	R	NO	0
D1784* ↓ D1799*	The data which is written into slave ID#10 in the PLC LINK	0	-	-	R/W	NO	0
D1800* ↓ D1815*	The data which is read from slave ID#11 in the PLC LINK	0	-	-	R	NO	0
D1816* ↓ D1831*	The data which is written into slave ID#11 in the PLC LINK	0	-	-	R/W	NO	0
D1832* ↓ D1847*	The data which is read from slave ID#12 in the PLC LINK	0	-	-	R	NO	0
D1848* ↓ D1863*	The data which is written into slave ID#12 in the PLC LINK	0	-	-	R/W	NO	0
D1864* ↓ D1879*	The data which is read from slave ID#13 in the PLC LINK	0	-	-	R	NO	0
D1880* ↓ D1895*	The data which is written into slave ID#13 in the PLC LINK	0	-	-	R/W	NO	0
D1896* ↓ D1911*	The data which is read from slave ID#14 in the PLC LINK	0	-	-	R	NO	0
D1900* ↓ D1931*	Specify the station number of Slaves for PLC-Link when M1356 is ON. Consecutive station numbers set by D1399 will be invalid in this case. Note that the registers are latched only when M1356 is ON.	0	-	-	R/W	NO	
D1912* ↓ D1927*	The data which is written into slave ID#14 in the PLC LINK	0	-	-	R/W	NO	0
D1928* ↓ D1943*	The data which is read from slave ID#15 in the PLC LINK	0	-	-	R	NO	0
D1944* ↓ D1959*	The data which is written into slave ID#15 in the PLC LINK	0	-	-	R/W	NO	0
D1960* ↓ D1975*	The data which is read from slave ID#16 in the PLC LINK	0	-	-	R	NO	0

Special D	Content	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch -ed	Default
D1976* ↓ D1991*	The data which is written into slave ID#16 in the PLC LINK	0	-	-	R/W	NO	0
D1994	Remaining times for PLC password setting on DVP-PCC01	0	-	-	R/W	NO	0
D1995	Data length for PLC ID Setting on DVP-PCC01	0	-	-	R/W	NO	0
D1996	1st Word of PLC ID Setting for DVP-PCC01 (Indicated by Hex format corresponding to ASCII codes)	0	-	-	R/W	NO	0
D1997	2nd Word of PLC ID Setting for DVP-PCC01 (Indicated by Hex format corresponding to ASCII codes)	0	-	-	R/W	NO	0
D1998	3rd Word of PLC ID Setting for DVP-PCC01 (Indicated by Hex format corresponding to ASCII codes)	0	-	-	R/W	NO	0
D1999	4th word of PLC ID Setting for DVP-PCC01 (Indicated by Hex format corresponding to ASCII codes)	0	-	-	R/W	NO	0
D4000 ↓ D4999	Present value of an object in the TP program D4000: Present value of object 1 D4001: Present value of object 2 … D4999: Present value of object 999	-	-	-	R/W	NO	0

C.2.1 PLC CPU Built-in Inputs and Outputs

- Analog inputs

Type \ Model		22XA1R/T	21EX1R/T
Voltage	-10V ~ +10V	-2000 ~ +2000	N/A
Current	-20mA ~ +20mA	-1000 ~ +1000	N/A
	+4mA ~ +20mA	+0 ~ +1000	+0 ~ +4000
	0~20mA	N/A	+0 ~ +4000
Temperature	Pt100 -20℃~300℃	N/A	-200 ~ +3000

- Analog outputs

Type \ Model		22XA1R/T	21EX1R/T
Voltage	-10V ~ +10V	-2000 ~ +2000	N/A
Current	+0mA ~ +20mA	+0 ~ +4000	+0 ~ +4000
	+4mA ~ +20mA	+0 ~ +4000	+0 ~ +4000

- Special devices for TP analog

Device	22XA1R/T	21EX1R/T
D1110	The present value of the analog input channel 0 (AD0)	The present value of the analog input channel 0 (AD0)
D1111	The present value of the analog input channel 1 (AD1)	The present value of the analog input channel 1 (AD1)
D1112	The present value of the analog input channel 2 (AD2)	The present value of the temperature input channel 3 (PT3)
D1113	The present value of the analog input channel 3 (AD3)	The present value of the temperature input channel 4 (PT4)
D1114	Mode setting for analog channel of TP04P	
D1115	Mode setting for analog channel of TP70P	
D1116	The present value of the analog output channel 4 (DA4)	The present value of the analog output channel 2 (DA2)
D1117	The present value of the analog output channel 4 (DA4)	N/A

Note: the temperature input channel only supports Pt100. No setup is needed.

- Device settings for 22XA1R/T

<table>
<tr><th colspan="7">D1114 (TP04P) / D1115 (TP70P)</th></tr>
<tr><td>Bit15-12</td><td>Bit11-10</td><td>Bit9-8</td><td>Bit7-6</td><td>Bit5-4</td><td>Bit3-2</td><td>Bit1-0</td></tr>
<tr><td>N/A</td><td>DA5</td><td>DA4</td><td>AD3</td><td>AD2</td><td>AD1</td><td>AD0</td></tr>
</table>

Value	AD mode	DA mode
00	-10V~10V	-10V~10V
01	-20mA~+20mA	0mA~20mA
10	4mA~20mA	4mA~20mA

- Device settings for 21EX1R/T

<table>
<tr><th colspan="6">D1114(TP04P) / D1115 (TP70P)</th></tr>
<tr><td>Bit15-10</td><td>Bit9-8</td><td>Bit7-6</td><td>Bit5-4</td><td>Bit3-2</td><td>Bit1-0</td></tr>
<tr><td>N/A</td><td>DA2</td><td>N/A</td><td>N/A</td><td>AD1</td><td>AD0</td></tr>
</table>

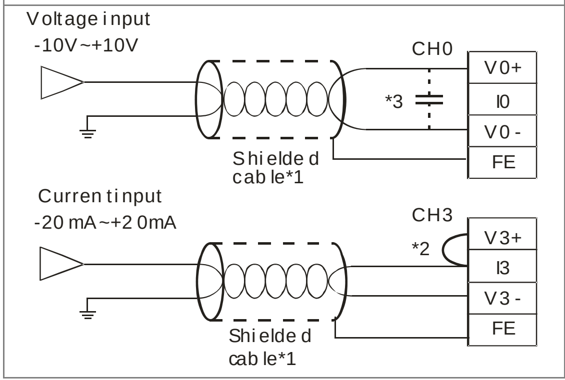
Value	AD mode	DA mode
00	0mA~20mA	0mA~20mA
01	4mA~20mA	4mA~20mA

Note: the temperature input channel only supports Pt100. No setup is needed.

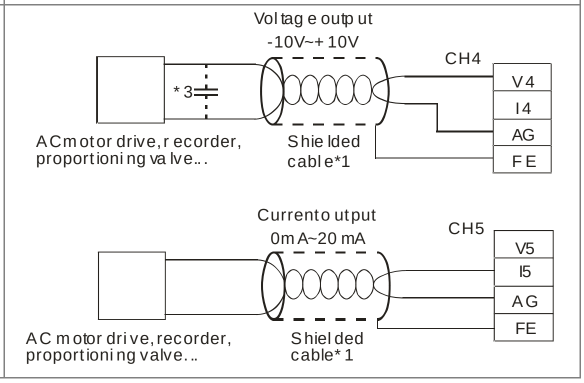
- Wiring Analog / Temperature Terminals

22XA1R/T

Analog Input

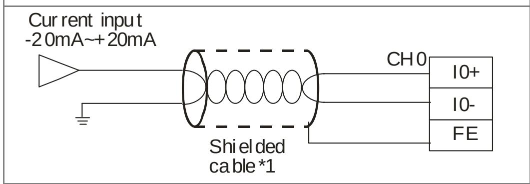


Analog Output

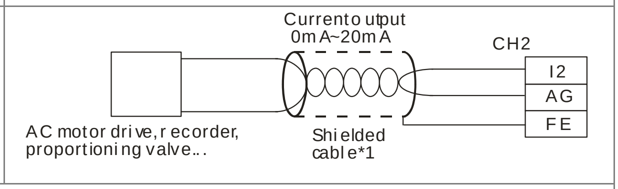


21EX1R/T

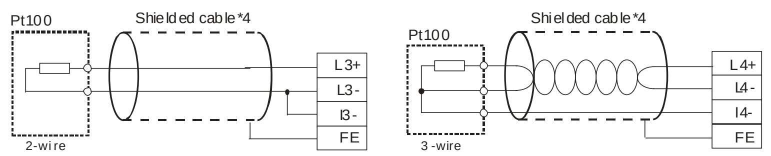
Analog input



Analog Output



Pt100



*1. Use shielded cables to isolate the analog input signal cable from other power cables.

*2. If the module is connected to a current signal, the terminals V3+ and I3 must be short-circuited.

*3. If variability in the input voltage results in interference within the wiring, connect the module to a capacitor with a capacitance between 0.1–0.47 μF and a working voltage of 25 V.

*4. To prevent too much noise and interference, connect the FE of the shielded cable to ground.

*5. Connect the ground terminal of a power supply module and the analog input terminal FE to the system ground and then ground the system ground or connect the system ground to a distribution box.

C.3 Special Auxiliary Relay

The types and functions of special auxiliary relays (special M) are listed in the table below. Care should be taken that some devices of the same No. may bear different meanings in different series MPUs. Columns marked with “R” refers to “read only”, “R/W” refers to “read and write”, “-“ refers to the status remains unchanged and “#” refers to that system will set it up according to the status of the PLC.

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1000*	Monitor normally open contact	OFF	ON	OFF	R	NO	OFF
M1001*	Monitor normally closed contact	ON	OFF	ON	R	NO	ON
M1002*	Enable single positive pulse at the moment when RUN is activate (Normally OFF)	OFF	ON	OFF	R	NO	OFF
M1003*	Enable single negative pulse at the moment when RUN is activate (Normally ON)	ON	OFF	ON	R	NO	ON
M1004*	ON when syntax errors occur	OFF	OFF	-	R	NO	OFF
M1008*	Watchdog timer (ON: PLC WDT time out)	OFF	OFF	-	R	NO	OFF
M1009	Indicate LV signal due to 24VDC insufficiency	OFF	-	-	R	NO	OFF
M1011*	10ms clock pulse, 5ms ON/5ms OFF	OFF	-	-	R	NO	OFF
M1012*	100ms clock pulse, 50ms ON / 50ms OFF	OFF	-	-	R	NO	OFF
M1013*	1s clock pulse, 0.5s ON / 0.5s OFF	OFF	-	-	R	NO	OFF
M1014*	1 min clock pulse, 30s ON / 30s OFF	OFF	-	-	R	NO	OFF
M1015*	Enable high-speed timer	OFF	-	-	R/W	NO	OFF
M1016*	Indicate Year display mode of RTC.	OFF	-	-	R/W	NO	OFF
M1017*	±30 seconds correction on real time clock	OFF	-	-	R/W	NO	OFF
M1018	Flag for Radian/Degree, ON for degree	OFF	-	-	R/W	NO	OFF
M1020	Zero flag	OFF	-	-	R	NO	OFF
M1021	Borrow flag	OFF	-	-	R	NO	OFF
M1022	Carry flag	OFF	-	-	R	NO	OFF
M1024	COM1 monitor request	OFF	-	-	R/W	NO	OFF
M1025*	Indicate incorrect request for communication	OFF	-	-	R	NO	OFF
M1026	RAMP mode selection	OFF	-	-	R/W	NO	OFF
M1027	PR output mode selection (8/16 bytes)	OFF	-	-	R/W	NO	OFF
M1028	Switch T64~T126 timer resulotion (10ms/100ms). ON =10ms	OFF	-	-	R/W	NO	OFF
M1031*	Clear all non-latched memory	OFF	-	-	R/W	NO	OFF

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1032*	Clear all latched memory	OFF	-	-	R/W	NO	OFF
M1033*	Output state latched at STOP	OFF	-	-	R/W	NO	OFF
M1034*	Disable all Y outputs	OFF	-	-	R/W	NO	OFF
M1035*	Enable X7 input point as RUN/STOP switch	-	-	-	R/W	YES	OFF
M1037*	Enable 8-sets SPD function (Has to be used with D1037)	OFF	OFF	OFF	R/W	NO	OFF
M1038	Switch T200~T255 timer resulotion (10ms/1ms). ON = 1ms	OFF	-	-	R/W	NO	OFF
M1039*	Fix scan time	OFF	-	-	R/W	NO	OFF
M1040	Disable step transition	OFF	-	-	R/W	NO	OFF
M1041	Step transition start	OFF	-	OFF	R/W	NO	OFF
M1042	Enable pulse operation	OFF	-	-	R/W	NO	OFF
M1043	Zero return completed	OFF	-	OFF	R/W	NO	OFF
M1044	Zero point condition	OFF	-	OFF	R/W	NO	OFF
M1045	Disable “all output reset” function	OFF	-	-	R/W	NO	OFF
M1046	Indicate STL status	OFF	-	-	R	NO	OFF
M1047	Enable STL monitoring	OFF	-	-	R/W	NO	OFF
M1048	Indicate alarm status	OFF	-	-	R	NO	OFF
M1049	Enable alarm monitoring	OFF	-	-	R/W	NO	OFF
M1050	Disable interruption I000 / I001	OFF	-	-	R/W	NO	OFF
M1051	Disable interruption I100 / I101	OFF	-	-	R/W	NO	OFF
M1058	COM3 monitor request	OFF	-	-	R/W	NO	OFF
M1059	Disable high-speed counter interruptions I010~I080	OFF	-	-	R/W	NO	OFF
M1060	System error message 1	OFF	-	-	R	NO	OFF
M1061	System error message 2	OFF	-	-	R	NO	OFF
M1062	System error message 3	OFF	-	-	R	NO	OFF
M1063	System error message 4	OFF	-	-	R	NO	OFF
M1064	Incorrect use of operands	OFF	OFF	-	R	NO	OFF
M1065	Syntax error	OFF	OFF	-	R	NO	OFF
M1066	Loop error	OFF	OFF	-	R	NO	OFF
M1067*	Program execution error	OFF	OFF	-	R	NO	OFF
M1068*	Execution error locked (D1068)	OFF	-	-	R	NO	OFF
M1072	PLC status (RUN/STOP), ON = RUN	OFF	ON	OFF	R/W	NO	OFF

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1075	Error occurring when write in Flash ROM	OFF	-	-	R	NO	OFF
M1080	COM2 monitor request	OFF	-	-	R/W	NO	OFF
M1081	Changing conversion mode for FLT instruction	OFF	-	-	R/W	NO	OFF
M1085	Selecting DVP-PCC01 duplicating function	OFF	-	-	R/W	NO	OFF
M1086	Enabling password function for DVP-PCC01	OFF	-	-	R/W	NO	OFF
M1088	Matrix comparison. Comparing between equivalent values (M1088 = ON) or different values (M1088 = OFF).	OFF	OFF	-	R/W	NO	OFF
M1089	Indicating the end of matrix comparison. When the comparison reaches the last bit, M1089 = ON.	OFF	OFF	-	R	NO	OFF
M1090	Indicating start of matrix comparison. When the comparison starts from the first bit, M1090 = ON.	OFF	OFF	-	R	NO	OFF
M1091	Indicating matrix searching results. When the comparison has matched results, comparison will stop immediately and M1091 = ON.	OFF	OFF	-	R	NO	OFF
M1092	Indicating pointer error. When the pointer Pr exceeds the comparison range, M1092 = ON	OFF	OFF	-	R	NO	OFF
M1093	Matrix pointer increasing flag. Adding 1 to the current value of the Pr.	OFF	OFF	-	R/W	NO	OFF
M1094	Matrix pointer clear flag. Clear the current value of the Pr to 0	OFF	OFF	-	R/W	NO	OFF
M1095	Carry flag for matrix rotation/shift/output.	OFF	OFF	-	R	NO	OFF
M1096	Borrow flag for matrix rotation/shift/input	OFF	OFF	-	R/W	NO	OFF
M1097	Direction flag for matrix rotation/displacement	OFF	OFF	-	R/W	NO	OFF
M1098	Counting the number of bits which are “1” or “0”	OFF	OFF	-	R/W	NO	OFF
M1099	ON when the bits counting result is “0”	OFF	OFF	-	R/W	NO	OFF
M1120*	Retaining the communication setting of COM2 (RS-485), modifying D1120 will be invalid when M1120 is set.	OFF	OFF	-	R/W	NO	OFF
M1121	For COM2(RS-485), data transmission ready	OFF	ON	-	R	NO	OFF
M1122	For COM2(RS-485), sending request	OFF	OFF	-	R/W	NO	OFF
M1123	For COM2(RS-485), data receiving completed	OFF	OFF	-	R/W	NO	OFF
M1124	For COM2(RS-485), data receiving ready	OFF	OFF	-	R/W	NO	OFF
M1125	For COM2(RS-485), communication ready status reset	OFF	OFF	OFF	R/W	NO	OFF
M1126	For COM2(RS-485), set STX/ETX as user defined or system defined	OFF	OFF	OFF	R/W	NO	OFF
M1127	For COM2(RS-485), data sending/receiving/converting completed. (RS instruction is not supported)	OFF	OFF	OFF	R/W	NO	OFF
M1128	For COM2(RS-485), Transmitting/Receiving status Indication	OFF	OFF	OFF	R/W	NO	OFF
M1129	For COM2(RS-485), receiving time out	OFF	OFF	-	R/W	NO	OFF

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1130	For COM2(RS-485), STX/ETX selection	OFF	OFF	-	R/W	NO	OFF
M1131	For COM2(RS-485), ON when MODRD/RDST/MODRW data is being converted from ASCII to Hex	OFF	OFF	-	R	NO	OFF
M1132	ON when there are no communication related instructions in the program	OFF	-	-	R	NO	OFF
M1136*	For COM3(RS-485/USB), retaining communication setting	OFF	-	-	R/W	NO	OFF
M1137	Retain DNET mapping data during non-executing period	-	-	-	R/W	NO	OFF
M1138*	For COM1 (RS-232), retaining communication setting. Modifying D1036 will be invalid when M1138 is set.	OFF	-	-	R/W	NO	OFF
M1139*	For COM1(RS-232), ASCII/RTU mode selection (OFF: ASCII; ON: RTU)	OFF	-	-	R/W	NO	OFF
M1140	For COM2 (RS-485), MODRD / MODWR / MODRW data receiving error	OFF	OFF	-	R	NO	OFF
M1141	For COM2 (RS-485), MODRD / MODWR / MODRW parameter error	OFF	OFF	-	R	NO	OFF
M1142	Data receiving error of VFD-A handy instructions	OFF	OFF	-	R	NO	OFF
M1143*	For COM2(RS-485), ASCII/RTU mode selection (OFF: ASCII; ON: RTU)	OFF	-	-	R/W	NO	OFF
M1161	8/16 bit mode (ON = 8 bit mode)	OFF	-	-	R/W	NO	OFF
M1162	Switching between decimal integer and binary floating point for SCLP instruction. ON: binary floating point; OFF: decimal integer	OFF	-	-	R/W	NO	OFF
M1167	16-bit mode for HKY input	OFF	-	-	R/W	NO	OFF
M1168	Designating work mode of SMOV	OFF	-	-	R/W	NO	OFF
M1177	Enable the communication instruction for Delta VFD series inverter. ON: VFD-A (Default), OFF: other models of VFD	OFF	-	-	R/W	NO	OFF
M1200	C200 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1201	C201 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1202	C202 counting mode ON: count down)	OFF	-	-	R/W	NO	OFF
M1203	C203 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1204	C204 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1205	C205 counting mode (ON :count down)	OFF	-	-	R/W	NO	OFF
M1206	C206 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1207	C207 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1208	C208 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1209	C209 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1210	C210 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1211	C211 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1212	C212 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1213	C213 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1214	C214 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1215	C215 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1216	C216 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1217	C217 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1218	C218 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1219	C219 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1220	C220 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1221	C221 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1222	C222 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1223	C223 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1224	C224 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1225	C225 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1226	C226 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1227	C227 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1228	C228 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1229	C229 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1230	C230 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1231	C231 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1232	C232 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
	C232 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1233	C233 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1234	C234 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1235	C235 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1236	C236 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1237	C237 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1238	C238 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1239	C239 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1240	C240 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1241	C241 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1242	C242 counting mode (ON: count down)	OFF	-	-	R/W	NO	OFF
M1243	C243 Reset function control. ON = R function disabled	OFF	-	-	R/W	NO	OFF
M1244	C244 Reset function control. ON = R function disabled	OFF	-	-	R/W	NO	OFF
M1245	C245 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1246	C246 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1247	C247 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1248	C248 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1249	C249 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1250	C250 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1251	C251 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1252	C252 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1253	C253 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1254	C254 counter monitor (ON: count down)	OFF	-	-	R	NO	OFF
M1270	C235 counting mode (ON: falling-edge count)	OFF	-	-	R/W	NO	OFF
M1271	C236 counting mode ON: falling-edge count)	OFF	-	-	R/W	NO	OFF
M1272	C237 counting mode (ON: falling-edge count)	OFF	-	-	R/W	NO	OFF
M1273	C238 counting mode (ON: falling-edge count)	OFF	-	-	R/W	NO	OFF
M1274	C239 counting mode (ON: falling-edge count)	OFF	-	-	R/W	NO	OFF
M1275	C240 counting mode (ON: falling-edge count)	OFF	-	-	R/W	NO	OFF
M1276	C241 counting mode (ON: falling-edge count)	OFF	-	-	R/W	NO	OFF
M1277	C242 counting mode (ON: falling-edge count)	OFF	-	-	R/W	NO	OFF
M1280*	For I000 / I001, reverse interrupt trigger pulse direction (Rising/Falling)	OFF	OFF	-	R/W	NO	OFF
M1284*	For I400 / I401, reverse interrupt trigger pulse direction (Rising/Falling)	OFF	OFF	-	R/W	NO	OFF
M1286*	For I600 / I601, reverse interrupt trigger pulse direction (Rising/Falling)	OFF	OFF	-	R/W	NO	OFF
M1303	High / low bits exchange for XCH instruction	OFF	-	-	R/W	NO	OFF
M1304*	Enable force-ON/OFF of input point X	OFF	-	-	R/W	NO	OFF
M1312	For COM1(RS-232), sending request (Only applicable for MODRW and RS instruction)	OFF	OFF	-	R/W	NO	OFF
M1313	For COM1(RS-232), ready for data receiving (Only applicable for MODRW and RS instruction)	OFF	OFF	-	R/W	NO	OFF
M1314	For COM1(RS-232), data receiving completed (Only applicable for MODRW and RS instruction)	OFF	OFF	-	R/W	NO	OFF

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1315	For COM1(RS-232), data receiving error (Only applicable for MODRW and RS instruction)	OFF	OFF	-	R/W	NO	OFF
M1316	For COM3(RS-485), sending request (Only applicable for MODRW and RS instruction)	OFF	OFF	-	R/W	NO	OFF
M1317	For COM3(RS-485), ready for data receiving (Only applicable for MODRW and RS instruction)	OFF	OFF	-	R/W	NO	OFF
M1318	For COM3(RS-485), data receiving completed (Only applicable for MODRW and RS instruction)	OFF	OFF	-	R/W	NO	OFF
M1319	For COM3(RS-485), data receiving error (Only applicable for MODRW and RS instruction)	OFF	OFF	-	R/W	NO	OFF
M1320*	For COM3 (RS-485), ASCII/RTU mode selection. (OFF: ASCII; ON: RTU)	OFF	-	-	R/W	NO	OFF
M1350*	Enable PLC LINK	OFF	-	OFF	R/W	NO	OFF
M1351*	Enable auto mode on PLC LINK	OFF	-	-	R/W	NO	OFF
M1352*	Enable manual mode on PLC LINK	OFF	-	-	R/W	NO	OFF
M1353*	Enable access up to 50 words through PLC LINK (If M1353 is ON, D1480~D1511 are latched devices.)	-	-	-	R/W	YES	OFF
M1354*	Enable simultaneous data read/write in a polling of PLC LINK	-	-	-	R/W	YES	OFF
M1355*	Select Slave linking mode in PLC LINK (ON: manual; OFF: auto-detection)	-	-	-	R/W	YES	OFF
M1356*	Enable station number selection function. When both M1353 and M1356 are ON, the user can specify the station number in D1900~D1931	-	-	-	R/W	YES	OFF
M1360*	Slave ID#1 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1361*	Slave ID#2 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1362*	Slave ID#3 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1363*	Slave ID#4 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1364*	Slave ID#5 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1365*	Slave ID#6 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1366*	Slave ID#7 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1367*	Slave ID#8 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1368*	Slave ID#9 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1369*	Slave ID#10 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1370*	Slave ID#11 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1371*	Slave ID#12 status on PLC LINK network	-	-	-	R/W	YES	OFF

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1372*	Slave ID#13 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1373*	Slave ID#14 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1374*	Slave ID#15 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1375*	Slave ID#16 status on PLC LINK network	-	-	-	R/W	YES	OFF
M1376*	Indicate Slave ID#1 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1377*	Indicate Slave ID#2 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1378*	Indicate Slave ID#3 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1379*	Indicate Slave ID#4 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1380*	Indicate Slave ID#5 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1381*	Indicate Slave ID#6 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1382*	Indicate Slave ID#7 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1383*	Indicate Slave ID#8 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1384*	Indicate Slave ID#9 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1385*	Indicate Slave ID#10 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1386*	Indicate Slave ID#11 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1387*	Indicate Slave ID#12 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1388*	Indicate Slave ID#13 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1389*	Indicate Slave ID#14 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1390*	Indicate Slave ID#15 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1391*	Indicate Slave ID#16 data interchange status on PLC LINK	OFF	-	-	R	NO	OFF
M1392*	Slave ID#1 linking error	OFF	-	-	R	NO	OFF
M1393*	Slave ID#2 linking error	OFF	-	-	R	NO	OFF
M1394*	Slave ID#3 linking error	OFF	-	-	R	NO	OFF
M1395*	Slave ID#4 linking error	OFF	-	-	R	NO	OFF
M1396*	Slave ID#5 linking error	OFF	-	-	R	NO	OFF
M1397*	Slave ID#6 linking error	OFF	-	-	R	NO	OFF
M1398*	Slave ID#7 linking error	OFF	-	-	R	NO	OFF
M1399*	Slave ID#8 linking error	OFF	-	-	R	NO	OFF
M1400*	Slave ID#9 linking error	OFF	-	-	R	NO	OFF
M1401*	Slave ID#10 linking error	OFF	-	-	R	NO	OFF
M1402*	Slave ID#11 linking error	OFF	-	-	R	NO	OFF

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1403*	Slave ID#12 linking error	OFF	-	-	R	NO	OFF
M1404*	Slave ID#13 linking error	OFF	-	-	R	NO	OFF
M1405*	Slave ID#14 linking error	OFF	-	-	R	NO	OFF
M1406*	Slave ID#15 linking error	OFF	-	-	R	NO	OFF
M1407*	Slave ID#16 linking error	OFF	-	-	R	NO	OFF
M1408*	Indicate that reading from Slave ID#1 is completed	OFF	-	-	R	NO	OFF
M1409*	Indicate that reading from Slave ID#2 is completed	OFF	-	-	R	NO	OFF
M1410*	Indicate that reading from Slave ID#3 is completed	OFF	-	-	R	NO	OFF
M1411*	Indicate that reading from Slave ID#4 is completed	OFF	-	-	R	NO	OFF
M1412*	Indicate that reading from Slave ID#5 is completed	OFF	-	-	R	NO	OFF
M1413*	Indicate that reading from Slave ID#6 is completed	OFF	-	-	R	NO	OFF
M1414*	Indicate that reading from Slave ID#7 is completed	OFF	-	-	R	NO	OFF
M1415*	Indicate that reading from Slave ID#8 is completed	OFF	-	-	R	NO	OFF
M1416*	Indicate that reading from Slave ID#9 is completed	OFF	-	-	R	NO	OFF
M1417*	Indicate that reading from Slave ID#10 is completed	OFF	-	-	R	NO	OFF
M1418*	Indicate that reading from Slave ID#11 is completed	OFF	-	-	R	NO	OFF
M1419*	Indicate that reading from Slave ID#12 is completed	OFF	-	-	R	NO	OFF
M1420*	Indicate that reading from Slave ID#13 is completed	OFF	-	-	R	NO	OFF
M1421*	Indicate that reading from Slave ID#14 is completed	OFF	-	-	R	NO	OFF
M1422*	Indicate that reading from Slave ID#15 is completed	OFF	-	-	R	NO	OFF
M1423*	Indicate that reading from Slave ID#16 is completed	OFF	-	-	R	NO	OFF
M1424*	Indicate that writing to Slave ID#1 is completed	OFF	-	-	R	NO	OFF
M1425*	Indicate that writing to Slave ID#2 is completed	OFF	-	-	R	NO	OFF
M1426*	Indicate that writing to Slave ID#3 is completed	OFF	-	-	R	NO	OFF
M1427*	Indicate that writing to Slave ID#4 is completed	OFF	-	-	R	NO	OFF
M1428*	Indicate that writing to Slave ID#5 is completed	OFF	-	-	R	NO	OFF
M1429*	Indicate that writing to Slave ID#6 is completed	OFF	-	-	R	NO	OFF
M1430*	Indicate that writing to Slave ID#7 is completed	OFF	-	-	R	NO	OFF
M1431*	Indicate that writing to Slave ID#8 is completed	OFF	-	-	R	NO	OFF
M1432*	Indicate that writing to Slave ID#9 is completed	OFF	-	-	R	NO	OFF
M1433*	Indicate that writing to Slave ID#10 is completed	OFF	-	-	R	NO	OFF

Special M	Function	OFF ⇩ ON	STOP ⇩ RUN	RUN ⇩ STOP	Attrib.	Latch-ed	Default
M1434*	Indicate that writing to Slave ID#11 is completed	OFF	-	-	R	NO	OFF
M1435*	Indicate that writing to Slave ID#12 is completed	OFF	-	-	R	NO	OFF
M1436*	Indicate that writing to Slave ID#13 is completed	OFF	-	-	R	NO	OFF
M1437*	Indicate that writing to Slave ID#14 is completed	OFF	-	-	R	NO	OFF
M1438*	Indicate that writing to Slave ID#15 is completed	OFF	-	-	R	NO	OFF
M1439*	Indicate that writing to Slave ID#16 is completed	OFF	-	-	R	NO	OFF

C.4 Instructions applicable to TP

The instructions which are applicable to TP are listed below. Please refer to chapter 3 for more information about the instructions.

C.4.1 Basic Instructions

Instruction	Function
LD	Load NO contact
LDI	Load NC contact
AND	Connect NO contact in series
ANI	Connect NC contact in series
OR	Connect NO contact in parallel
ORI	Connect NC contact in parallel
ANB	Connect a block in series
ORB	Connect a block in parallel
MPS	Start of branches. Stores current result of program evaluation
MRD	Reads the stored current result from previous MPS
MPP	End of branches. Pops (reads and resets) the stored result in previous MPS
OUT	Output coil
SET	Latches the ON status
RST	Resets contacts, registers or coils
MC	Master control Start
MCR	Master control Reset
END	Program End
NOP	No operation
P	Pointer
I	Interrupt program pointer
STL	Step ladder start instruction
RET	Step ladder return instruction
NP	Negative contact to Positive contact
PN	Positive contact to Negative contact

C.4.2 Numerical List of Instructions

Classification	API	Mnemonic		PULSE	Function
		16 bits	32 bits		
Loop Control	**00**	CJ	–	✓	Conditional jump
	01	CALL	–	✓	Call subroutine
	02	SRET	–	–	Subroutine return
	03	IRET	–	–	Interrupt return
	04	EI	–	–	Enable interrupt
	05	DI	–	–	Disable interrupt
	06	FEND	–	–	The end of the main program (First end)
	07	WDT	–	✓	Watchdog timer refresh
	08	FOR	–	–	Start of a For-Next Loop
	09	NEXT	–	–	End of a For-Next Loop
Transmission Comparison	**10**	CMP	DCMP	✓	Compare
	11	ZCP	DZCP	✓	Zone compare
	12	MOV	DMOV	✓	Move
	13	SMOV	–	✓	Shift move

Classification	API	Mnemonic		PULSE	Function
		16 bits	32 bits		
	14	CML	DCML	✓	Complement
	15	BMOV	–	✓	Block move
	16	FMOV	DFMOV	✓	Fill move
	17	XCH	DXCH	✓	Exchange
	18	BCD	DBCD	✓	Convert BIN to BCD
	19	BIN	DBIN	✓	Convert BCD to BIN
Four Arithmetic Operations	20	ADD	DADD	✓	Addition
	21	SUB	DSUB	✓	Subtraction
	22	MUL	DMUL	✓	Multiplication
	23	DIV	DDIV	✓	Division
	24	INC	DINC	✓	Increment
	25	DEC	DDEC	✓	Decrement
	26	WAND	DAND	✓	Logical Word AND
	27	WOR	DOR	✓	Logical Word OR
	28	WXOR	DXOR	✓	Logical XOR
	29	NEG	DNEG	✓	2's Complement (Negation)
Rotation and Displacement	30	ROR	DROR	✓	Rotate right
	31	ROL	DROL	✓	Rotate left
	32	RCR	DRCR	✓	Rotate right with carry
	33	RCL	DRCL	✓	Rotate left with carry
	34	SFTR	–	✓	Bit shift right
	35	SFTL	–	✓	Bit shift left
	36	WSFR	–	✓	Word shift right
	37	WSFL	–	✓	Word shift left
	38	SFWR	–	✓	Shift register write
Rotation and Displacement	39	SFRD	–	✓	Shift register read
Data Processing	40	ZRST	–	✓	Zone reset
	41	DECO	–	✓	Decode
	42	ENCO	–	✓	Encode
	43	SUM	DSUM	✓	Sum of Active bits
	44	BON	DBON	✓	Check specified bit status
	45	MEAN	DMEAN	✓	Mean
	46	ANS	–	–	Timed Annunciator Set
	47	ANR	–	✓	Annunciator Reset
	48	SQR	DSQR	✓	Square Root
	49	FLT	DFLT	✓	Floating point
High Speed Processing	53	–	DHSCS	–	High speed counter SET
	54	–	DHSCR	–	High speed counter RESET
	55	–	DHSZ	–	High speed zone compare
Handy Instructions	60	IST	–	–	Initial state
	61	SER	DSER	✓	Search a data stack
	62	ABSD	DABSD	–	Absolute drum sequencer

Classification	API	Mnemonic		PULSE	Function
		16 bits	32 bits		
	63	INCD	–	–	Incremental drum sequencer
	64	TTMR	–	–	Teaching timer
	65	STMR	–	–	Special timer
	66	ALT	–	✓	Alternate state
	67	RAMP	–	–	Ramp variable value
	69	SORT	–	–	Data sort
Serial I/O	80	RS	–	–	Serial communication
	82	ASCI	–	✓	Convert HEX to ASCII
	83	HEX	–	✓	Convert ASCII to HEX
	87	ABS	DABS	✓	Absolute value
	88	PID	DPID	–	PID control
Basic Instructions	89	PLS	–	–	Rising-edge output
	90	LDP	–	–	Rising–edge detection operation
	91	LDF	–	–	Falling–edge detection operation
	92	ANDP	–	–	Rising-edge series connection
	93	ANDF	–	–	Falling-edge series connection
	94	ORP	–	–	Rising-edge parallel connection
	95	ORF	–	–	Falling-edge parallel connection
	96	TMR	–	–	Timer
	97	CNT	DCNT	–	Counter
	98	INV	–	–	Inverse operation
	99	PLF	–	–	Falling-edge output
Communication Instructions	100	MODRD	–	–	Read Modbus data
	101	MODWR	–	–	Write Modbus Data
	102	FWD	–	–	Forward Operation of VFD
	103	REV	–	–	Reverse Operation of VFD
Communication Instructions	104	STOP	–	–	Stop VFD
	105	RDST	–	–	Read VFD Status
	106	RSTEF	–	–	Reset Abnormal VFD
	107	LRC	–	✓	LRC checksum
	108	CRC	–	✓	CRC checksum
	150	MODRW	–	–	MODBUS Read/ Write
	206	ASDRW	–	–	ASDA servo drive R/W
Floating Point Operation	110	–	DECMP	✓	Floating point compare
	111	–	DEZCP	✓	Floating point zone compare
	112	–	DMOVR	✓	Move floating point data
	116	–	DRAD	✓	Degree → Radian
	117	–	DDEG	✓	Radian → Degree
	118	–	DEBCD	✓	Float to scientific conversion
	119	–	DEBIN	✓	Scientific to float conversion
	120	–	DEADD	✓	Floating point addition
	121	–	DESUB	✓	Floating point subtraction
	122	–	DEMUL	✓	Floating point multiplication

Classification	API	Mnemonic		PULSE	Function
		16 bits	32 bits		
	123	–	DEDIV	✓	Floating point division
	124	–	DEXP	✓	Float exponent operation
	125	–	DLN	✓	Float natural logarithm operation
	126	–	DLOG	✓	Float logarithm operation
	127	–	DESQR	✓	Floating point square root
	128	–	DPOW	✓	Floating point power operation
	129	INT	DINT	✓	Float to integer
	130	–	DSIN	✓	Sine
	131	–	DCOS	✓	Cosine
	132	–	DTAN	✓	Tangent
	133	–	DASIN	✓	Arc Sine
	134	–	DACOS	✓	Arc Cosine
	135	–	DATAN	✓	Arc Tangent
	172	–	DADDR	✓	Floating point addition
	173	–	DSUBR	✓	Floating point subtraction
	174	–	DMULR	✓	Floating point multiplication
	175	–	DDIVR	✓	Floating point division
Additional Instruction	143	DELAY	–	✓	Delay
	144	GPWM	–	–	General PWM output
	147	SWAP	DSWAP	✓	Byte swap
	154	RAND	–	✓	Random number
	168	MVM	DMVM	✓	Mask and combine designated Bits
	176	MMOV	–	✓	16-bit→32-bit Conversion
	179	WSUM	DWSUM	✓	Sum of multiple devices
	202	SCAL	–	✓	Proportional value calculation
	203	SCLP	–	✓	Parameter proportional value calculation
	205	CMPT	DCMPT	✓	Compare table
Positioning Control	155	–	DABSR	–	Absolute position read
Real Time Calendar	160	TCMP	–	✓	Time compare
	161	TZCP	–	✓	Time Zone Compare
	162	TADD	–	✓	Time addition
	163	TSUB	–	✓	Time subtraction
	166	TRD	–	✓	Time read
	167	TWR	–	✓	Time write
	169	HOUR	DHOUR	–	Hour meter
Gray Code	170	GRY	DGRY	✓	BIN → Gray Code
	171	GBIN	DGBIN	✓	Gray Code → BIN
Matrix Operation	180	MAND	–	✓	Matrix AND
	181	MOR	–	✓	Matrix OR
	182	MXOR	–	✓	Matrix XOR
	183	MXNR	–	✓	Matrix XNR
	184	MINV	–	✓	Matrix inverse
	185	MCMP	–	✓	Matrix compare
	186	MBRD	–	✓	Matrix bit read
	187	MBWR	–	✓	Matrix bit write

Classification	API	Mnemonic		PULSE	Function
		16 bits	32 bits		
	188	MBS	–	✓	Matrix bit shift
	189	MBR	–	✓	Matrix bit rotate
	190	MBC	–	✓	Matrix bit status count
Contact Type Logic Operation	215	LD&	DLD&	–	S_1 & S_2
	216	LD|	DLD|	–	$S_1 \mid S_2$
	217	LD^	DLD^	–	S_1 ^ S_2
	218	AND&	DAND&	–	S_1 & S_2
	219	AND|	DAND|	–	$S_1 \mid S_2$
	220	AND^	DAND^	–	S_1 ^ S_2
	221	OR&	DOR&	–	S_1 & S_2
	222	OR|	DOR|	–	$S_1 \mid S_2$
	223	OR^	DOR^	–	S_1 ^ S_2
Contact Type Comparison	224	LD=	DLD=	–	$S_1 = S_2$
	225	LD>	DLD>	–	$S_1 > S_2$
	226	LD<	DLD<	–	$S_1 < S_2$
	228	LD<>	DLD<>	–	$S_1 \neq S_2$
	229	LD<=	DLD<=	–	$S_1 \leqq S_2$
	230	LD>=	DLD>=	–	$S_1 \geqq S_2$
	232	AND=	DAND=	–	$S_1 = S_2$
	233	AND>	DAND>	–	$S_1 > S_2$
	234	AND<	DAND<	–	$S_1 < S_2$
	236	AND<>	DAND<>	–	$S_1 \neq S_2$
	237	AND<=	DAND<=	–	$S_1 \leqq S_2$
Contact Type Comparison	238	AND>=	DAND>=	–	$S_1 \geqq S_2$
	240	OR=	DOR=	–	$S_1 = S_2$
	241	OR>	DOR>	–	$S_1 > S_2$
	242	OR<	DOR<	–	$S_1 < S_2$
	244	OR<>	DOR<>	–	$S_1 \neq S_2$
Contact Type Comparison	245	OR<=	DOR<=	–	$S_1 \leqq S_2$
	246	OR>=	DOR>=	–	$S_1 \geqq S_2$
Specific Bit Control	266	BOUT	DBOUT	–	Output specified bit of a word
	267	BSET	DBSET	–	Set ON specified bit of a word
	268	BRST	DBRST	–	Reset specified bit of a word
	269	BLD	DBLD	–	Load NO contact by specified bit
	270	BLDI	DBLDI	–	Load NC contact by specified bit
	271	BAND	DBAND	–	Connect NO contact in series by specified bit
	272	BANI	DBANI	–	Connect NC contact in series by specified bit
	273	BOR	DBOR	–	Connect NO contact in parallel by specified bit
	274	BORI	DBORI	–	Connect NC contact in parallel by specified bit
Floating-Point Contact Type Comparison	275	–	FLD=	–	$S_1 = S_2$
	276	–	FLD>	–	$S_1 > S_2$
	277	–	FLD<	–	$S_1 < S_2$
	278	–	FLD<>	–	$S_1 \neq S_2$
	279	–	FLD<=	–	$S_1 \leqq S_2$
	280	–	FLD>=	–	$S_1 \geqq S_2$

Classification	API	Mnemonic 16 bits	Mnemonic 32 bits	PULSE	Function
	281	–	FAND=	–	$S_1 = S_2$
	282	–	FAND>	–	$S_1 > S_2$
	283	–	FAND<	–	$S_1 < S_2$
	284	–	FAND<>	–	$S_1 \neq S_2$
	285	–	FAND<=	–	$S_1 \leqq S_2$
	286	–	FAND>=	–	$S_1 \geqq S_2$
	287	–	FOR=	–	$S_1 = S_2$
	288	–	FOR>	–	$S_1 > S_2$
	289	–	FOR<	–	$S_1 < S_2$
	290	–	FOR<>	–	$S_1 \neq S_2$
	291	–	FOR<=	–	$S_1 \leqq S_2$
	292	–	FOR>=	–	$S_1 \geqq S_2$

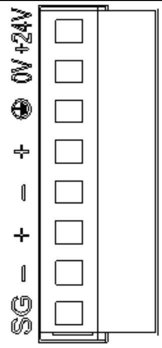
C.4.3 Additional Remarks on High-speed Instructions

1. TP only supports the high-speed inputs X0 and X1 (10KHz). (Please refer to section 2.12 for more information.) TP04P-08TP1R does not support high-speed inputs (only supports up to 500HZ).
2. TP only supports the software counters C235 and C236. The corresponding high-speed interrupts are I010 and I020. (Please refer to the explanations of API53 and API55 for more information.)
3. TP onlyt supports the hardware counter C251. The corresponding high-speed interrupt is I010. There is only one hardware comparator. (Please refer to the explanations of API53 and API55 for more information.)

C.5 Definitions of TP04P Communication Ports

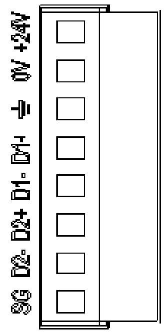
- TP04P-16TP1R/1T、TP04P-32TP1R/T、TP04P-22XA1R/T、TP04P-21EX1R/T

Name	Function
+24	+24V
0V	0V
⏚	FE
+	RS485 COM2 PLC Mode
-	
+	RS485 COM3 TP Mode
-	
SG	SG



- TP04P-20EXL1T、TP04P-08TP1R

Name	Function
+24	+24V
0V	0V
⏚	FE
D1+	RS485 COM2 PLC Mode
D1-	
D2+	RS485 COM3 TP Mode
D2-	
SG	SG



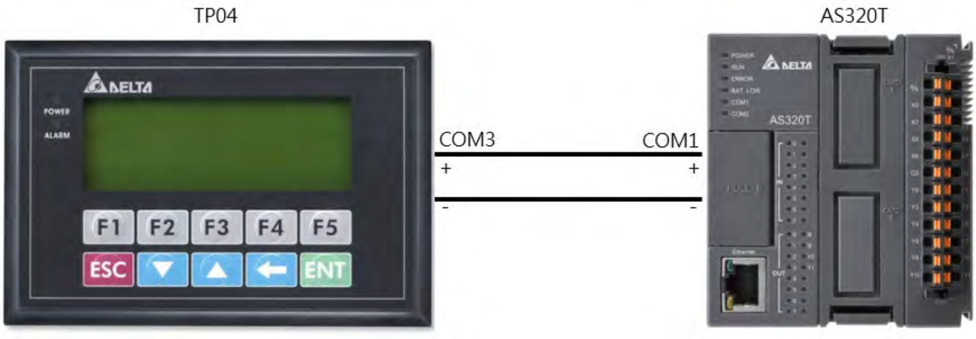
Communication port	**Descriptions**	
COM1	Type	USB
	Mode	Uploading or downloading programs
	Description	Not supporting PLC CPU communication
COM2	Type	RS-485
	Mode	Supporting PLC mode
	Description	Data length: 7 bits or 8 bits Stop bit: 1 bit or 2 bits Parity check: None/Odd/Even Transmission rate: 9,600 bps~115,200 bps
COM3	Type	RS-485
	Mode	Supporting TP mode
	Description	Data length: 7 bits or 8 bits Stop bit: 1 bit or 2 bits Parity check: None/Odd/Even Transmission rate: 9,600 bps~115,200 bps

C.6 Example of Setting up the Communication

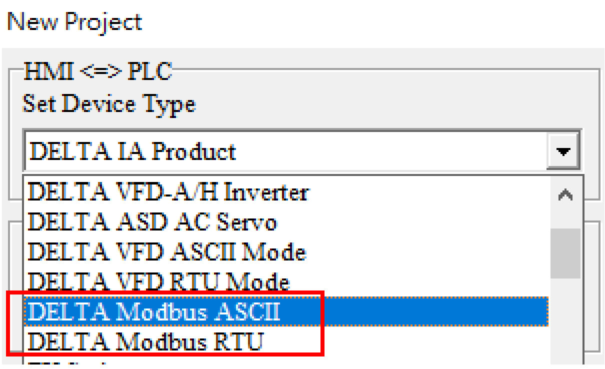
C.6.1 Communication with AH/AS Series PLC

The following uses AS Series PLC as a demonstration example.

1. Configuration diagram

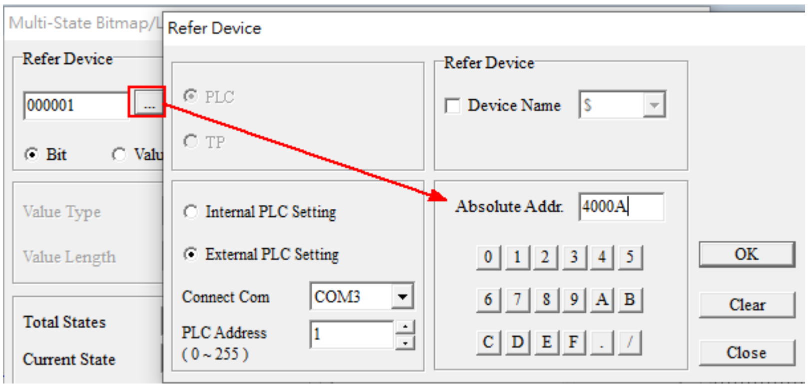


2. Select a driver (DELTA Modbus ASCII or DELTA Modbus RTU) as required.

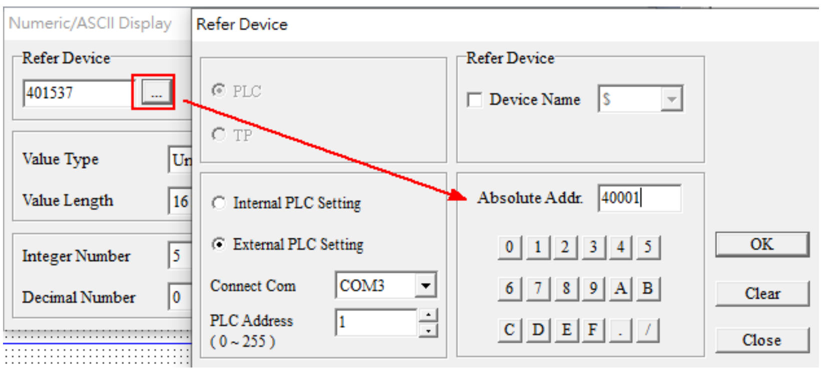


3. If you need to check the D0 and M0 statuses of AS Series PLC on TP04, you' ll need to add a number "4" as a prefix in the refer device address of the TP object.

Device	Type	AS Series Address (Hex)	TP
M10	Bit	000A	4000A



Device	Type	AS Series Address (Hex)	TP
D1	Word	0001	40001



Refer to section B.1

Standard Modbus Device Address from AS Series Hardware and Operation Manual for more information.

MEMO

Appendix

Introducing the Current Consumption of Slim PLCs/Extension Modules

Contents

D.1 Current Consumption of a Slim PLC/an Extension Module

Users can calculate the maximum current consumed by the combination of a slim PLC and modules by means of the data in the table below.

D.1.1 Current supply and current consumption of a PLC (+24VDC)

Model	Internal maximum current consumed (mA)	Maximum current consumed by the external DIO (A) (The current consumption of all inputs and outputs is calculated.) #1
12SA211R	100	5.1
12SA211T	70	2.1
12SE11R	110	5.1
12SE11T	80	2.1
12SS211S	50	2.1
14SS211R	100	9.1
14SS211T	50	3.1
20SX211R	220	9.1
20SX211S	170	1.9
20SX211T	170	3.1
24SV11T2	170	3.8
26SE11R	160	18.1
26SE11T	90	6.1
26SE11S	90	6.1
28SA211R	155	18.1
28SA211T	85	6.1
28SS211R	150	18.1
28SS211T	85	6.1
28SV11R2	210	18.1
28SV11S2	170	3.8
28SV11T2	170	3.8

#1: The external maximum current consumed is estimated on the basis of a worst condition. It is suggested that users should calculate the maximum current consumed according to the actual arrangement.

D.1.2 Current supply and current consumption of a digital input/output module (+24VDC)

Model	Internal maximum current consumed by the IO-BUS (mA)	Maximum current consumed by the external DIO (A)
06SN11R	82	36
08SM10N	40	0.08
08SM11N	15	0.05
08SN11R	55	5
08SN11T	55	1.2
08SN11TS	15	2
08SP11R	35	5
08SP11T	35	1.2
08SP11TS	15	1.25
08ST11N	55	0
16SM11N	25	0.1
16SN11T	65	1.2
16SN11TS	30	2
16SP11R	65	5
16SP11T	65	1.2
16SP11TS	30	2
32SM11N	40	0.16
32SN11TN	40	2

D.1.3 Current consumption of a special input/output module (+24VDC)

A special input/output module must be supplied with +24VDC power.

Model	Internal maximum current consumed by the IO-BUS (mA)	Maximum current consumed by the external DIO (A)
01PU-S	100	105
02DA-S	30	125
02TUL-S	75	40
02TUN-S	55	600
02TUR-S	60	600
04AD-S	30	83
04AD-S2	30	83
04DA-S	30	167
04DA-S2	30	167
04PT-S	30	83
04TC-S	30	83
06AD-S	30	83
06PT-S	83	-
06XA-S	30	83
06XA-S2	30	83
08NTC-S	30	-

DT01-S	25	29
PF01-S	55	-
02TKL-S	75	-
02TKN-S	55	-
02TKR-S	60	-

D.1.4 Current consumption of a left-side high-speed special module (+24VDC)

Model	Internal maximum current consumed by the IO-BUS (mA)	Maximum current consumed by the external DIO (A)
01LC-SL	40	125
02LC-SL	40	125
02PU-SL	45	-
04AD-SL	40	15
04DA-SL	40	80
COPM-SL	50	0
DNET-SL	50	26
201LC-SL	100	-
202LC-SL	140	-
211LC-SL	150	4030
EN01-SL	62.5	0
PF02-SL	60	-
SCM12-SL	62.5	-
SCM52-SL	62.5	-

D.1.5 Calculating the maximum current consumed by a system

Example: 28SV2 + 16SP + 04AD-S + 04TC-S + EN01-SL

The power module optionally purchased is DVPPS02. (It supplies 2A current.)

Model	Internal current consumption	External current consumption
DVP28SV11T2	170mA	3.8A
DVP16SP11R	65mA	5A
DVP04AD-S	30mA	83mA
DVP04TC-S	30mA	83mA
DVPEN01-SL	60mA	0

Maximum current consumed: Internal → 170 + 65 + 30 + 30 + 60 = 355 (mA) < 2A Pass

External → 3.8A + 5A + 83mA + 83mA = 9A > 2A Not pass

Conclusion: The 2A current supplied by DVPPS02 is sufficient for the PLC and the special modules. If the external I/O terminals are connected to loads, it is suggested that users should purchase an extra power module.

Appendix

E

Communication of DVP Series Slim Type Special Modules

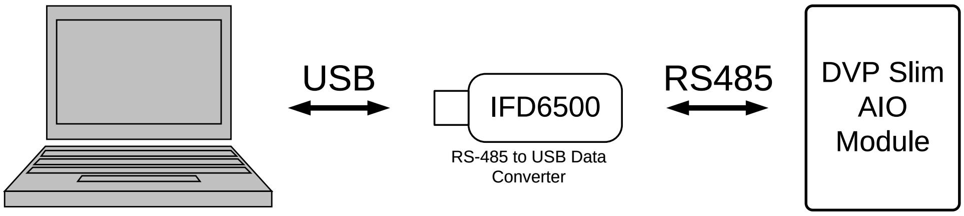
Contents

E.1 DVP Series Slim Type Special Modules

Series	Modules
DVP Slim Type Series	DVP04AD-S, DVP02DA-S, DVP04DA-S, DVP06XA-S, DVP06AD-S, DVP04TC-S, DVP04PT-S, DVP06PT-S, DVP04AD-S2, DVP04DA-S2, DVP06XA-S2

E.2 Connections of a Slim Type Special Module (Work alone)

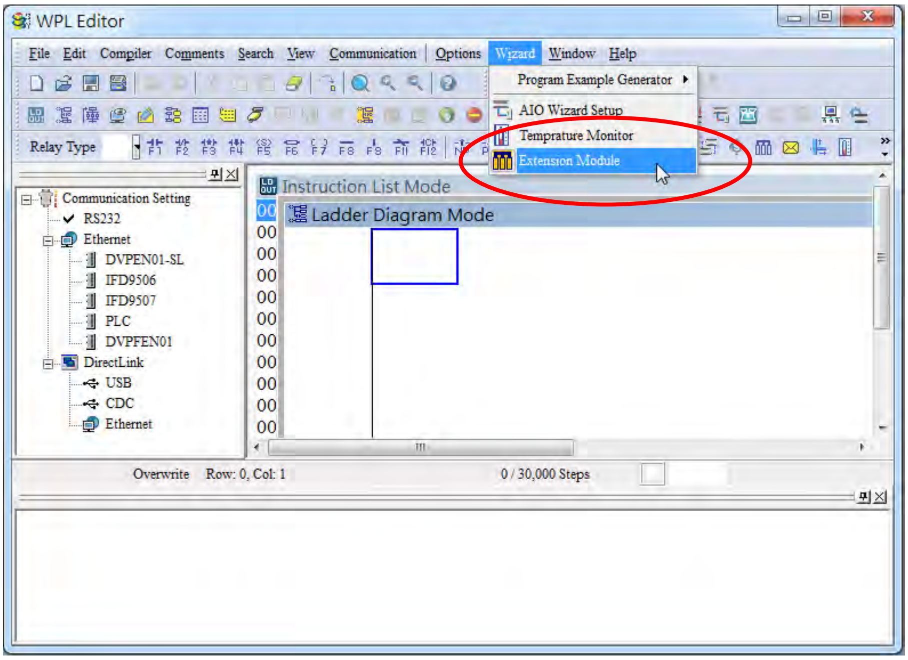
See the following connection example for reference when using a slim type special module alone.



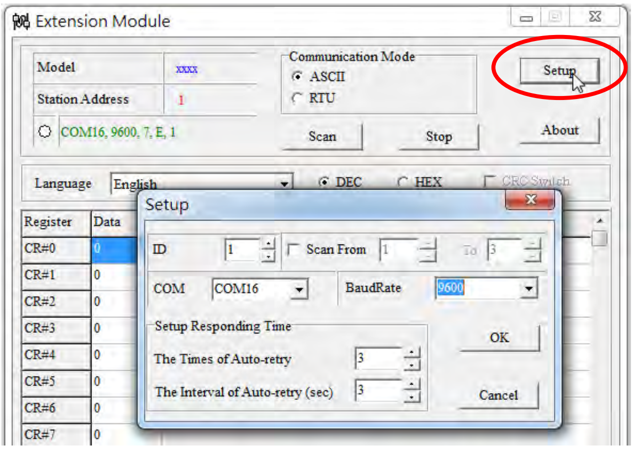
E.3 Using WPL Editor

You can use the option **Extension Module** to check or modify the control registers (CR) of the slim type special module.

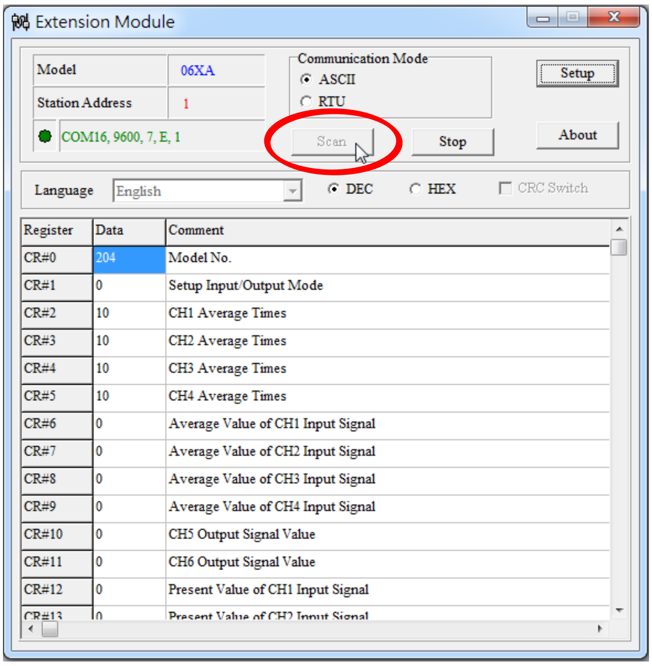
Step 1: Click **Wizard > Extension Module** to open the setting page.



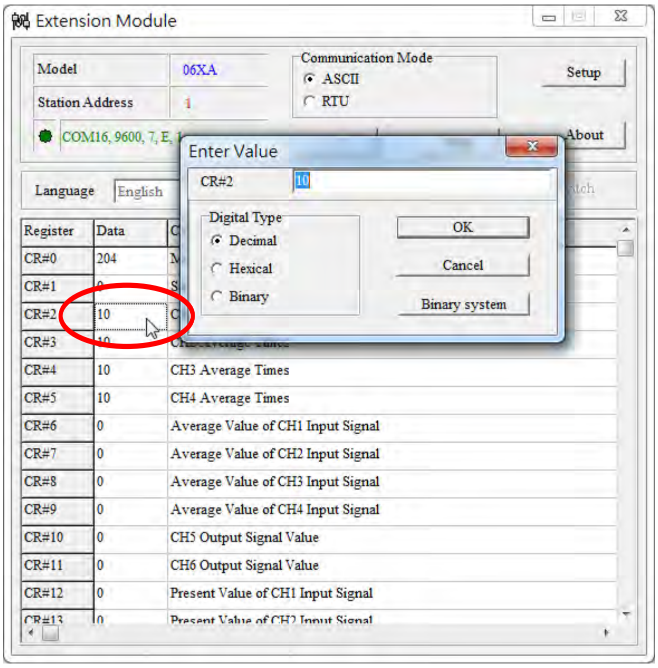
Step 2: Make sure the module is supplied with power and is connected to RS485 before setting. Click **Setup** and start to set up the COM port and the baud rate. After the setup is done, click **OK** to save the setting.



Step 3: Click **Scan** to connect to the module and monitor the current values of the control registers



Step 4: Double-click the CR that you need to modify. A setting window appears and you can start to modify its value. If the CR is a latched type, its last value can be retained even after power-off.



Step 5: After monitoring or editing, you can click **Stop** to end the communication.

Appendix F

General Specifications

Contents

F.1 ES2 and EX2 Series - General Specifications

This specification is applicable to DVP-ES2/EX2 PLC CPU and their compatible extension modules.

Item	Specifications
Operating temperature	0 to 55°C
Storage temperature	-25 to 70°C
Operating humidity	5–95% No condensation
Storage humidity	5–95% No condensation
Work environment	No corrosive gas exists.
Installation location	In a control box
Pollution degree	2
Ingress protection (IP ratings)	IP20
Surge voltage withstand level	1,500 VAC (Primary-secondary), 1,500 VAC (Primary-PE), 500 VAC (Secondary-PE)
Insulation voltage	Above 5MΩ The voltage between all inputs/outputs and the ground is 500 VDC.
Noise Immunity	ESD: IEC 61000-4-2 8KV Air Discharge, 4KV Contact Discharge EFT: IEC 61000-4-4 Power Line: 2KV, Digital I/O: 1KV, Analog & Communication I/O: 1KV RS: IEC 61000-4-3 80MHz ~ 1GHz, 10V/m Surge(IEC 61131-2, IEC 61000-4-5) : AC power: CM ±2 kV, DM ±1 kV DC power: CM ±0.5 kV, DM ±0.5 kV Analog I/O, RS-232(shielded): CM ±1 kV Digital I/O, RS-485 (unshielded): CM ±1 kV
Ground	The diameter of the ground should not be less than the diameters of the cables connected to the terminals L and N. It is required to use grounding if more than one PLC is being used at the same time.
Vibration / Shock resistance	Vibration: IEC 60068-2-6(test Fc), 1g DIN rail mount Shock resistance: IEC 60068-2-27(test Ea), 15 g, 11 ms pulse, 6 shocks in each of 3 axes
Ambient air temperature-barometric pressure-altitude	Operating: 1013 ~ 795hPa (0 ~ 2000 m) Storage:1013 ~ 700hPa (0 ~ 3000m)
UL 94 Flame Classifications	UL 94 V-0

F.2 SA2, SS2, SX2, SE Series - General Specifications

This specification is applicable to DVP-SA2/SS2/SX2/SE PLC CPU and their compatible extension modules.

Item	Specifications
Operating temperature	0 to 55°C
Storage temperature	-25 to 70°C
Operating humidity	5–95% No condensation
Storage humidity	5–95% No condensation
Work environment	No corrosive gas exists.
Installation location	In a control box
Pollution degree	2
Ingress protection (IP ratings)	IP20
Surge voltage withstand level	1,500 VAC (Primary-secondary), 1,500 VAC (Primary-PE), 500 VAC (Secondary-PE)
Insulation voltage	Above 5MΩ The voltage between all inputs/outputs and the ground is 500 VDC.
Noise Immunity	ESD: IEC 61000-4-2 8KV Air Discharge, 4KV Contact Discharge EFT: IEC 61000-4-4 Power Line: 2KV, Digital I/O: 1KV, Analog & Communication I/O: 1KV RS: IEC 61000-4-3 80MHz ~ 1GHz, 10V/m Surge(IEC 61131-2, IEC 61000-4-5) : DC power: CM ±0.5 kV, DM ±0.5 kV Analog I/O, RS-232, USB (shielded): CM ±1 kV Digital I/O, RS-485 (unshielded): CM ±1 kV
Ground	The diameter of the ground should not be less than the diameters of the cables connected to the terminals L and N. It is required to use grounding if more than one PLC is being used at the same time.
Vibration / Shock resistance	Vibration: IEC 60068-2-6(test Fc), 1g DIN rail mount Shock resistance: IEC 60068-2-27(test Ea), 15 g, 11 ms pulse, 6 shocks in each of 3 axes
Ambient air temperature-barometric pressure-altitude	Operating: 1013 ~ 795hPa (0 ~ 2000 m) Storage:1013 ~ 700hPa (0 ~ 3000m)

F.3 EC5 Series - General Specifications

This specification is applicable to DVP-EC5 Series PLC CPU and its compatible extension modules.

Item	Specifications
Operating temperature	0 to 55°C
Storage temperature	-25 to 70°C
Operating humidity	5–95% No condensation
Storage humidity	5–95% No condensation
Work environment	No corrosive gas exists.
Installation location	In a control box
Pollution degree	2
Ingress protection (IP ratings)	IP20
Surge voltage withstand level	2,300 VAC (Primary-secondary), 1,350 VAC (Primary-PE), 500 VAC (Secondary-PE)
Insulation voltage	Above 5MΩ The voltage between all inputs/outputs and the ground is 500 VDC.
Noise Immunity	ESD: IEC 61000-4-2 8KV Air Discharge, 6KV Contact Discharge EFT: IEC 61000-4-4 Power Line: 4KV, Digital I/O: 2KV, Analog & Communication I/O: 2KV RS: IEC 61000-4-3 26MHz ~ 1GHz, 10V/m Surge (IEC 61131-2, IEC 61000-4-5): AC power: CM ±2 kV, DM ±1 kV, RS-232(shielded): CM ±1 kV digital I/O, RS-485 (unshielded): CM ±1 kV
Ground	The diameter of the ground should not be less than the diameters of the cables connected to the terminals L and N. It is required to use grounding if more than one PLC is being used at the same time.
Vibration / Shock resistance	Vibration: IEC 60068-2-6(test Fc), 1g DIN rail mount Shock resistance: IEC 60068-2-27(test Ea), 15 g, 11 ms pulse, 6 shocks in each of 3 axes
Ambient air temperature-barometric pressure-altitude	Operating: 1013 ~ 795hPa (0 ~ 2000 m) Storage:1013 ~ 700hPa (0 ~ 3000m)
UL 94 Flame Classifications	UL 94 V-0

F.4 TP Series - General Specifications

This specification is applicable to DVP-TP Series PLC CPU and their compatible extension modules.

Item	Specifications
Operating temperature	0 to 55°C
Storage temperature	-20 to 60°C
Operating humidity	20–90% RH (No condensation)
Work environment	No corrosive gas exists.
Installation location	In a control box
Pollution degree	2
Ingress protection of front panel (IP ratings)	IP66/NEMA4/UL Type4X (indoor use only)
Insulation voltage	Above 5MΩ The voltage between all inputs/outputs and the ground is 500 VDC.
Noise Immunity	ESD (IEC 61131-2, IEC 61000-4-2): 8KV Air Discharge, 4KV Contact Discharge EFT (IEC 61131-2, IEC 61000-4-4): Power Line: 2KV, Digital I/O: 1KV, Analog & Communication I/O: 1KV RS (IEC 61131-2, IEC 61000-4-3): 80MHz ~ 1GHz, 10V/m Surge (IEC 61131-2, IEC 61000-4-5): DC power: CM ±0.5 kV, DM ±0.5 kV, Analog I/O, digital I/O, communication I/O: CM ±1 kV, AC IO (Relay Output): CM ±2 kV, DM ±1 kV
Ground	The diameter of the ground should not be less than the diameters of the cables connected to the terminals L and N. It is required to use grounding if more than one PLC is being used at the same time.
Vibration / Shock resistance	Vibration: IEC 60068-2-6(test Fc), 1g DIN rail mount Shock resistance: IEC 60068-2-27(test Ea), 15 g, 11 ms pulse, 6 shocks in each of 3 axes
Cooling Method	Natural air-cooling

MEMO

Smarter. Greener. Together.

Industrial Automation Headquarters

Delta Electronics, Inc.
Taoyuan Technology Center
No.18, Xinglong Rd., Taoyuan District,
Taoyuan City 330477, Taiwan
TEL: +886-3-362-6301 / FAX: +886-3-371-6301

Asia

Delta Electronics (Shanghai) Co., Ltd.
No.182 Minyu Rd., Pudong Shanghai, P.R.C.
Post code : 201209
TEL: +86-21-6872-3988 / FAX: +86-21-6872-3996
Customer Service: 400-820-9595

Delta Electronics (Japan), Inc.
Industrial Automation Sales Department
2-1-14 Shibadaimon, Minato-ku
Tokyo, Japan 105-0012
TEL: +81-3-5733-1155 / FAX: +81-3-5733-1255

Delta Electronics (Korea), Inc.
1511, 219, Gasan Digital 1-Ro., Geumcheon-gu,
Seoul, 08501 South Korea
TEL: +82-2-515-5305 / FAX: +82-2-515-5302

Delta Energy Systems (Singapore) Pte Ltd.
4 Kaki Bukit Avenue 1, #05-04, Singapore 417939
TEL: +65-6747-5155 / FAX: +65-6744-9228

Delta Electronics (India) Pvt. Ltd.
Plot No.43, Sector 35, HSIIDC Gurgaon,
PIN 122001, Haryana, India
TEL: +91-124-4874900 / FAX: +91-124-4874945

Delta Electronics (Thailand) PCL.
909 Soi 9, Moo 4, Bangpoo Industrial Estate (E.P.Z),
Pattana 1 Rd., T.Phraksa, A.Muang,
Samutprakarn 10280, Thailand
TEL: +66-2709-2800 / FAX: +66-2709-2827

Delta Electronics (Australia) Pty Ltd.
Unit 2, Building A, 18-24 Ricketts Road,
Mount Waverley, Victoria 3149 Australia
Mail: IA.au@deltaww.com
TEL: +61-1300-335-823 / +61-3-9543-3720

Americas

Delta Electronics (Americas) Ltd.
5101 Davis Drive, Research Triangle Park, NC 27709, U.S.A.
TEL: +1-919-767-3813 / FAX: +1-919-767-3969

Delta Electronics Brazil Ltd.
Estrada Velha Rio-São Paulo, 5300 Eugênio de
Melo - São José dos Campos CEP: 12247-004 - SP - Brazil
TEL: +55-12-3932-2300 / FAX: +55-12-3932-237

Delta Electronics International Mexico S.A. de C.V.
Gustavo Baz No. 309 Edificio E PB 103
Colonia La Loma, CP 54060
Tlalnepantla, Estado de México
TEL: +52-55-3603-9200

EMEA

Delta Electronics (Netherlands) B.V.
Sales: Sales.IA.EMEA@deltaww.com
Marketing: Marketing.IA.EMEA@deltaww.com
Technical Support: iatechnicalsupport@deltaww.com
Customer Support: Customer-Support@deltaww.com
Service: Service.IA.emea@deltaww.com
TEL: +31(0)40 800 3900

Delta Electronics (Netherlands) B.V.
Automotive Campus 260, 5708 JZ Helmond, The Netherlands
Mail: Sales.IA.Benelux@deltaww.com
TEL: +31(0)40 800 3900

Delta Electronics (Netherlands) B.V.
Coesterweg 45, D-59494 Soest, Germany
Mail: Sales.IA.DACH@deltaww.com
TEL: +49 2921 987 238

Delta Electronics (France) S.A.
ZI du bois Challand 2, 15 rue des Pyrénées,
Lisses, 91090 Evry Cedex, France
Mail: Sales.IA.FR@deltaww.com
TEL: +33(0)1 69 77 82 60

Delta Electronics Solutions (Spain) S.L.U
Ctra. De Villaverde a Vallecas, 265 1º Dcha Ed.
Hormigueras – P.I. de Vallecas 28031 Madrid
TEL: +34(0)91 223 74 20

Carrer Llacuna 166, 08018 Barcelona, Spain
Mail: Sales.IA.Iberia@deltaww.com

Delta Electronics (Italy) S.r.l.
Via Meda 2–22060 Novedrate(CO)
Piazza Grazioli 18 00186 Roma Italy
Mail: Sales.IA.Italy@deltaww.com
TEL: +39 039 8900365

Delta Greentech Elektronik San. Ltd. Sti. (Turkey)
Şerifali Mah. Hendem Cad. Kule Sok. No:16-A
34775 Ümraniye – İstanbul
Mail: Sales.IA.Turkey@deltaww.com
TEL: + 90 216 499 9910

Eltek Dubai (Eltek MEA DMCC)
OFFICE 2504, 25th Floor, Saba Tower 1,
Jumeirah Lakes Towers, Dubai, UAE
Mail: Sales.IA.MEA@deltaww.com
TEL: +971(0)4 2690148

DVP-0139720-15
2023/12/22